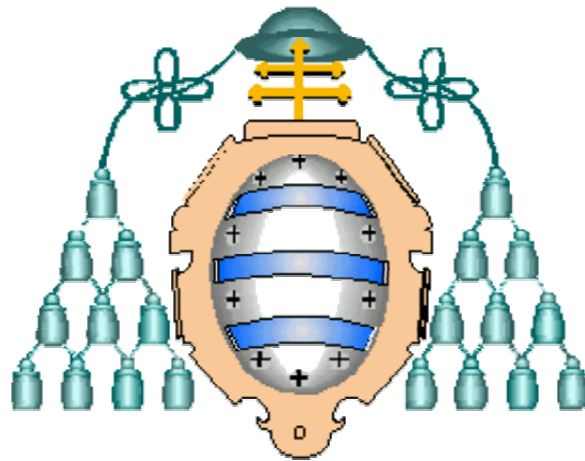




Universidad de Oviedo

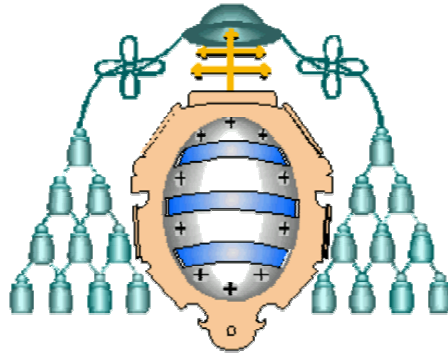
Escuela Universitaria de Ingeniería Técnica Industrial

Ingeniería Técnica Industrial, Especialidad Electrónica
Industrial

**Plataforma modular educativa para
Aprendizaje de microcontroladores**

Adrián Bulnes Parra

30 de junio de 2010



Universidad de Oviedo

Escuela Universitaria de Ingeniería Técnica Industrial

Ingeniería Técnica Industrial, Especialidad Electrónica
Industrial

**Plataforma modular educativa para
aprendizaje de microcontroladores**

Documento Memoria

Departamento de Ingeniería Eléctrica,
Electrónica, de Computadores y Sistemas
Autor: Adrián Bulnes Parra
Tutor: Miguel Ángel José Prieto

Gijón, 30 de junio de 2010

Fdo.: Adrián Bulnes Parra

1. DEFINICIONES Y ABREVIATURAS.....	5
2. INTRODUCCIÓN	6
2.1. OBJETIVOS	6
2.2. CONTEXTO	7
2.3. ALCANCE	8
2.4. VISIÓN GENERAL.....	8
3. DESCRIPCIÓN DEL PROYECTO.....	10
3.1. ORGANIZACIÓN.....	10
3.2. NOMENCLATURA.....	10
3.3. FUNCIONES DEL PRODUCTO.....	12
3.4. LENGUAJES DE PROGRAMACIÓN Y COMPILADORES	12
3.4.1 Para microcontroladores.....	12
3.4.2 Para ordenadores	13
3.4.3 Scripts para consola	13
3.5. CARACTERÍSTICAS DE LOS USUARIOS.....	14
4. CIRCUITOS ELECTRÓNICOS.....	15
4.1. ESTÁNDARES.....	16
4.1.1 Conector de datos	16
4.1.2 Conector a alimentación.....	18
4.1.3 Conector de programación.....	19
4.2. PLACAS PRINCIPALES.....	22
4.2.1 MBP8	22
4.2.2 MBP14	23
4.2.3 MBP18	24
4.2.4 MBP40	25
4.3. MÓDULOS	27
4.3.1 LTIND-A	27
4.3.2 SISW-SPST.....	28
4.3.3 LTIL-A.....	29
4.3.4 SIBW-1Y.....	30
4.3.5 AO-SPK.....	31

4.3.6	<i>SIDST-GP2</i>	31
4.3.7	<i>LT7S-HN</i>	33
4.3.8	<i>CMIR-RC</i>	34
4.3.9	<i>MC2A</i>	34
4.3.10	<i>MC2B</i>	36
4.3.11	<i>PWR-DD5</i>	37
4.3.12	<i>PWR-AD5</i>	41
4.3.13	<i>CMSP-MAX</i>	45
4.4.	MÓDULOS DESARROLLADOS POR TERCEROS	47
4.4.1	<i>LTIND-FourSquare</i>	47
4.4.2	<i>Double SIBW</i>	47
4.4.3	<i>AO-SPK-Perfboard</i>	48
4.4.4	<i>PWREG5</i>	48
4.4.5	<i>No-Wires Module Adapter</i>	48
5.	HERRAMIENTAS INFORMÁTICAS	49
5.1.	ENTORNO DE DESARROLLO	49
5.2.	LIBRERÍAS	51
5.3.	SCRIPTS PARA LÍNEA DE COMANDOS	52
6.	APLICACIONES DE EJEMPLO	53
6.1.	ELECTROOCULÓGRAFO	53
6.2.	“KETJO” - OCTÓPODO	54
6.3.	CLAQUETA DIGITAL	55
6.4.	“PCBOT”	57
6.5.	APLICACIONES DESARROLLADAS POR TERCEROS	59
6.5.1	“POV” – <i>Beltrán Fernández</i>	59
6.5.2	“Manu-fakture” – <i>Manuel Fernández</i>	60
6.5.3	“Octapas” – <i>Jorge Aparicio</i>	61
6.5.4	“CoyáN 2.0” – <i>Rodrigo Fernández</i>	61
7.	PUBLICACIONES, CONCURSOS, TALLERES	62
7.1.	IV CUSL	62
7.2.	ARTÍCULO EN LINUX MAGAZINE	62

7.3.	TALLERES DE ROBÓTICA	63
8.	CALENDARIO	64
9.	BIBLIOGRAFÍA.....	66

1. Definiciones y abreviaturas

- **Bootloader:** programa parcial incluido en un microcontrolador junto con el programa original escrito por el usuario, normalmente utilizado para programar y depurar el propio microcontrolador a través del puerto serie.
- **ICSP:** *In-Circuit Serial Programming*. Protocolo y conector definido por Microchip para programar sus microcontroladores.
- **IDE:** *Integrated Development Environment*. Entorno de Desarrollo Integrado. Programa informático para ordenador que facilita el desarrollo de programas para ordenadores o microcontroladores. Normalmente ofrece ayudas en la edición de código fuente y automatiza las tareas de compilación, grabación en MCU y testeo.
- **“La plataforma”, “esta plataforma”:** Se refiere al propio proyecto, “Plataforma modular educativa para aprendizaje de microcontroladores”.
- **MCU:** *Micro-Controller Unit*. Microcontrolador.
- **Mil, mils:** Milésimas de pulgada. Medida muy habitual en el diseño de circuitos electrónicos. Los agujeros de las placas perforadas y las patillas de los encapsulados PDIP están separadas entre sí 100mils, es decir, 2.54mm.
- **Nivel alto/bajo:** los componentes o variables digitales/lógicas pueden tomar dos valores: alto/bajo o 1/0, que normalmente se corresponden con la tensión de alimentación para el estado alto (típicamente 5V) y 0V en estado bajo.
- **PCB:** *Printed Circuit Board*. Placa de circuito impreso.
- **PDIP:** *Parallel Dual-Inline Package*. Encapsulado de inserción muy habitual. Todos los circuitos integrados de este proyecto que tienen más de tres patillas utilizan este tipo de encapsulado.
- **Vrms:** Tensión eficaz (voltios eficaces).

2. Introducción

2.1. Objetivos

El objetivo principal del proyecto es elaborar una plataforma electrónica modular y un conjunto de herramientas informáticas para ayudar en el aprendizaje de desarrollo de aplicaciones electrónicas basadas en microcontroladores, o para ser utilizada por desarrolladores con conocimientos para desarrollar aplicaciones o prototipos de manera rápida y sencilla.

La plataforma debe cumplir los siguientes requisitos:

- Todos los módulos electrónicos se ofrecerán con su esquema electrónico y, en caso de que sea necesario, con esquema de distribución de pistas y componentes, diagramas de cableado y fotolitos para posibilitar la construcción del circuito mediante el método descrito.
- Se ofrecerán circuitos que cumplan diferentes funciones y tengan diferentes grados de complejidad para que el usuario pueda decidir qué funciones necesita y cuán complejas pueden ser, adaptándose a sus conocimientos y necesidades o gustos.
- Todos los circuitos electrónicos incluirán el código fuente de al menos un pequeño programa de ejemplo que sirva para probar el correcto funcionamiento del circuito.
- Todos los circuitos deberán poder programarse o probarse utilizando las herramientas informáticas ofrecidas con la propia plataforma.

Es necesario aclarar que la plataforma podría ampliarse infinitamente, por lo que este proyecto pretende cubrir las características básicas necesarias para poder construir una suficiente variedad de aplicaciones electrónicas relativamente simples.

2.2. Contexto

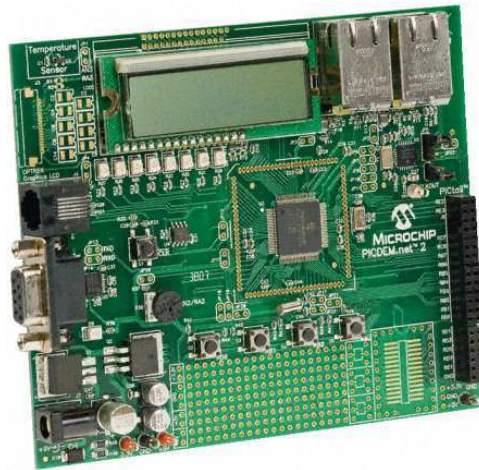


Figura 2.1

Actualmente, la inmensa mayoría de plataformas disponibles para aprender a desarrollar aplicaciones utilizando microcontroladores, como por ejemplo placas entrenadoras (ver Figura 2.1), no pueden utilizarse correctamente en aplicaciones electrónicas reales una vez que se conoce su funcionamiento, siendo muchas de ellas totalmente monolíticas y prefabricadas, dificultando el aprendizaje de su funcionamiento a nivel electrónico y la posibilidad de adaptarlo a las posibles aplicaciones que le quiera dar el usuario final.

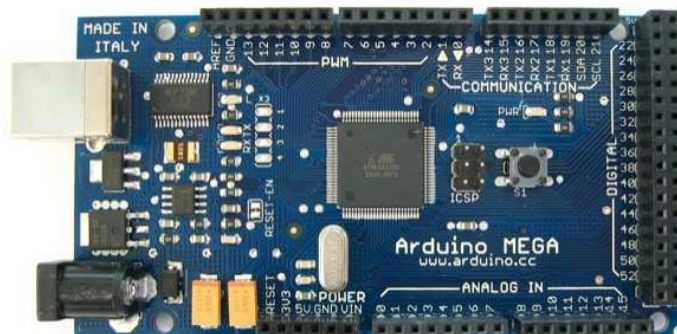


Figura 2.2

Existen algunas plataformas ligeramente modulares que se adaptan en mayor medida a los objetivos de este proyecto, siendo Arduino la más famosa de ellas (Figura 2.2). Los desarrolladores de Arduino la definen como una “plataforma de prototipado basada en hardware y software fácil de usar y flexible, orientada a artistas, diseñadores, aficionados y cualquiera interesado en crear entornos u objetos interactivos”.

No obstante para el desarrollo de este proyecto se han tomado las siguientes decisiones de diseño que difieren de las de plataformas existentes como Arduino, debido a que no se adaptan al objetivo del proyecto o de manera que se aumente la flexibilidad, simplicidad, eficiencia de la plataforma, y se disminuya el coste inicial de desarrollo para los clientes:

- Las placas que contienen microcontroladores **no** incluyen la circuitería necesaria para programar el microcontrolador. Esto hace que se disminuya en gran medida su precio, tamaño, complejidad, y un solo programador pueda utilizarse para todos los microcontroladores.

- Se ha evitado el uso de *bootloaders* para no consumir recursos innecesarios en los microcontroladores ya que no ofrecen ninguna característica importante que no esté disponible dentro de la circuitería del propio microcontrolador.
- Se utilizan los recursos propios de la circuitería del microcontrolador en lugar de implementaciones por software de las mismas. Por ejemplo, los microcontroladores se programan mediante ICSP en lugar de utilizando la funcionalidad de un *bootloader*. El uso de *bootloaders* requeriría un programador ICSP para grabar el propio *bootloader*, y a partir de entonces se utilizaría una circuitería distinta para cambiar el programa del microcontrolador. Utilizando la circuitería interna e ICSP se hace posible utilizar el mismo programador para todos los microcontroladores en cualquier circunstancia.
- Todos los módulos deben poder utilizarse en casi todas las Placas Principales (ver sección 3.1), y su método de interconexión debe evitar en la medida de lo posible que los pines de datos del microcontrolador adyacentes a cada Módulo queden bloqueados y no puedan usarse por otros Módulos.
- Los Módulos (sección 3.1) que ofrecen funciones adicionales al circuito forman parte de la plataforma, y no se necesita acudir a desarrolladores externos para obtenerlas.

2.3. Alcance

El proyecto *Plataforma modular educativa para aprendizaje de microcontroladores* dará como resultado el producto Curuxa que cumplirá con todos los objetivos y especificaciones indicados en el apartado anterior.

El producto Curuxa se distingue en cuatro partes bien diferenciadas: las Placas Principales, los Módulos, las herramientas informáticas y las aplicaciones de ejemplo.

La principal meta del proyecto es que sea un punto de partida robusto que permita la construcción de múltiples circuitos completos de cualquier tipo seleccionando los módulos necesarios para ello. Además, cuando el usuario tenga conocimientos suficientes, podría utilizar todo el material para prototipar cualquier circuito y ampliarlo con módulos diseñados por él mismo.

2.4. Visión general

En cuanto a la estructura de esta Memoria del Proyecto de Fin de Carrera, tras este capítulo donde se presentan los objetivos y la visión general del proyecto, se describe el proyecto en su conjunto. Cómo está organizado, en qué partes está dividido, cuál es su propósito y qué se ha utilizado para su desarrollo.

En el capítulo siguiente se describen los estándares utilizados y todos los circuitos electrónicos que forman parte del proyecto, es decir, cada una de las Placas Principales y de los Módulos.

Tras los circuitos, se describen también todas las herramientas informáticas disponibles para los usuarios, desarrolladas como parte de esta plataforma, seguidas por las aplicaciones de ejemplo.

Finalmente, se comenta dónde se ha utilizado la plataforma en la actualidad, se muestra el diagrama de Gantt con el calendario del desarrollo del proyecto, y se lista la bibliografía empleada.

3. Descripción del proyecto

3.1. Organización

Una de las características principales de esta plataforma es que, desde el principio, fue concebida como un sistema totalmente modular, de manera que cada circuito electrónico fuese fácil de construir por los usuarios y ofreciese una sola funcionalidad específica de manera que se pudiesen combinar los módulos necesarios de manera que en su conjunto cubran todas las funcionalidad requeridas en el producto final que el usuario pretende construir.

Por lo tanto, la plataforma consta de dos tipos de circuitos electrónicos: las **Placas Principales** y los **Módulos**. Las primeras están formadas, a grandes rasgos, por un microcontrolador y un conjunto de conectores a los cuales se conectan todos los demás Módulos según sea necesario. Ambos tipos de circuitos se describen en profundidad en las secciones 4.2 y 4.3.

Para simplificar la programación de ciertas funcionalidades de los microcontroladores y comunicarse con algunos módulos, existe un pequeño conjunto de librerías disponible para los usuarios. Además, para desarrollar programas, grabarlos en microcontroladores y probarlos, puede utilizarse el Entorno de Desarrollo incluido en el mismo proyecto, o los *scripts* para la línea de comandos. Todas estas herramientas informáticas están descritas en el capítulo 5.

Toda la plataforma está orientada a ser ampliada tanto como sea necesario en el futuro, debido a que al ser totalmente modular pueden diseñarse tantos Módulos y Placas Principales como se desee, haciéndolos perfectamente compatibles con los circuitos actuales gracias al uso de estándares internos (ver sección 4.1). La nomenclatura, explicada en la siguiente sección, también está orientada en gran medida a que se puedan añadir más Placas Principales y Módulos que cumplan funciones nuevas, pero puedan continuar utilizando la nomenclatura actual. Las nuevas Placas Principales podrían, sin ningún problema, utilizar otros microcontroladores e incluso otras arquitecturas totalmente distintas de MCUs, y aún así continuarían cumpliendo la nomenclatura definida.

El proyecto pretende que el usuario se involucre en el desarrollo y adaptación de todos los circuitos electrónicos y programas para microcontrolador y ordenador tanto como sea posible. En las secciones 4.4 y 6.5 se describen algunos de los Módulos y Aplicaciones que en la actualidad diversos usuarios han diseñado y construido de manera independiente a este proyecto, pero íntimamente relacionados con él.

3.2. Nomenclatura

Todas las Placas Principales y Módulos se han nombrado en base a la nomenclatura descrita a continuación. Toda la nomenclatura está basada en palabras inglesas para facilitar su comprensión en mercados internacionales, y está diseñada de manera que sea fácil de recordar la función que cumple cada uno de los circuitos solamente viendo su nombre.

Las Placas Principales siempre comienzan con *MB*, que significa *Main Board* (Placa Principal). Actualmente todas ellas están basadas en la arquitectura PIC de instrucciones de 14 bits de Microchip (normalmente PIC12x y PIC16x), por lo que todas las Placas Principales en la actualidad comienzan con el nombre *MBP*. En el futuro, si se utilizasen otras arquitecturas de MCUs (PIC de 32 bits, dsPIC, AVR...) debería cambiarse esta última letra. La numeración final del nombre de las Placas Principales representa el número de patillas que posee el microcontrolador utilizado en cada una de ellas.

Los Módulos, por su parte, se basan en una nomenclatura que los divide en grupos según su utilidad o aplicación. Los nombres de los tipos y subtipos de Módulos se detallan a continuación, siendo estos nombres los prefijos del nombre completo del Módulo, y añadiendo un sufijo a cada uno de ellos para diferenciar cada uno de los Módulos disponibles en cada tipo. Por ejemplo, el Módulo LTIND-A (sección 4.3.1) pertenece al tipo *LT* (control de luces), subtipo *IND* (luces indicadoras) y contiene el sufijo *A* que lo distingue de otros Módulos que pudiesen existir en su misma categoría.

- *LT: Light control*. Control de luces.
 - *LT7S: Displays* de siete segmentos.
 - *LTIL: Luces* de iluminación.
 - *LTIND: Luces* indicadoras.
- *MC: Motor control*. Control de motores.
 - *MC2: Control* bidireccional de motores.
- *SI: Sensor input*. Sensores (sólo con entrada de datos en el microcontrolador).
 - *SISW: Switches*. Botones, pulsadores.
 - *SIBW: Black/white*. Sensores de blanco/negro.
 - *SIDST: Sensores* de distancia.
- *CM: Comunicaciones*.
 - *CMIR: Infra-red*. Comunicación mediante infrarrojos.
 - *CMSP: Serial Port*. Comunicación a través del puerto serie.
- *PWR: Power supplies*. Fuentes de alimentación. Reguladores de tensión.
- *AO: Audio output*. Salida de audio.

Algunas de estas categorías sólo poseen un Módulo que pertenezca a ellas, debido a que como ya se ha comentado anteriormente, todo el proyecto (incluida la nomenclatura) está previsto que pueda ser ampliado en el futuro.

3.3. Funciones del producto

La plataforma está principalmente orientada a la educación, de manera que todos los circuitos electrónicos están pensados para ser contruidos por los propios usuarios, y los microcontroladores ser programados también por ellos.

Es una solución ideal para ser utilizada en institutos, universidades, centros de formación o asociaciones que organicen clases, talleres o cualquier actividad enfocada al aprendizaje sobre electrónica o incluso robótica. Idealmente, los organizadores de la actividad pueden ofrecer todos los componentes electrónicos necesarios para construir una Placa Principal y ciertos Módulos seleccionados, y los alumnos o participantes dedicarían su tiempo en aprender cómo funcionan, construir, programar y probar estos circuitos utilizando la documentación ofrecida. En otros tipos de actividades que por ejemplo pudiesen estar más orientadas a la programación que a la electrónica, se podrían ofrecer a los participantes los circuitos ya completamente contruidos o bien la placa de circuito impreso lista para colocarle los componentes, haciendo disminuir la carga de trabajo de montaje electrónico de los participantes, limitando el tiempo necesario y pudiendo centrarse más en la programación y aplicación práctica de los circuitos, en caso de que ese sea el objetivo de la actividad.

Los componentes y documentación del producto también podrían ofrecerse a usuarios o clientes individuales que no formen parte de ninguna actividad específicamente organizada, pero deseen emplear su propio tiempo libre en aprender a construir circuitos electrónicos y aparatos completos adaptados a sus gustos, pero sin necesidad de diseñar desde cero toda la circuitería.

Por otro lado, usuarios individuales que ya posean conocimientos suficientes y no necesiten aprender, pueden estar interesados también en la plataforma ya que es una buena solución para poder hacer prototipos de aparatos completos de manera rápida y sencilla, sin necesidad de diseñar la circuitería completa para cada una de las aplicaciones que quieran dársele. Por ejemplo, un aficionado a la electrónica podría estar interesado en construir tantas Placas Principales y Módulos como sea posible, para que en el momento de construir un robot de competición que este aficionado esté interesado en implementar, ya tenga un conjunto de circuitos disponibles que cumplan multitud de funciones distintas y pueda seleccionar una Placa Principal y varios Módulos según necesite para construir su robot completo rápidamente y sin diseñar un circuito electrónico desde cero sólo para un robot.

3.4. Lenguajes de programación y compiladores

3.4.1 Para microcontroladores

Las librerías ofrecidas para programar microcontroladores (ver sección 5.2), así como los ejemplos ofrecidos para probar el correcto funcionamiento de los circuitos (ver anexos de este documento), y también los programas completos para microcontrolador de las Aplicaciones de Ejemplo (ver capítulo 6), han sido desarrolladas en lenguaje C debido a que es una solución conocida y fácil de obtener documentación, ofrece gran control a bajo nivel del propio microcontrolador, y el código puede desarrollarse de manera relativamente intuitiva y simple, sin necesidad de complicar al usuario con características innecesarias en ciertas aplicaciones.

Para todo el código fuente escrito en lenguaje C se utiliza el compilador SDCC. Es el único compilador para la arquitectura de microcontroladores utilizada en las Placas Principales que es libre, multiplataforma y gratuito.

Siendo un compilador gratuito, el desembolso que deben realizar los usuarios para utilizar esta plataforma de aprendizaje es mínimo. Siendo multiplataforma, los usuarios pueden trabajar con los circuitos de este proyecto casi sin importar el sistema operativo que utilicen en sus ordenadores personales, o los organizadores de la actividad en que se utilice esta plataforma pueden elegir el sistema operativo empleado, sin estar limitados a Microsoft Windows, como es habitual.

Siendo un compilador libre, los usuarios pueden compartir sin problemas el propio compilador con otras personas, lo cual mejoraría la difusión de la propia plataforma. Además, los usuarios avanzados podrían conocer el funcionamiento interno del propio compilador e incluso modificarlo y ampliarlo.

Este compilador carece de librerías propias habitualmente disponibles en compiladores comerciales, por lo que el usuario se ve obligado a manejar la memoria del microcontrolador a bajo nivel para configurar el circuito como desee. Para facilitar la tarea, se han desarrollado una serie de librerías que simplifican las tareas más habituales y repetitivas. Ver sección 5.2 para más información.

3.4.2 Para ordenadores

Se han desarrollado dos programas con interfaz gráfica para ordenador: el entorno de desarrollo (sección 5.1) y la interfaz de control de una de las Aplicaciones de Ejemplo (sección 6.4).

Ambos programas se han desarrollado en base a la plataforma .NET/Mono utilizando el lenguaje C# 3.0. Es un lenguaje estándar, con muchísimas librerías disponibles, muy bien documentado, y para el que existen implementaciones en una gran cantidad de sistemas operativos, permitiendo que los programas (especialmente interesante para el Entorno de Desarrollo, sección 5.1) puedan desarrollarse, modificarse y ser ejecutados por los usuarios independientemente del sistema operativo que utilicen en sus propios ordenadores.

3.4.3 Scripts para consola

Se incluyen una serie de *scripts* que ofrecen casi la misma funcionalidad que el Entorno de Desarrollo pero en este caso desde la línea de comandos. Estos *scripts* funcionan en cualquier consola Unix o derivado, como Linux o BSD, en la que esté disponible Bash o una *shell* equivalente.

Esto significa que, por norma general, podrán programarse todas las Placas Principales desde la línea de comandos de casi cualquier sistema operativo excepto Windows (a no ser que se haya instalado Bash, lo cual es posible pero muy poco habitual). Los usuarios de Windows, al contrario que los usuarios de muchos otros sistemas operativos, no suelen utilizar la línea de comandos, por lo que siguen teniendo disponible toda la funcionalidad desde el Entorno de Desarrollo con interfaz gráfica.

3.5. Características de los usuarios

Esta plataforma, estando orientada mayoritariamente al aprendizaje de electrónica y microcontroladores, pretende ser utilizada en gran medida por estudiantes universitarios o de instituto. No obstante, como puede utilizarse la plataforma en actividades no organizadas por centros formativos, no sería extraño que una buena parte de los usuarios no fuesen estudiantes, sino aficionados autodidactas.

Por otro lado, como la plataforma también está orientada a aficionados que quieran utilizarlo para el prototipado en lugar del aprendizaje, también cabría esperar que una buena parte de usuarios de la plataforma fuesen aficionados con conocimientos previos en electrónica y microcontroladores.

4. Circuitos electrónicos

El proyecto consta de dos grupos diferenciados de circuitos electrónicos: las Placas Principales y los Módulos. Todos ellos están diseñados para que sean tan fáciles de construir por los propios usuarios/clientes como sea posible, por lo que cada circuito cumple una función muy específica, la plataforma es muy modular y cada circuito ha sido simplificado lo máximo posible sin por ello renunciar a funcionalidad, calidad y fiabilidad.

Debido a esto, algunos circuitos son tan simples que no es necesario construirlos sobre una placa. Por ejemplo, LTIND-A, LTIL-A y SIBW-1Y son ejemplos de circuitos contruidos “**punto a punto**”, es decir, circuitos muy simples que no utilizan placas de circuito impreso ni similares, sino que todos los componentes se sueldan entre sí mediante sus propias patillas o mediante cables.

Los circuitos ligeramente más complejos se construyen utilizando **placas perforadas** como la de la figura 4.1. Éstas son placas de circuito impreso prefabricadas, con una rejilla de agujeros separados 100mils entre sí, y con una bahía de cobre alrededor de cada agujero. En este proyecto, las pistas de la cara inferior se construyen uniendo las distintas bahías mediante estaño, y las de la cara superior mediante pequeños cables. Estas placas son muy baratas, muy fáciles de conseguir y muy fáciles de construir, por lo que son la mejor opción para que cada usuario pueda construir Módulos relativamente simples sin necesidad de fabricar las mucho más complejas placas de circuito impreso.

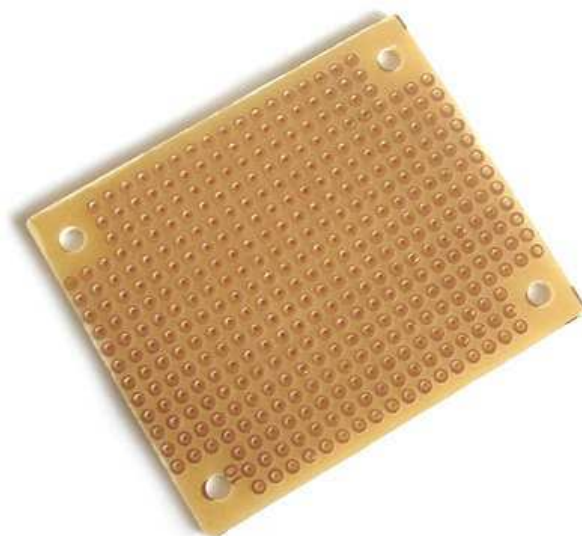


Figura 4.1

No es nada práctico construir los circuitos más complejos y con mayor cantidad de componentes sobre placas perforadas, por lo que en estos casos se diseñan **placas de circuito impreso** (PCB). Éstas resultan en un circuito final mucho más pequeño incluso con mucha más complejidad y número de componentes, pero los usuarios habitualmente necesitarían aprender a fabricar PCBs desde cero, lo cual es muy laborioso si no se dispone de conocimientos previos y un local adecuado.

Por lo tanto, las Placas Principales y Módulos se han construido mediante una de las tres técnicas descritas (punto a punto, placa perforada o PCB) según su complejidad. De esta manera, los usuarios no sólo pueden elegir qué Módulos construir en base a las

funcionalidades necesarias, sino que también pueden decidir construir uno u otro Módulo según su complejidad, pudiendo además ir construyendo Módulos más complejos a medida que el propio usuario adquiere experiencia en cada una de las técnicas.

Los circuitos, al ser modulares, se han diseñado para ser totalmente interconectables. Para conectar los Módulos a las Placas Principales se utilizarán habitualmente cables y pines macho y hembra de 100mils de separación. Si bien el uso de cables y conectores dificulta ligeramente la construcción de cada Módulo, es la mejor solución para que los Módulos no bloqueen la posible conexión de otro Módulo y así no se desperdicien pines de datos del microcontrolador, pudiendo así utilizar en el mismo circuito tantos Módulos como pines de datos haya disponibles en la Placa Principal.

4.1. Estándares

Todas las Placas Principales y Módulos han sido diseñados utilizando los estándares internos descritos en esta sección, de manera que todos los Módulos puedan ser conectados a cualquier Placa Principal, todas las placas se puedan alimentar de la misma manera, y se puedan programar utilizando el mismo tipo de programador.

4.1.1 Conector de datos

El conector de datos es una configuración estándar para interconectar Placas Principales y Módulos. El conector de datos simple, que es el más habitual, está formado por tres pines:

Pin N°1 - Siempre es la patilla más cercana al microcontrolador. Éste es el pin de datos, es decir, todas las señales analógicas o digitales viajan a través de él al comunicarse las Placas Principales y Módulos

Pin N°2 - La patilla central. Se corresponde con V_{cc} , es decir, la tensión de alimentación del circuito.

Pin N°3 - La patilla más alejada del microcontrolador. Se corresponde con GND, es decir, la masa de todo el circuito.

Todas las Placas Principales poseen un conjunto de pines macho separados 100 mils unos de otros, colocados de manera que rodeen al microcontrolador. Todos los pines de entrada o salida digital o analógica del microcontrolador tiene su correspondiente conjunto de 3 pines macho asociados, a los cuales se conectará el Módulo que pretenda comunicarse con el microcontrolador a través de dicho pin.

Todos los Módulos que necesitan comunicarse con el microcontrolador de la Placa Principal (es decir, todos excepto los reguladores de tensión) poseen al menos un conector de datos como el aquí descrito, formado por tres pines hembra.

Por lo general no se especifica ningún código de colores para los cables utilizados en el conector de alimentación. No obstante, el pin N°1 deberá ser indicado mediante una marca.

La figura 4.2 representa el circuito construido en placa perforada de una de las Placas Principales, en la que se resalta uno de los conectores de datos. La figura 4.3 es una fotografía de la misma Placa Principal una vez construida. La figura 4.4 es una fotografía del conector de datos de uno de los Módulos, y la figura 4.5 se corresponde con la representación gráfica

del conector de datos hembra utilizado en los diagramas de cableado de algunos Módulos (ver Documento Planos).

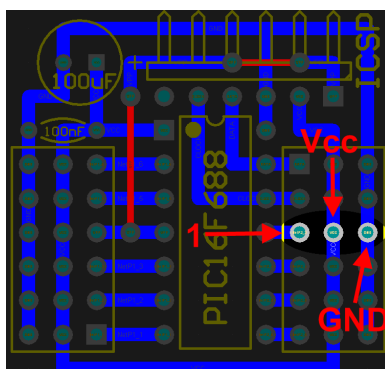


Figura 4.2

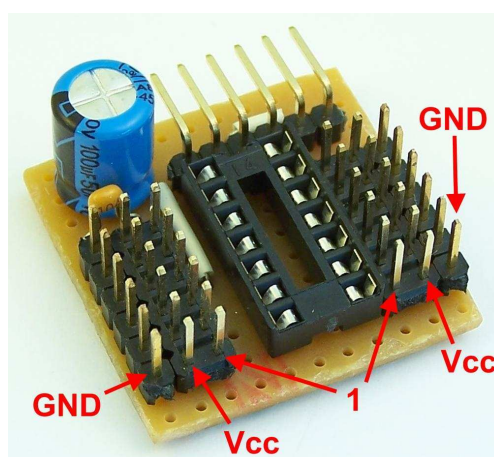


Figura 4.3

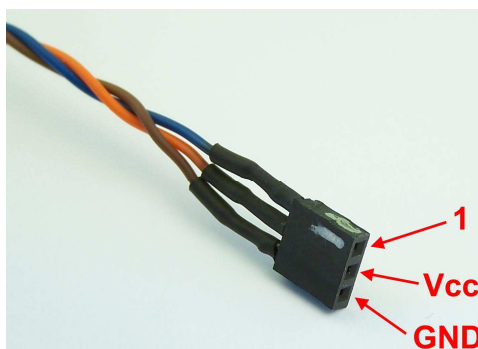


Figura 4.4



Figura 4.5

La mayoría de Módulos sólo necesitan un pin de datos analógico o digital para comunicarse con el microcontrolador de la Placa Principal. No obstante, algunos Módulos

más complejos, como por ejemplo MC2A (ver sección 4.3.9), necesitan más de un pin de datos para comunicarse. Estos Módulos, como norma general, podrán basarse en conectores de más de 3 pines, pero siempre siguiendo el orden y tipo de conexonado aquí explicado de manera que puedan comunicarse adecuadamente con las Placas Principales.

4.1.2 Conector a alimentación

Las pilas, baterías o fuentes de alimentación, se conectan a la Placa Principal del circuito que se esté construyendo mediante el **conector de alimentación**. Este conector está formado por dos pines que se corresponden con *Vcc* y *GND*, es decir, tensión de alimentación y masa.

El conector de alimentación puede conectarse a los pines 2 y 3 de cualquiera de los conectores de datos de la Placa Principal, siempre que se respete la polaridad y ese conector de datos no esté siendo utilizado. En la mayoría de aplicaciones construidas utilizando Placas Principales de muchos pines (como MBP18 o MBP40) lo más probable es que algunos de los conectores de datos no sean utilizados, quedando a disposición del conector de alimentación. No obstante, algunas Placas Principales, especialmente las más pequeñas como MBP8, incluyen parejas de pines macho junto con los conectores de datos, pudiendo ser utilizados por el conector de alimentación.

En todos los circuitos *GND* es la masa y, por tanto, la tensión de referencia de todo el circuito, incluyendo Placas Principales, Módulos y posibles componentes externos. *Vcc* siempre se corresponde con la tensión de alimentación y, salvo que se indique lo contrario, esta tensión será 5V ($\pm 0.5V$) respecto a masa.

Una vez conectada la fuente de alimentación a la Placa Principal mediante el conector de alimentación, ésta proveerá la tensión necesaria a la Placa Principal y todos los Módulos conectados a ella. Por lo general, ningún módulo consumirá más de 150mA, y el circuito en su conjunto no debería sobrepasar un amperio. Los Módulos que requieran alimentación a mayor tensión o corriente estarán alimentados, al menos parcialmente, desde fuentes de alimentación independientes (por ejemplo MC2A, sección 4.3.9).

Se recomienda utilizar un cable rojo para *Vcc* y uno negro para *GND*, pero no es obligatorio. No obstante, si no se utilizan cables rojo y negro se hace totalmente necesario una marca indicadora del pin *Vcc*.

La figura 4.6 es una fotografía de un conector de alimentación proveniente de un regulador de tensión, que puede conectarse a los pines 2 y 3 de cualquier conector de datos en cualquier Placa Principal. La figura 4.7 muestra un zócalo para cuatro pilas AA con su correspondiente conector de alimentación. La figura 4.8 muestra una Placa Principal siendo alimentada a través de un conector de alimentación a través de un conector de datos que no está siendo utilizado.

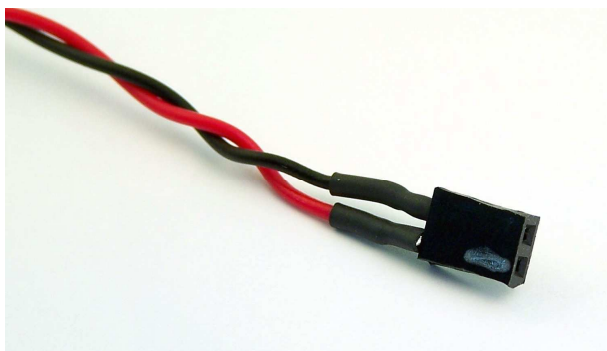


Figura 4.6

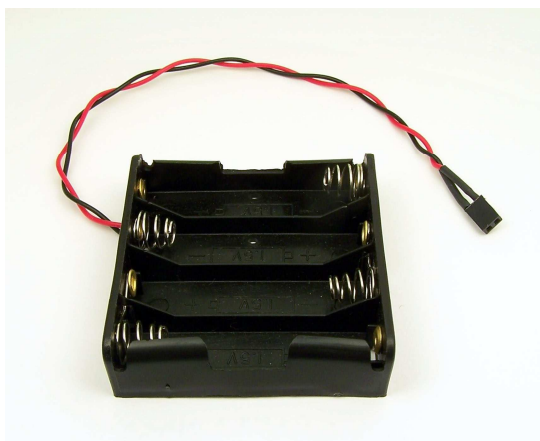


Figura 4.7

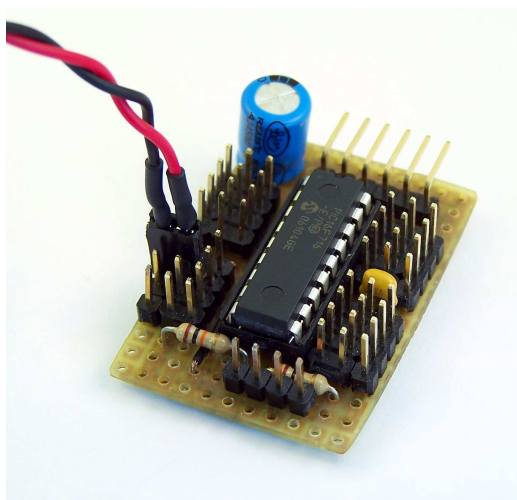


Figura 4.8

4.1.3 Conector de programación

El conector de programación es un conector de seis pines a través del cual se transmiten las señales de datos de ICSP. Este protocolo está especificado por Microchip, el fabricante de los microcontroladores utilizados en las Placas Principales, y es implementado por el dispositivo programador y por el propio microcontrolador, por lo que la especificación de ICSP no recae en este proyecto ni es necesaria su implementación en ningún momento, pero sí es necesario proveer a las Placas Principales de un tipo de conector a través del cual puedan transmitirse las señales utilizadas para programar el microcontrolador.

Consta de los siguientes pines:

- Pin N°1 - *Vpp*. Tensión de programación y reinicio del microcontrolador. La tensión de programación es mayor que *Vcc*, la de alimentación, y se utiliza para hacer que el microcontrolador entre en modo de programación.
- Pin N°2 - *Vcc*. Tensión de alimentación. A través de este pin se puede alimentar la Placa Principal y, por tanto, todo el circuito desde el propio programador.
- Pin N°3 - *GND*. Masa de todo el circuito.
- Pin N°4 - *Data*. Transmisión de datos digitales para programar el microcontrolador.
- Pin N°5 - *Clock*. Transmisión de la señal de reloj para la comunicación síncrona utilizada por el pin de datos.
- Pin N°6 - *NC*. No conectado.

Este conector de alimentación es el mismo que emplean los programadores oficiales de Microchip PICkit 1, 2 y 3, por lo que estos programadores pueden conectarse directamente a cualquier Placa Principal sin necesidad de adaptadores.

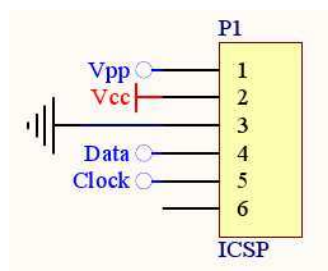


Figura 4.7

La figura 4.9 muestra la representación gráfica del conector de programación utilizada en los esquemas electrónicos de las Placas Principales (ver Documento Planos). Las figuras 4.10 y 4.11 muestran la distribución de pines del conector de programación en una Placa Principal y en el programador Microchip PICkit 2. La figura 4.12 muestra MBP40 conectada al PICkit 2 mediante el conector de programación.

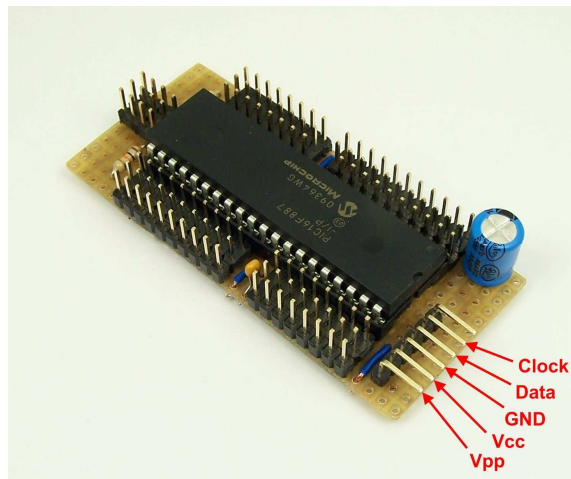


Figura 4.8

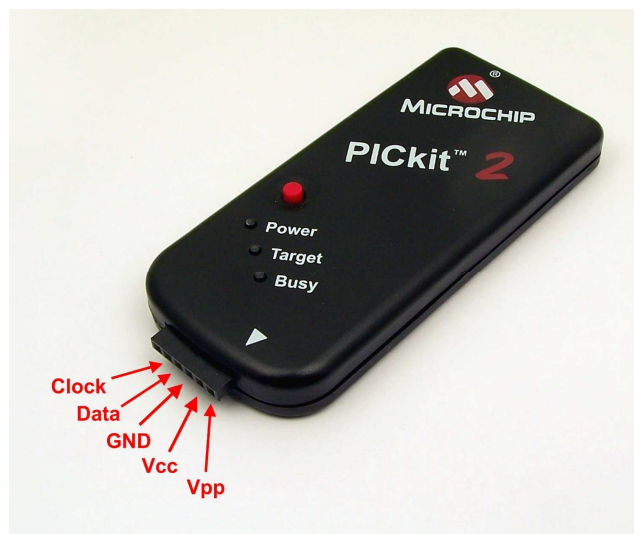


Figura 4.9

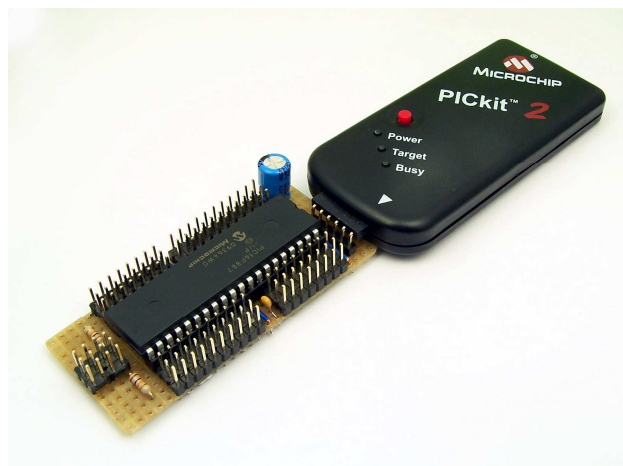


Figura 4.10

4.2. Placas Principales

Las Placas Principales son circuitos electrónicos contruidos utilizando placas perforadas con bahías individuales o en placas de circuito impreso, las cuales contienen al menos un microcontrolador PIC de Microchip, un conector de programación (ver sección 4.1.3) y un conjunto de pines macho que se corresponden con un conector de datos de 3 pines por cada pin analógico o digital del microcontrolador.

Las Placas Principales están pensadas para ser *el cerebro*, el centro de recepción de todos los sensores (SISW-SPST, SIBW-1Y...), lugar de procesado de toda la información, y quien envía las señales a los Módulos que actúan sobre componentes externos, como por ejemplo MC2B (sección 4.3.10).

Con cada una de las Placas Principales se incluye su esquema electrónico y su diagrama para placa perforada (ver Documento Planos), al menos un pequeño programa para microcontrolador útil para comprobar la correcta construcción y funcionamiento de la placa (ver anexos de este documento), una serie de fotografías de su apariencia una vez construido, y la versión imprimible del circuito, es decir, un archivo listo para ser impreso y poder utilizarse por el usuario como referencia durante la construcción del circuito, sin necesidad de utilizar un ordenador (ver CD adjunto).

Los microcontroladores se han seleccionado específicamente para cada Placa Principal de manera que incluyan la mayor cantidad posible de periféricos integrados, especialmente I²C y UART, que pueden utilizarse para comunicar unas Placas Principales con otras, y éstas con ciertos Módulos o componentes.

Todas las Placas Principales incluyen al menos un condensador de desacoplo, habitualmente dos: uno cerámico y uno electrolítico. Estos podrían omitirse durante su construcción como medida de simplificación del circuito. No obstante hacerlo no es recomendable ya que los condensadores estabilizan la tensión de alimentación del circuito y evitan funcionamientos erróneos.

A continuación, se describe brevemente cada una de las Placas Principales desarrolladas junto con una pequeña de las características más importantes, normalmente utilizada para seleccionar la Placa Principal a utilizar en cada aplicación.

4.2.1 MBP8

Ésta es la Placa Principal más simple y pequeña de la plataforma. Está orientada a aplicaciones muy simples que utilicen muy pocos Módulos y requieran poca memoria para su programación. Además, es una buena opción como iniciación al desarrollo de circuitos con microcontroladores por parte de los usuarios. Posee las siguientes características:

- Basada en un PIC12F683
 - o Alimentación: 2.0 to 5.5V
 - o Memoria de programa: 3584 bytes (2048 instrucciones)
 - o RAM: 128 bytes
 - o EEPROM: 256 bytes

- o Oscilador interno de 31kHz-8MHz
- o Oscilador externo de hasta 20MHz
- o 2 temporizadores de 8 bits y uno de 16 bits
- o Conversor analógico-digital de 10 bits y 4 canales
- o CCP/PWM
- o Comparador analógico
- Conector ICSP (de programación)
- Hasta 6 conectores de datos
- Un condensador de desacoplo

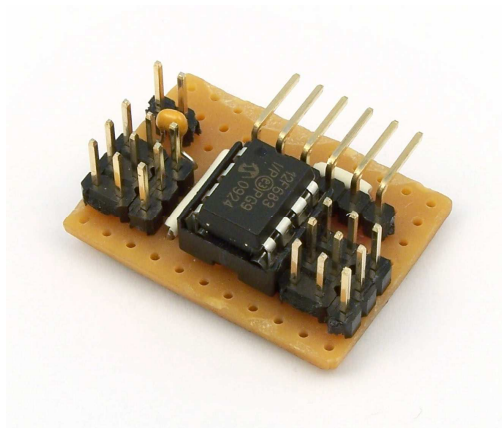


Figura 4.11

Ésta es la única Placa Principal que no soporta UART, necesario para el Módulo CMSP-MAX (sección 4.3.13) para comunicarse mediante puerto serie con un ordenador.

4.2.2 MBP14

Ésta es la Placa Principal más pequeña posible para poder comunicarse con un ordenador utilizando el módulo CMSP-MAX. No obstante, no soporta I2C, el cual podría ser necesario para comunicarse con otros circuitos. Sus características son:

- Basada en un PIC16F688
 - o Alimentación: 2.0 to 5.5V
 - o Memoria de programa: 7168 bytes (4096 instrucciones)
 - o RAM: 256 bytes
 - o EEPROM: 256 bytes
 - o Oscilador interno de 32kHz-8MHz
 - o Oscilador externo de hasta 20MHz

- o Un temporizador de 8 bits y uno de 16 bits
- o Conversor analógico-digital de 10 bits y 8 canales
- o EUSART
- o Doble comparador analógico
- Conector ICSP (de programación)
- Hasta 12 conectores de datos
- Dos condensadores de desacoplo

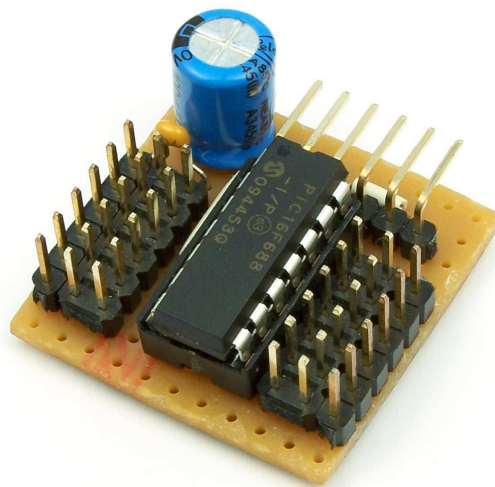


Figura 4.12

4.2.3 MBP18

Ésta es probablemente la Placa Principal más genérica. Su microcontrolador incluye una buena cantidad de periféricos integrados útiles para realizar multitud de tareas en muy diversas aplicaciones, incluyendo UART para comunicarse con ordenadores e I2C para comunicarse con otras placas. Además, posee suficientes conectores de datos para poder utilizar junto con ella una cantidad de Módulos más que aceptable para muchas aplicaciones distintas. Sus características son:

- Basada en un PIC16F88
 - o Alimentación: 2.0 to 5.5V
 - o Memoria de programa: 7168 bytes (4096 instrucciones)
 - o RAM: 360 bytes
 - o EEPROM: 256 bytes
 - o Oscilador interno de 31kHz-8MHz

- o Oscilador externo de hasta 20MHz
- o 2 temporizadores de 8 bits y uno de 16 bits
- o CPP/PWM
- o Conversor analógico-digital de 10 bits y 7 canales
- o SPP (SPI, slave I2C, master I2C must be implemented in software)
- o AUSART/SCI/RS-232
- o Doble comparador analógico
- Conector ICSP (de programación)
- Hasta 16 conectores de datos
- Conector I2C con resistencias de pull-up
- Dos condensadores de desacoplo

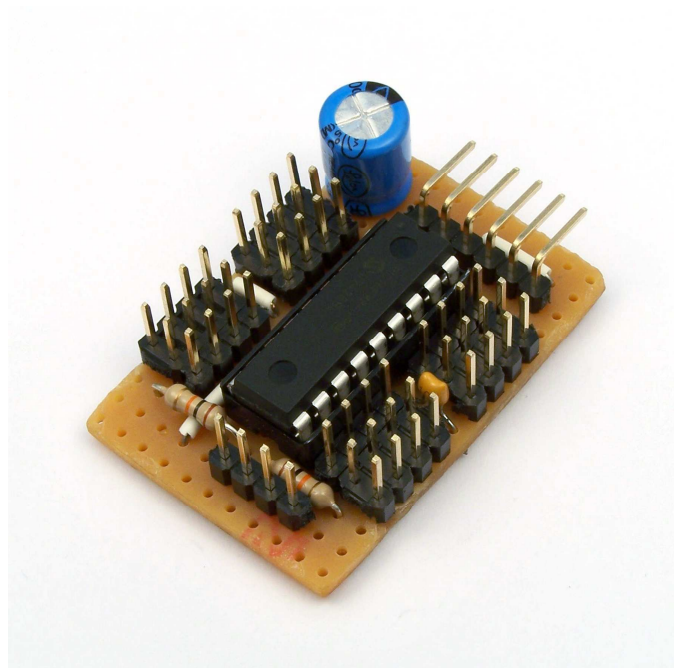


Figura 4.13

4.2.4 MBP40

MBP40 es la mayor de las Placas Principales del proyecto. Es la que incluye mayor cantidad de conectores de datos y su microcontrolador mayor cantidad de periféricos integrados. Por lo tanto, pretende ser la Placa Principal ideal para aplicaciones que requieran una gran cantidad de Módulos o una gran memoria. Sus características son:

- Basada en un PIC16F887
 - o Alimentación: 2.0 to 5.5V

- o Memoria de programa: 14336 bytes (8192 instrucciones)
 - o RAM: 368 bytes
 - o EEPROM: 256 bytes
 - o Oscilador interno de 31kHz-8MHz
 - o Oscilador externo de hasta 20MHz
 - o 2 temporizadores de 8 bits y uno de 16 bits
 - o Dos módulos CCP/PWM
 - o Conversor analógico-digital de 10 bits y 14 canales
 - o MSPP (SPI, I2C maestro/esclavo)
 - o EUSART/SCI/RS-232
 - o Doble comparador analógico
- Conector ICSP (de programación)
 - Hasta 35 conectores de datos
 - Tres conectores I2C con resistencias de pull-up
 - Dos condensadores de desacoplo
 - Un conector de alimentación independiente, y espacio suficiente para otros tres

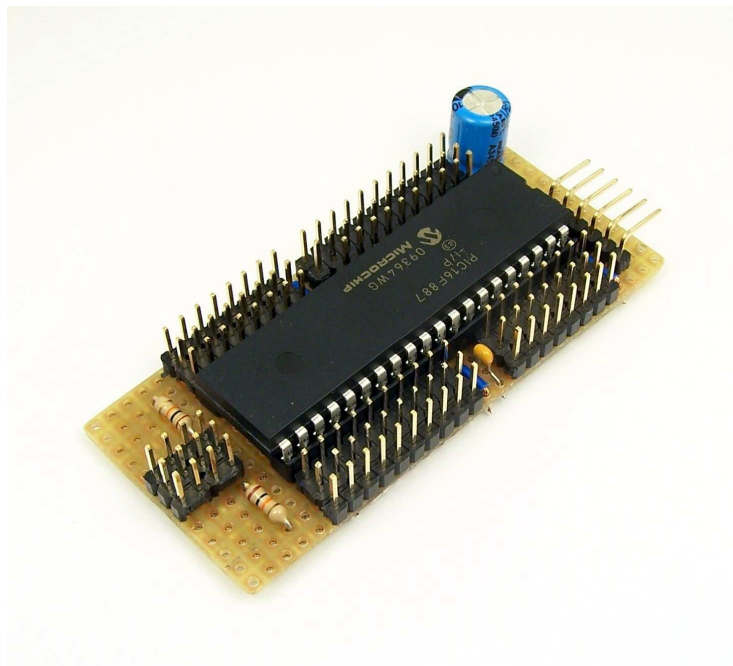


Figura 4.14

4.3. Módulos

Los Módulos son circuitos electrónicos diseñados para ser conectados a las Placas Principales o a otros Módulos. Casi todos los Módulos incluyen al menos un conector de datos para comunicarse mediante señales analógicas o digitales con el microcontrolador de la Placa Principal. Otros Módulos, como los reguladores de tensión, solamente incluyen un conector de alimentación que se utilizará para alimentar a la Placa Principal, a otro Módulo, o más habitualmente a todo el circuito.

El componente principal de algunos Módulos es tan simple como un LED o un pulsador, mientras que otros pueden ser bastante más complejos e incluir multitud de componentes, conectores, y necesidad de construir una placa de circuito impreso sólo para la función que desempeña cierto Módulo. La plataforma completa está muy enfocada a la educación, y ello se puede apreciar en la diferencia en complejidad de unos Módulos a otros, de manera que los usuarios puedan elegir qué Módulos construir y utilizar no sólo por la función que desempeñan, sino por su grado de complejidad en función de las capacidades y conocimientos del propio usuario.

Debido a esta diferencia de complejidad entre Módulos, cada uno ha podido ser construido de una de las tres maneras siguientes (ver introducción del capítulo 4):

- **“Punto a punto”:** en los Módulos más simples los componentes se sueldan entre sí utilizando sus propias patillas o mediante cables.
- **Placa perforada:** los Módulos de complejidad intermedia se construyen utilizando placas perforadas agujereadas en rejilla de 100x100mils. Las pistas inferiores se crean uniendo las bahías con estaño y las pistas superiores con trozos de cable.
- **Placa de circuito impreso:** los Módulos más complejos requieren la construcción de un PCB completo sólo para ese Módulo.

En este proyecto, con cada Módulo se incluye (según sea necesario para cada Módulos) su esquema electrónico, su diagrama para placa perforada, su diseño para PCB, su diagrama de cableado (ver Documento Planos), al menos un pequeño programa para una de las Placas Principales que se puede utilizar para comprobar que cada Módulo funciona correctamente (ver anexos de este documento), fotografías de su construcción y la versión imprimible del circuito (ver CD adjunto).

Todos los componentes utilizados en los Módulos han sido elegidos de manera que sean lo más comunes posibles, baratos y fáciles de conseguir por los usuarios en cualquier tienda de electrónica.

4.3.1 LTIND-A

Éste es el Módulo más simple de la plataforma. Está formado por un simple LED, una resistencia y un conector de datos. Está pensado para ser el primer Módulo construido por los usuarios para que puedan comenzar por lo básico y a partir de entonces tengan la herramienta más simple posible para comprobar que tanto las Placas Principales como muchos otros Módulos que construyan a partir de entonces funcionan correctamente.

Una vez conectado LTIND-A a una Placa Principal, el LED se iluminará cuando el pin del microcontrolador al que está conectado se haya configurado como salida digital y se

encuentre en estado alto (“1”). Cuando el pin se encuentre en estado bajo (“0”) o configurado como entrada digital o analógica, el LED no se iluminará.

Este Módulo solamente soporta LEDs de muy baja potencia. El Módulo LTIL-A (ver sección 4.3.3) soporta LEDs de mucha más potencia y puede utilizarse para iluminación, no sólo como indicador.



Figura 4.15

LTIND-A puede construirse de dos maneras. La primera de ellas, mostrada en la figura 4.15, utiliza dos cables que unen el LED con la resistencia y el conector de datos, pudiendo colocar el LED a cierta distancia de la Placa Principal. La segunda versión de LTIND-A, mostrada en la figura 4.16, evita el uso de cables al soldar todos los componentes entre sí utilizando sus propias patillas, simplificando su construcción en gran medida, y pudiendo usarse como indicador simple del estado de funcionamiento interno del microcontrolador o de otros Módulos.



Figura 4.16

Para comprobar que LTIND-A se ha construido correctamente puede utilizarse el código de ejemplo disponible en el Anexo VIII. Además, los ejemplos de todas las Placas Principales y los ejemplos de muchos otros Módulos requieren el uso de LTIND-A (ver otros anexos de este documento).

4.3.2 SISW-SPST

Este Módulo está formado solamente por un botón, conmutador, pulsador o final de carrera, y pretende ser el método más simple de entrada de datos hacia el microcontrolador.

Habitualmente se utilizará para que el propio usuario del circuito final pueda enviar indicaciones simples al circuito, o pueda detectarse el contacto del aparato con otro objeto.

SIST-SPST envía señales digitales al microcontrolador utilizando lógica invertida. Esto quiere decir que, cuando el pin del microcontrolador esté configurado como entrada de datos digitales, el programa leerá un estado alto (“1”) cuando el pulsador esté en reposo (no pulsado) y un estado bajo (“0”) cuando esté pulsado.

Este Módulo puede utilizarse también como botón de reinicio del microcontrolador sin necesidad de ninguna modificación. Para ello deberá conectarse al pin *MCLR* del MCU y deberá activarse el uso de MCLR en los bits de configuración en el código fuente.



Figura 4.17

4.3.3 LTIL-A

LTIL-A está basado en un LED de unos 50-100mA que puede utilizarse para iluminación en baja potencia de pequeñas áreas u objetos. Este Módulo es una buena opción para iluminar salas de menos de unas pocas decenas de metros cuadrados que estén siendo grabadas con una cámara de vídeo y no se requiera gran iluminación. Por ejemplo, se utiliza en PCBot (ver sección 6.4) para iluminar salas completamente oscuras y que la cámara integrada pueda seguir utilizándose para controlar el robot.

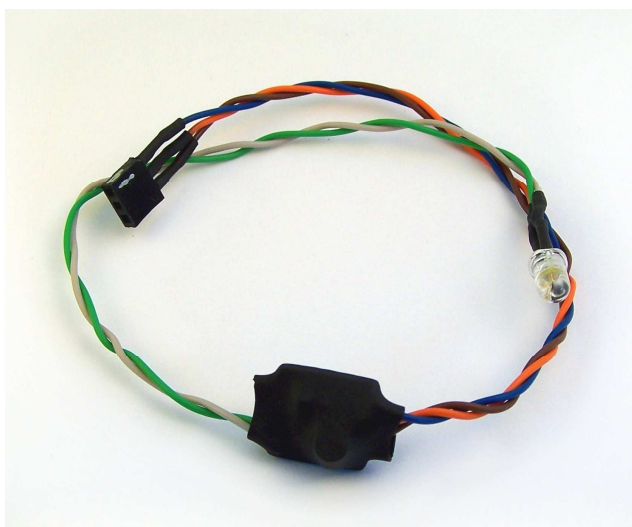


Figura 4.18

Su funcionamiento es similar a LTIND-A, de manera que el LED se iluminará cuando el pin de datos esté configurado a nivel digital alto y no se iluminará en caso contrario. Por lo

tanto, LTIL-A es eléctricamente compatible con LTIND-A, y todos los ejemplos disponibles para LTIND-A (ver anexos de este documento) pueden utilizarse para comprobar que LTIL-A funciona correctamente.

El LED consume mucha más potencia de la que los pines digitales del microcontrolador pueden proveer, por lo que el LED es alimentado a través de un transistor BJT NPN (2N3904 o equivalente) trabajando en corte/saturación y controlado desde un pin digital del MCU. Debido a que la tensión de alimentación habitual del circuito es de 5V pero el LED funciona a 3V, se han colocado dos diodos en serie con el LED para disminuir la tensión a la que es alimentado, en lugar de utilizar una resistencia de 1/4W, la cual no soportaría la potencia que debe disipar.

4.3.4 SIBW-1Y

Sensor de reflectividad que devuelve un valor analógico entre 0V y la tensión de alimentación del circuito en función de la cantidad de luz infrarroja reflejada por el objeto colocado delante del sensor. El rango de funcionamiento es de menos de un centímetro, por lo que si el objeto se coloca más alejado de 1cm del sensor, no se detectará prácticamente ningún reflejo.

SIBW-1Y es una opción muy simple y útil para diferenciar entre objetos blancos y negros, conectando su conector de datos a una entrada digital del microcontrolador de la Placa Principal. No obstante, los tonos grises y objetos blancos que no reflejen suficiente luz infrarroja pueden dar problemas con este sensor.

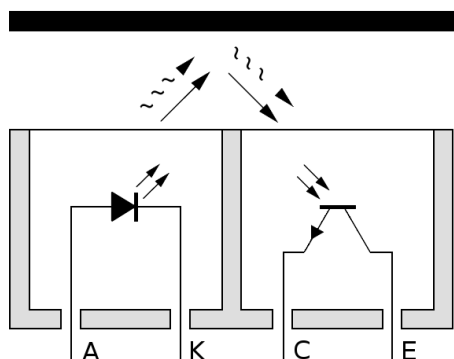


Figura 4.19

El componente principal de SIBW-1Y es el sensor CNY70, el cual integra un LED infrarrojo y un fototransistor (ver figura 4.19). Una vez conectado a la Placa Principal, el LED emite luz constantemente. El fototransistor permitirá circular corriente en función de la cantidad de luz reflejada en un objeto cercano. La resistencia R2 (ver Documento Planos) permite que según la corriente circulante se pueda recibir una tensión entre 0V y Vcc en el microcontrolador.

Cuando se encuentre un objeto reflectivo cercano o un objeto blanco se recibirán alrededor de cero voltios en el microcontrolador. En cambio, cuando no hay ningún objeto cercano, o ese objeto no es reflectivo, el microcontrolador leerá valores cercanos al 1 lógico.

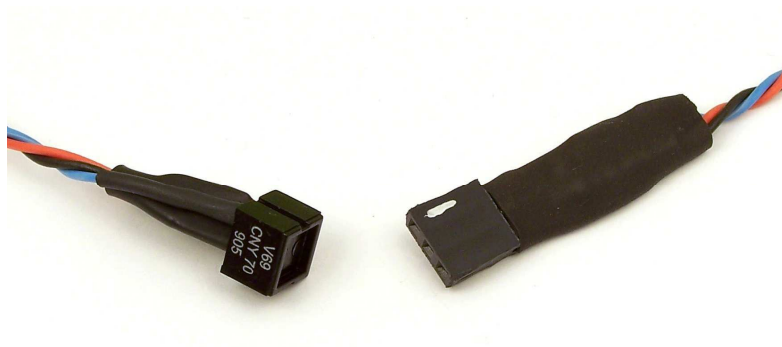


Figura 4.20

4.3.5 AO-SPK

Este Módulo es un simple altavoz de muy baja potencia utilizado habitualmente para emitir sonidos simples mediante ondas cuadradas generadas desde un pin de salida digital del microcontrolador. Su uso más típico sería como emisor de sonidos de alarma o para indicar algún tipo de actividad en el circuito.

La resistencia R1 (ver Documento Planos) es opcional, pero se recomienda su colocación para evitar cortocircuitos. Esta resistencia puede sustituirse por un pequeño condensador cerámico. Los programas de ejemplo que se incluyen (ver Anexos XI, XII y XIII) generan ondas cuadradas de frecuencia constante utilizando distintas Placas Principales y métodos. Otras aplicaciones pueden generar distintos tonos, y posibles circuitos externos podrían utilizarse para generar señales más complejas que produzcan distintos sonidos.

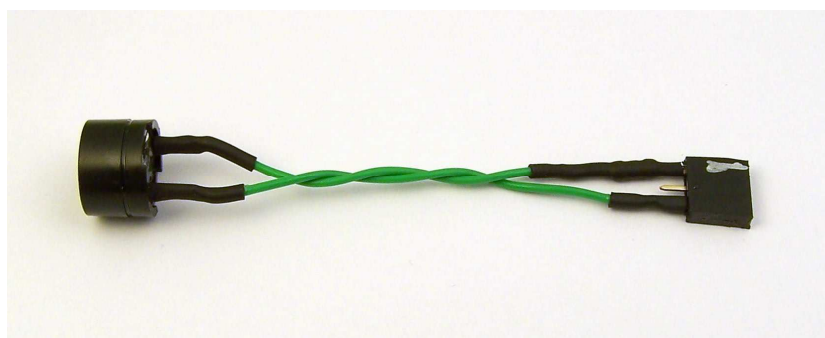


Figura 4.21

4.3.6 SIDST-GP2

Éste es un Módulo sensor de distancia con salida analógica. Puede utilizarse para medir la distancia al objeto más cercano situado enfrente del GP2 (GP2D-12 o GP2Y0A21YK), el componente principal del Módulo. El rango de trabajo de este sensor es de entre 10 y 80 centímetros. No se incluye esquema electrónico con este Módulo, solamente diagrama de cableado, debido a que el circuito solamente está compuesto por el conector de datos y el sensor GP2 (ver Documento Planos); no obstante es aconsejable colocar una resistencia de entre 1000 y 10000 ohmios en el pin de datos para evitar posibles cortocircuitos.

Una vez conectado el Módulo a la Placa Principal, el microcontrolador recibirá una tensión analógica en función de la distancia a la que está situado el objeto más cercano, por lo que debería ser leída utilizando el conversor analógico-digital del propio MCU.

La tensión obtenida está relacionada inversamente con la distancia al objeto, por ejemplo:

- 15cm - 1.65V
- 25cm - 1.1V
- 40cm - 0.75V
- 80cm - 0.4V

(Para más información, consultar la figura 4.22)

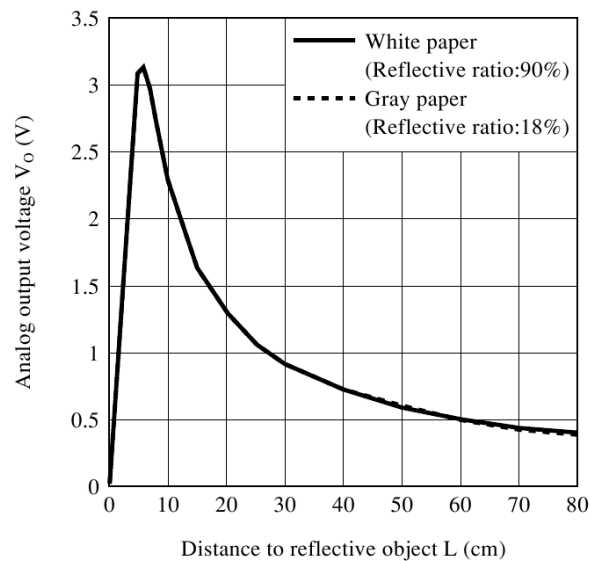


Figura 4.22

Se advierte que, aunque en el diagrama de cableado (ver Documento Planos) se muestran dos conectores hembras de tres pines, el conector que une el cable con el sensor GP2 debe tener un espaciado entre pines de 2mm, en lugar de los 100mils (2.54mm) que se usan en todos los demás conectores hembras del proyecto.

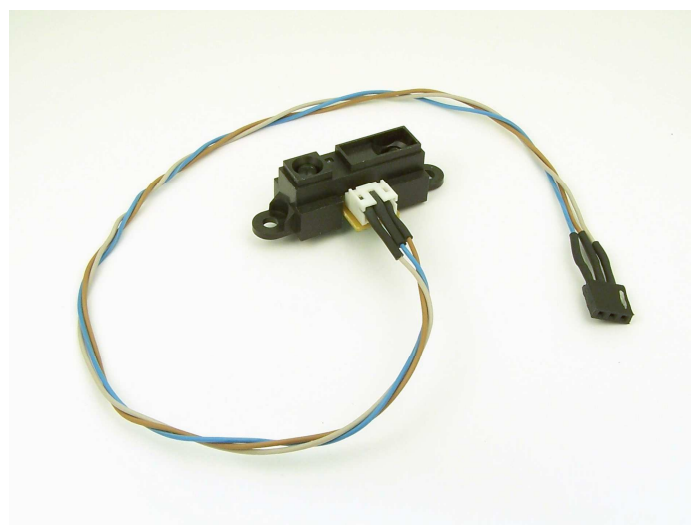


Figura 4.23

4.3.7 LT7S-HN

LT7S-HN (“*Light, 7-segment, High-voltage, N-displays*”) es un Módulo que sirve para controlar *displays* de segmentos a mayor tensión y corriente de las que los pines digitales del microcontrolador pueden soportar, es decir, *displays* de gran tamaño que funcionen a más tensión o consuman más corriente de la habitual.

Este Módulo es ligeramente distinto a los demás: está diseñado para que pueda adaptarse muy fácilmente a cualquier cantidad de *displays* y de segmentos. Esto se consigue cortocircuitando los segmentos similares de todos los *displays*, y controlando el encendido y apagado de cada *display* individualmente. De esta manera, el microcontrolador apaga todos los *displays*, a continuación activa los segmentos necesarios para mostrar el carácter deseado en cierto *display*, se enciende dicho *display* durante un corto periodo de tiempo, y se vuelve a repetir la operación para todos los demás *displays*. De esta manera, utilizando unos pocos pines de salida digitales, se pueden controlar decenas de segmentos repartidos entre múltiples *displays*, haciendo que en cada momento sólo un *display* permanezca encendido, pero encendiendo y apagando todos los *displays* decenas o incluso cientos de veces por segundo, de manera que aparentemente todos los *displays* parezcan encendidos constantemente.

Este *display* se utiliza en la claqueta digital de ejemplo (ver sección 6.3), la cual requiere seis *displays* de siete segmentos cada uno alimentados a 7-9V. En dicho ejemplo, si no se hubiesen multiplexado los *displays* de la manera aquí descrita, se hubiesen necesitado 42 pines de salida digital en el microcontrolador (no disponibles en ninguna Placa Principal), junto con 42 transistores y 42 resistencias. En cambio, utilizando este Módulo, se ha podido construir la claqueta de ejemplo usando solamente 13 pines digitales, 20 transistores y 20 resistencias.

En la parte superior de la primera página del plano N°11 (ver Documento Planos) se muestra un esquema que representa las conexiones necesarias para uno de los segmentos en un circuito que utiliza tres *displays*. El transistor BJT NPN Q2, controlado desde un pin digital del microcontrolador (S1 (d)), activa otro transistor PNP (Q1) que, a su vez, activa y desactiva al mismo tiempo el mismo segmento en todos los *displays* del circuito (DS1-3). Esta porción de circuito es similar para todos los demás segmentos. Por otro lado, se utilizan varios transistores (Q3-5) para encender y apagar cada uno de los *displays* mediante los pines digitales D1 (d), D2 (d) y D3 (d).

En el mismo plano, bajo este esquema, se muestran dos pequeños esquemas electrónicos que representan cada uno de los segmentos y *displays* que se desean utilizar en el circuito y, bajo ellos, el diagrama para placa perforada de la porción de circuito que habría que añadir para cada uno de los segmentos y *displays* del circuito final.

En la segunda página del plano N°11 se muestra el esquema completo y el diagrama para placa perforada completo de un LT7S-HN diseñado para 7 segmentos y 6 *displays*. Como se puede apreciar, todos los segmentos y todos los *displays* comparten un mismo circuito electrónico y distribución de componentes en placa perforada el cual puede adaptarse muy fácilmente a cualquier cantidad de segmentos y *displays*.

En la lista de componentes facturados para LT7S-HN (ver Documento Presupuesto, sección 1.11) se ha cobrado lo equivalente a la construcción de un LT7S-HN de 6 *displays* y 7 segmentos, el mismo que se ha utilizado en la claqueta digital de ejemplo. El material necesario para construir un LT7S-HN con distinta cantidad de *displays* y/o segmentos, tendría distinto coste final.

4.3.8 CMIR-RC

Este Módulo es un receptor de datos mediante luz infrarroja. Se utiliza principalmente para recibir señales codificadas enviadas desde controles remotos comerciales para televisiones, reproductores de vídeo y similares.

El componente principal del circuito es el receptor TSOP1736. Este circuito integrado es un receptor infrarrojo con filtro de paso de banda y demodulador integrado, de manera que el propio TSOP1736 procesa la señal infrarroja recibida y la convierte en una onda cuadrada limpia que se envía al microcontrolador, y puede ser leída fácilmente desde una patilla de entrada digital.

El circuito originalmente se ha diseñado y probado utilizando un TSOP1736, el cual puede recibir señales infrarrojas moduladas a 36kHz. En caso de que se utilice un control remoto que transmita señales moduladas a distinta frecuencia, pueden utilizarse otros modelos como el TSOP1730, TSOP1738, TSOP1740, etc (ver *datasheet* “TSOP17xx.pdf” en el CD adjunto). Además, existen otras familias de receptores infrarrojos que pueden usarse en su lugar, como TSOP48xx, SFH51xx, etc. No obstante, esto no es recomendable y en la mayoría de casos requerirá una pequeña modificación en el conexionado de los componentes. En el mismo CD adjunto se ha incluido también la información necesaria para construir este Módulo utilizando un SFH5110, además del TSOP1736.

De manera similar al resto de Módulos, el código fuente de ejemplo incluido con este Módulo (ver Anexo XVI) solamente sirve para comprobar que el CMIR-RC es capaz de recibir datos correctamente. No obstante, si se quiere procesar completamente la señal recibida y ser capaz de leer el código enviado por el control remoto, el usuario debe implementar sus propias funciones para microcontrolador que procesen la señal recibida, según el protocolo empleado por el control remoto que esté utilizando.

Este Módulo se utiliza de manera simple en el robot octópodo de ejemplo (sección 6.2).

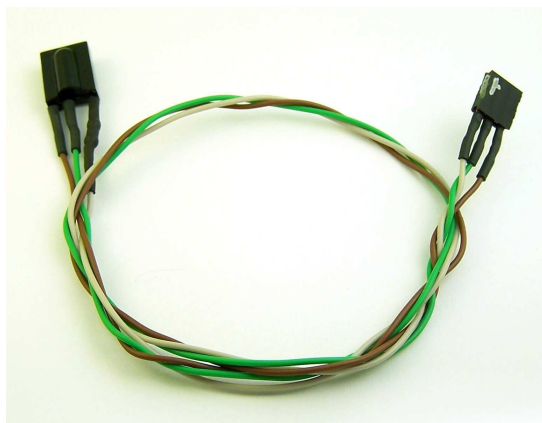


Figura 4.24

4.3.9 MC2A

MC2A es un Módulo que puede controlar dos motores de corriente continua de hasta un amperio y hasta 36V, y cada uno puede encontrarse en cuatro estados distintos, independientemente del estado del otro motor:

- Gira hacia la derecha

- Gira hacia la izquierda
- Gira libremente
- Frenado eléctricamente

Los pines necesarios para controlar de cada motor, así como el resto de conectores que se muestran en el diagrama de cableado (ver Documento Planos, plano N°13, hoja N°2) se describen a continuación:

- A. Pines digitales del motor 2
- B. Pines digitales del motor 1
- C. Alimentación principal. El pin 1 pueden ser 4.5-7V. El pin 2 es la masa
- D. Alimentación de los motores. Opcional. Si se desea alimentar los motores a tensión distinta que la principal, puede utilizarse este conector (4.5-36V). En caso contrario, no se utiliza
- E. *Jumper* selector de alimentación de los motores. Si se desea alimentar los motores a tensión distinta que la principal, conectar la fuente de alimentación secundaria al conector D y desconectar E. En caso de que se desee alimentar todo el circuito, incluyendo los motores, desde la alimentación primaria, deben cortocircuitarse estos pines con un *jumper*.
- F. Conectar al motor 1
- G. Conectar al motor 2

Los pines digitales de los motores 1 y 2 se distribuyen de la siguiente manera:

- **Pin N°1:** Habilidad. Cuando está a nivel digital alto, el motor recibe tensión. Si está a nivel bajo, el motor gira libremente.
- **Pines N°2 y N°3:** Entradas digitales. Cuando el pin N°1 está a nivel alto, estos pines indican la dirección de giro del motor. En caso de que ambos estén en nivel alto o ambos en nivel bajo, el motor estará frenado eléctricamente.

El circuito incluye ocho diodos que evitan que a causa de la inercia de los motores, estos se comporten como generadores de electricidad y generen picos de tensión que puedan quemar el circuito integrado.

El circuito puede entenderse en mayor profundidad a través de su esquema electrónico (ver Documento Planos, plano N°13, primera página) y del *datasheet* del circuito integrado L293B (ver CD adjunto).

Este circuito es lo suficientemente complejo como para requerir placa de circuito impreso para su fabricación, debido a que su construcción utilizando placa perforada resultaría en un entresijo de pistas excesivamente grande y complicado de construir. Para poder fabricar el PCB es necesario el uso del fotolito que puede encontrarse en el Documento Planos, plano N°13, segunda página.

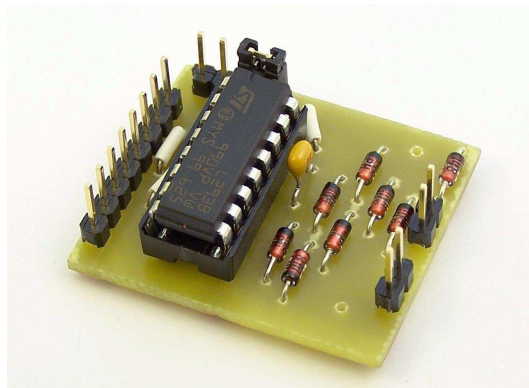


Figura 4.25

4.3.10 MC2B

MC2B es un Módulo que puede controlar dos motores de corriente continua de hasta 600mA a 5V. Ésta es una versión simplificada del Módulo MC2A, que utiliza el circuito integrado L293D, muy parecido al L293B/L293NE pero que soporta menos potencia e incluye diodos de protección internos. Además, en este circuito los motores utilizan la alimentación principal, y no es posible utilizar una fuente externa a mayor tensión.

El diagrama de cableado (ver Documento Planos, plano N°14) incluye los siguientes conectores:

- A. Conectar al motor 1
- B. Pines digitales del motor 1
- C. Alimentación. El pin 1 pueden ser 4.5-7V. El pin 2 es la masa
- D. Conectar al motor 2
- E. Pines digitales del motor 2

Los pines digitales de control de los motores funcionan de manera similar a MC2A. Debido a esto, las conexiones de datos para MC2A y MC2B son compatibles y los programas desarrollados para utilizar MC2A pueden utilizarse con MC2B y viceversa. Por lo tanto, no se incluye ningún programa de ejemplo de MC2B para microcontrolador. Para comprobar que MC2B funciona correctamente puede utilizarse el ejemplo de MC2A (ver Anexo XVII).

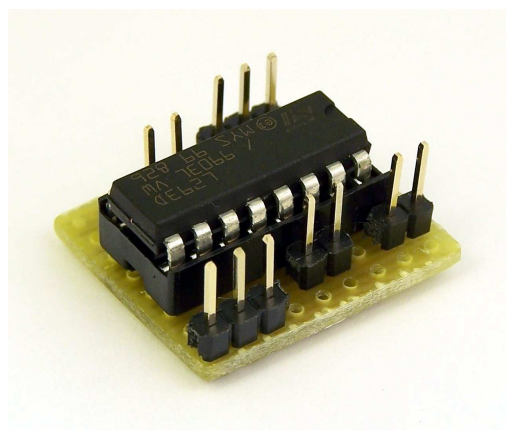


Figura 4.26

4.3.11 PWR-DD5

Éste es un Módulo regulador de tensión muy simple. Se utiliza para alimentar las Placas Principales (y, por tanto, todo el circuito) a 5V desde una batería o fuente de alimentación de corriente continua de entre 7,5 y 35V.

Debe conectarse la batería o fuente de alimentación al conector macho de este Módulo. La tensión de salida ofrecida en el conector hembra será de 5V ($\pm 0.3V$ aproximadamente). El pin marcado como N°1 es siempre el polo positivo, y el otro pin es la masa de todo el circuito (ver conector de alimentación, sección 4.1.2). La tensión se regula utilizando el circuito integrado 7805 (L7805, LM7805, LM7805T, L7805CV, etc., según fabricante). Este componente es simple, barato y muy común, pero también es muy ineficiente, por lo que disipará mucho calor cuando por él atraviesen más de unos pocos miliamperios. Este calor será energía desaprovechada de las baterías, y podría requerir el uso de un disipador para poder continuar funcionando, por lo que este Módulo solamente se recomienda para alimentar circuitos digitales de baja potencia. Si se pretende controlar motores de corriente continua mediante MC2A (sección 4.3.9) utilizando baterías de más de 5V se recomienda alimentar el circuito principal a través PWR-DD5, desconectar el *jumper* de selección de alimentación de MC2A, y utilizar el conector de alimentación externa para alimentar sólo los motores directamente desde las baterías.

A continuación se muestran todos los cálculos realizados para que el usuario pueda seleccionar la tensión de entrada adecuada, compruebe si necesita utilizar disipador, y qué tamaño de disipador necesitaría en caso de consumir mucha corriente. Todos ellos han sido basados en los datos del fabricante y en mediciones propias utilizando el regulador L7805 de STMicroelectronics. El Módulo podría construirse otros modelos similares del mismo u otros fabricantes, pero los parámetros podrían variar ligeramente. Las gráficas a mayor resolución pueden encontrarse en el CD adjunto.

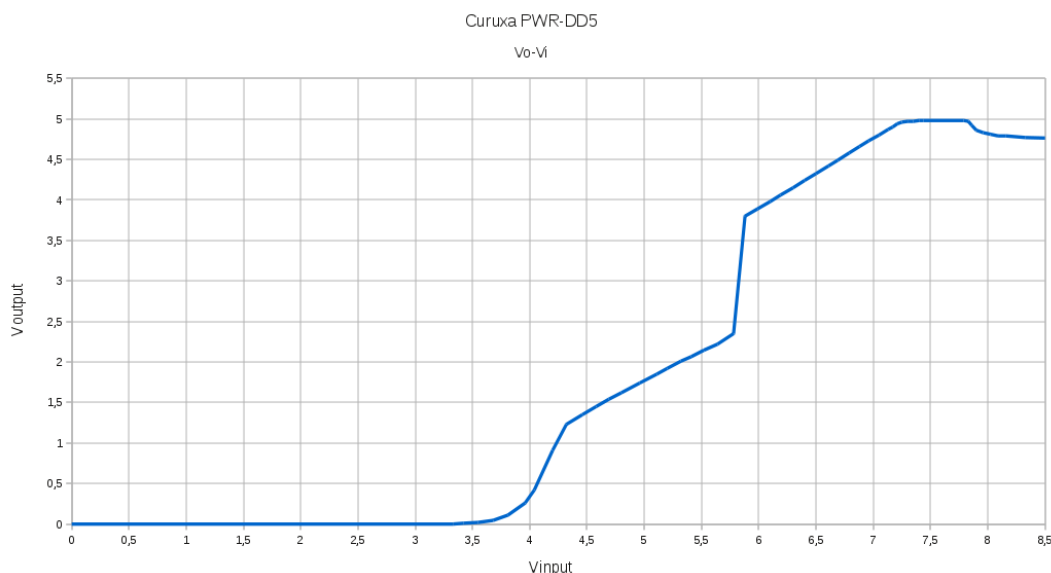


Figura 4.27

La figura 4.27 muestra la tensión de salida del Módulo en función de la tensión de entrada. La mayoría de Placas Principales y Módulos podrían funcionar a partir de 2,5-3V, por lo que siendo alimentados desde una batería y a través de PWR-DD5, podrían funcionar con tensiones de entrada a partir de 5,5-6V. No obstante, los 5V recomendados no se alcanzan hasta tensiones de entrada de al menos 7-7,5V. Por lo tanto, si se pretende alimentar el

circuito utilizando baterías de 4.8-5.5V no es necesario utilizar PWR-DD5, el cual se necesita cuando las baterías o fuente de alimentación superan los 7,5V. Con tensiones de entrada de entre 8,5 y 35V la tensión de salida se estabiliza en aproximadamente 4,8V, tensión adecuada para todas las Placas Principales y Módulos.

La figura 4.28 muestra la corriente máxima que puede circular a través del Módulo, según la temperatura ambiente y tensión de entrada. Corrientes mayores que las indicadas harán que el 7805 alcance su temperatura máxima de trabajo y se desconecte automáticamente. El componente no quemará, pero el circuito dejará de funcionar durante un tiempo. En caso de que se necesiten mayores corrientes que las aquí indicadas se hará necesario el uso de un disipador.

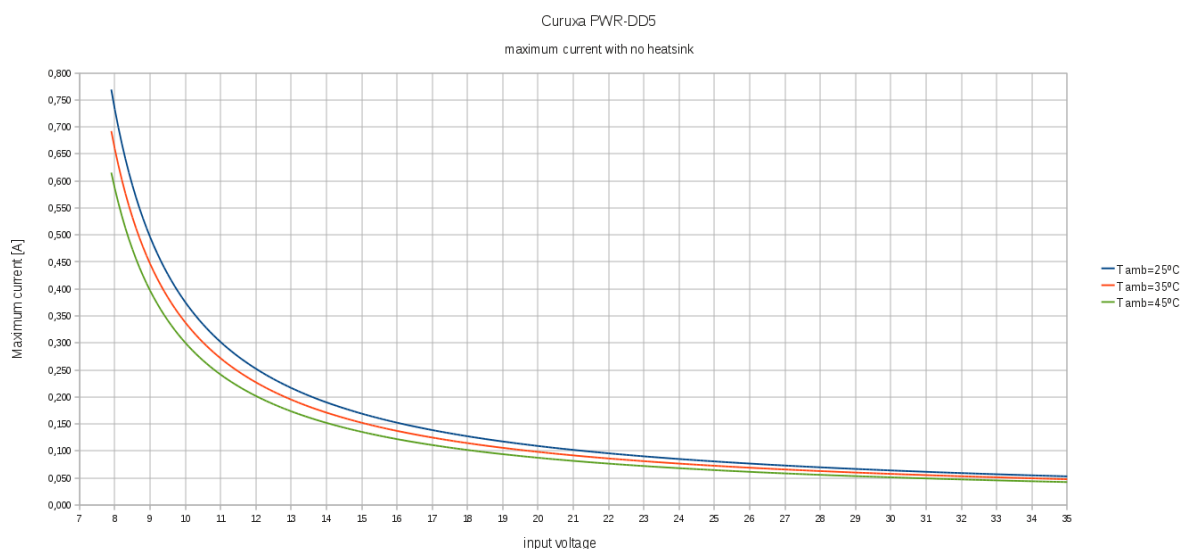


Figura 4.28

En caso de que la corriente necesaria sea mayor que la máxima indicada en la figura 4.28, debe utilizarse un disipador para el 7805 con una resistencia térmica menor que la indicada en las figuras 4.29, 4.30 y 4.31. Cuanto menor sea la resistencia térmica del disipador escogido, más calor podrá disipar y el 7805 podrá trabajar a menor temperatura, pero el disipador tendrá un mayor tamaño y precio. No deben utilizarse disipadores con resistencias térmicas mayores a las indicadas, lo cual haría que el 7805 se calentase excesivamente y el circuito dejaría de funcionar.

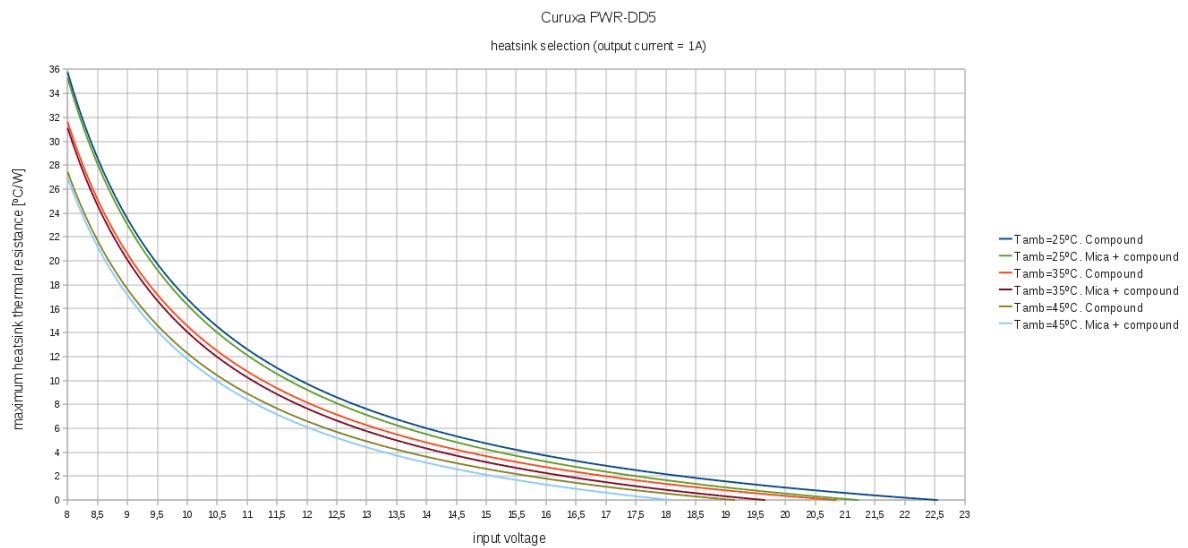


Figura 4.29

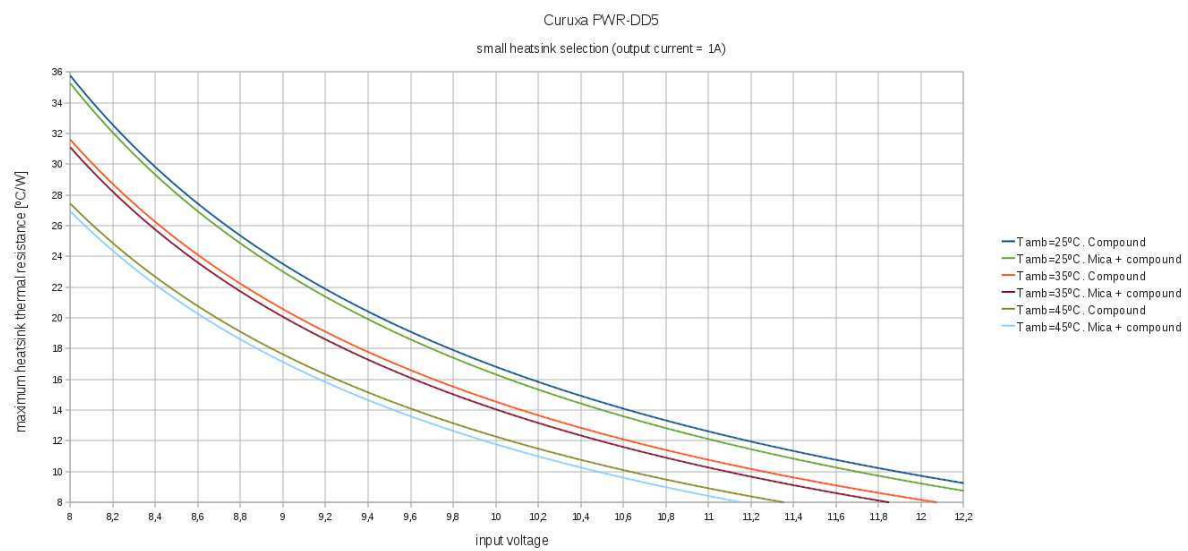


Figura 4.30

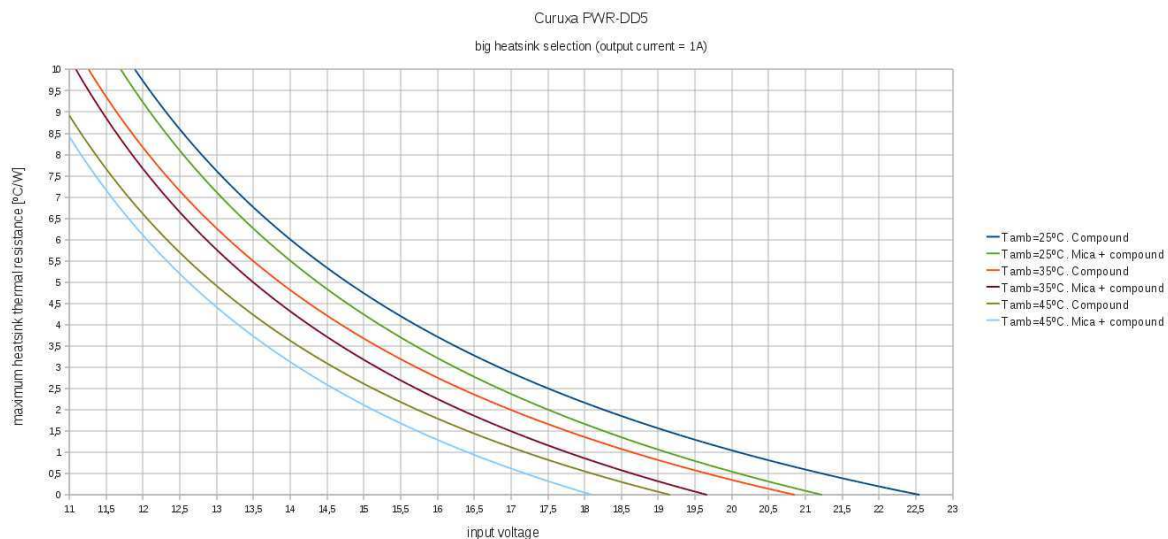


Figura 4.31

Como ejemplo, si se desea alimentar todo el circuito a 5V y se sabe que consumirá aproximadamente 1A en total, pero sólo se dispone de baterías de 9,3V, deben conectarse las baterías a la entrada de PWR-DD5, y la salida de este Módulo debe conectarse a la Placa Principal que se esté utilizando. En la figura 4.28 se comprueba que la corriente es demasiado elevada y se necesita utilizar un disipador para el encapsulado TO-220 (el utilizado por este circuito integrado). En las sucesivas figuras se calcula que el disipador necesario debe tener, como mucho 23°C/W de resistencia térmica, o bien 17°C/W si el circuito va a utilizarse en ambientes calurosos. Esta tensión de entrada no es demasiado elevada y no se disipa demasiada energía, así que aunque se necesite un disipador, éste no es demasiado grande. Habitualmente se comprarán disipadores de unos 3°C/W, que para este circuito servirá perfectamente.

Este regulador de tensión es cortocircuitable, pero deben respetarse las polaridades, especialmente la de salida, ya que una conexión invertida podría dañar el resto de Placas Principales y Módulos.

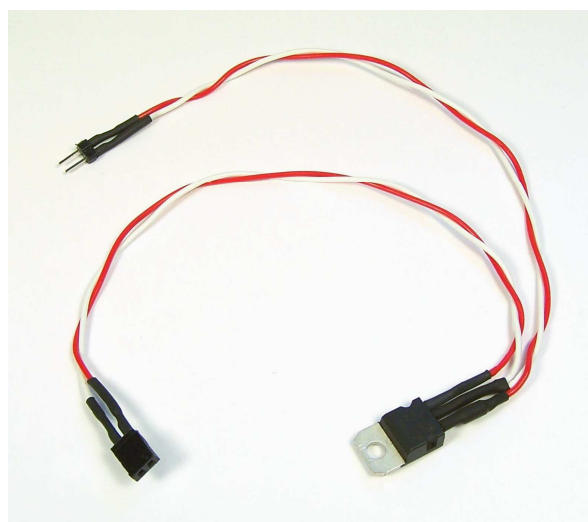


Figura 4.32

Este Módulo pretende ser lo más simple posible para alimentar el circuito a mayor tensión que 5V, por lo que se han omitido los condensadores que habitualmente se colocan a la entrada y salida del circuito. Esta decisión no debería afectar de ninguna manera al circuito, ya que el manual de referencia del fabricante especifica que el condensador en la entrada sólo es necesario si la fuente de alimentación está situada demasiado lejos, y el condensador en la salida mejora la respuesta en transitorio del regulador, pero no afecta a la estabilidad de la tensión de salida.

4.3.12 PWR-AD5

De manera similar a PWR-DD5, este Módulo es un regulador de tensión con salida de 5V. No obstante, PWR-AD5 acepta tensiones de entrada tanto de corriente continua como alterna. El uso habitual de este Módulo es el de alimentar el resto del circuito a la tensión adecuada desde la red eléctrica a 230V_{AC}, conectando un transformador cuyo primario se enchufa directamente a la red eléctrica y su secundario se conecta a la entrada de PWR-AD5. El conector de entrada no tiene polaridad, pero ésta debe respetarse en el conector de alimentación de salida, el cual irá habitualmente conectado a una Placa Principal.

Éste es el circuito conversor de corriente alterna a continua más simple posible, pero al igual que PWR-DD5, utiliza el ineficiente 7805, por lo que sólo se recomienda su uso para circuitos de muy baja potencia.

Si la batería o fuente de alimentación disponible ofrece 7,5-35V de corriente continua, se recomienda utilizar PWR-DD5. En cambio, si la tensión de entrada se desconoce si es continua o alterna, o se sabe que será una tensión alterna de entre 7Vrms y 25Vrms, debe utilizarse PWR-AD5.

Es importante seleccionar el condensador C1 (ver Documento Planos, plano N°16) adecuadamente. La figura 4.33 permite seleccionarlo fácilmente en función de la tensión eficaz y frecuencia de entrada, así como de la corriente necesaria para el circuito. Este condensador debe ser de tipo electrolítico, y su capacidad no debe ser menor que la indicada en esta figura, así como su tensión nominal especificada se recomienda que sea de al menos el doble de la máxima tensión de entrada. En caso de que se desconozcan los datos de la tensión de entrada, un condensador de 4700uF y 35V suele ser una opción adecuada en la mayoría de casos, así que es el condensador utilizado en el ejemplo construido y facturado. No obstante, un condensador demasiado pequeño causará caídas de tensión en la salida, y un condensador demasiado grande desperdiciará dinero, espacio y energía. Por lo tanto, es recomendable seleccionar el condensador adecuado más pequeño posible pero sin que su capacidad sea menor que la indicada en la figura 4.33.

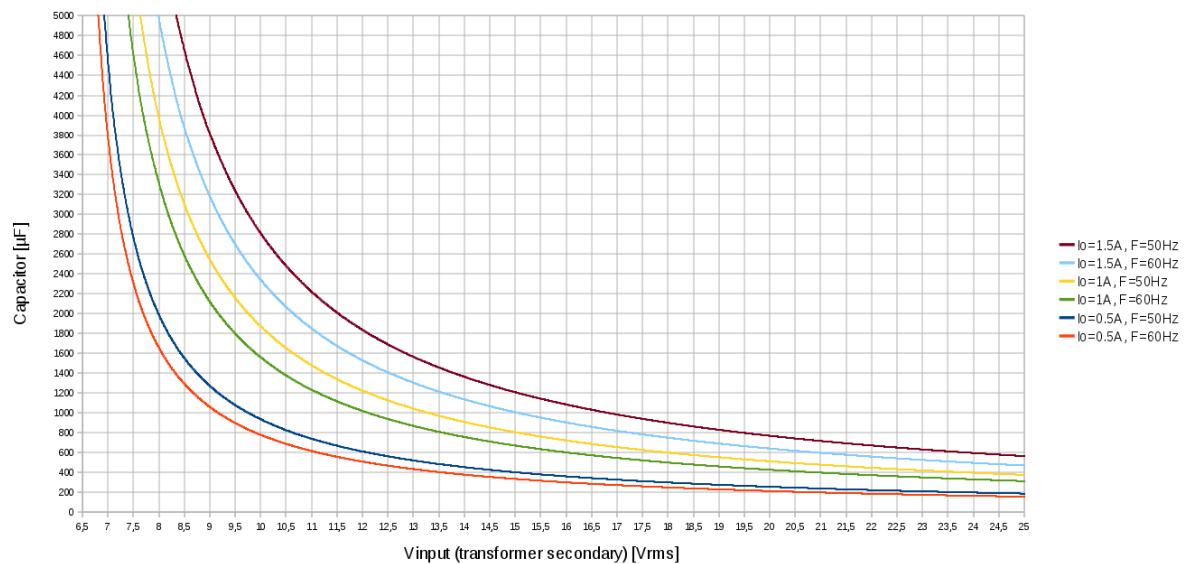


Figura 4.33

De manera similar al Módulo PWR-DD5, se necesitará utilizar un disipador para corrientes superiores a las indicadas en la figura 4.34.

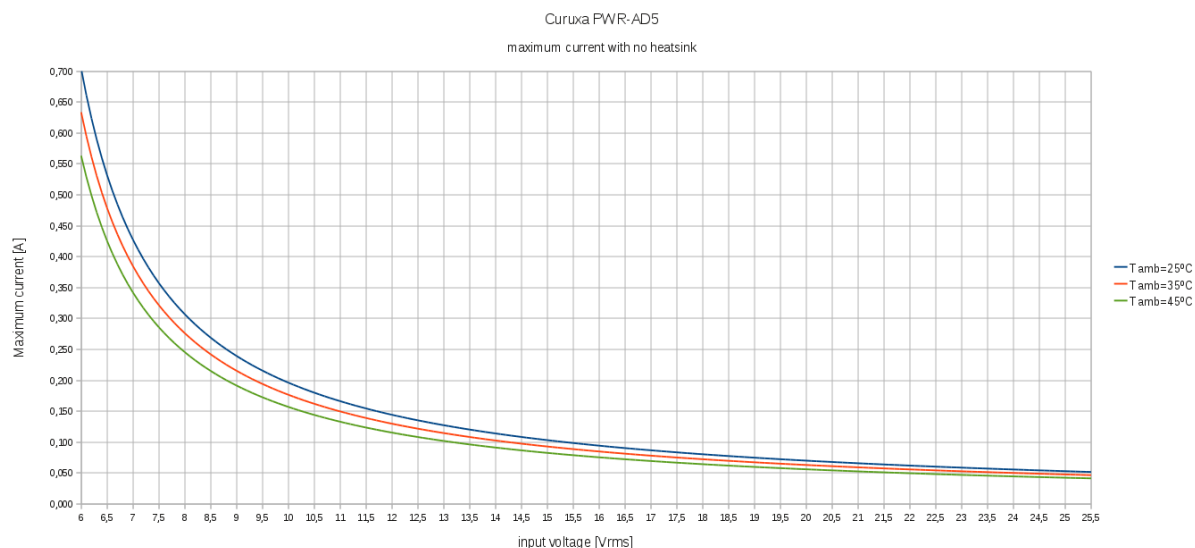


Figura 4.34

Las figuras 4.35 a 4.39 permiten elegir el tamaño del disipador. La figura 4.35 es para corrientes de hasta 100mA. La figura 4.36 hasta 500mA. La figura 4.37 detalla la resistencia térmica máxima para grandes condensadores al consumir 500mA. La figura 4.38 para consumos de hasta 1A, y la figura 4.39 para grandes disipadores hasta 1A.

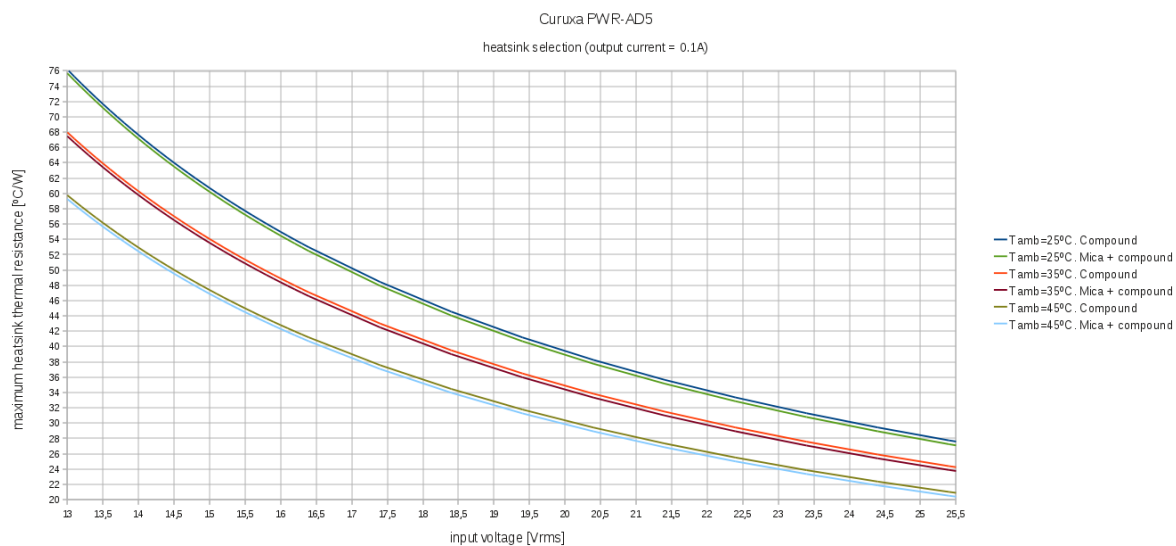


Figura 4.35

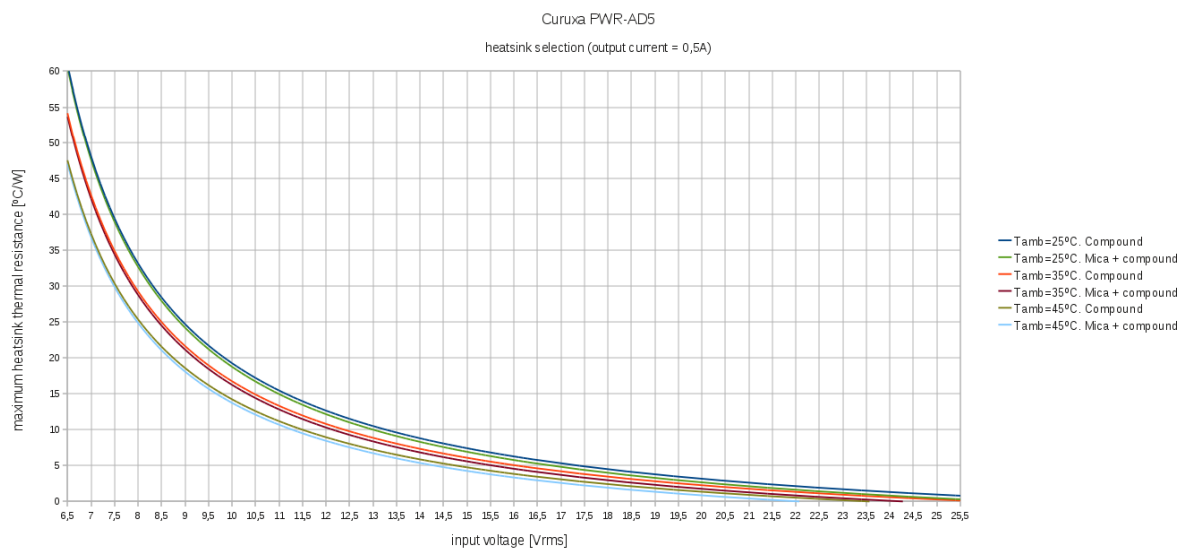


Figura 4.36

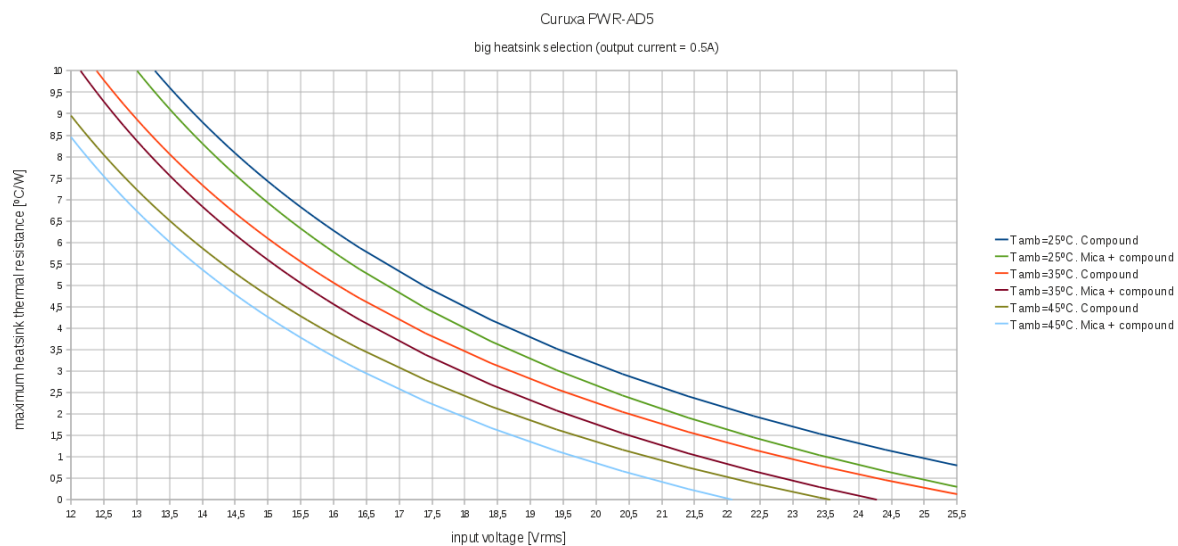


Figura 4.37

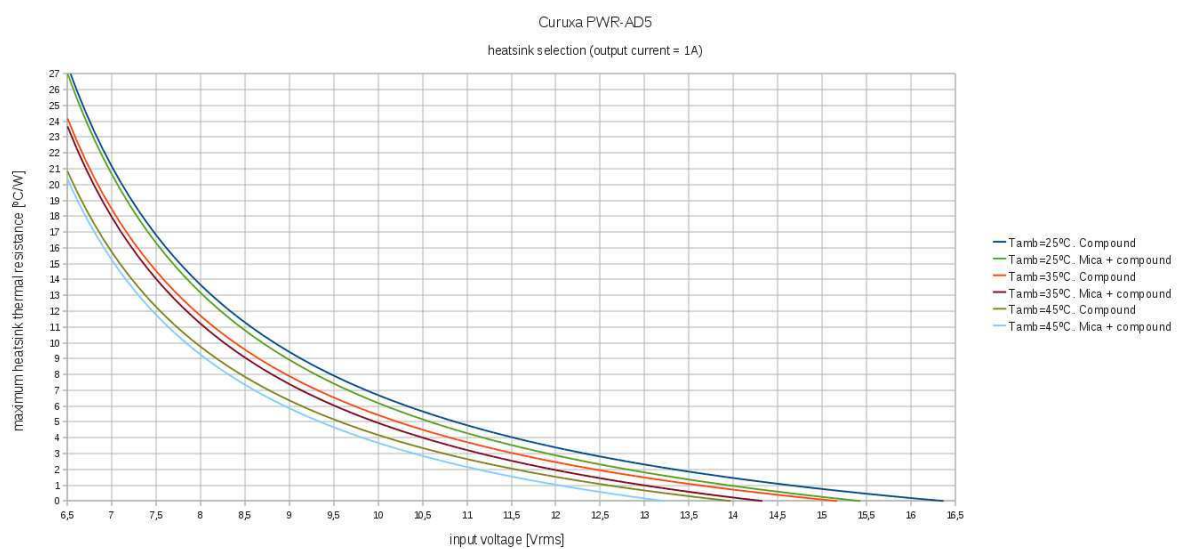


Figura 4.38

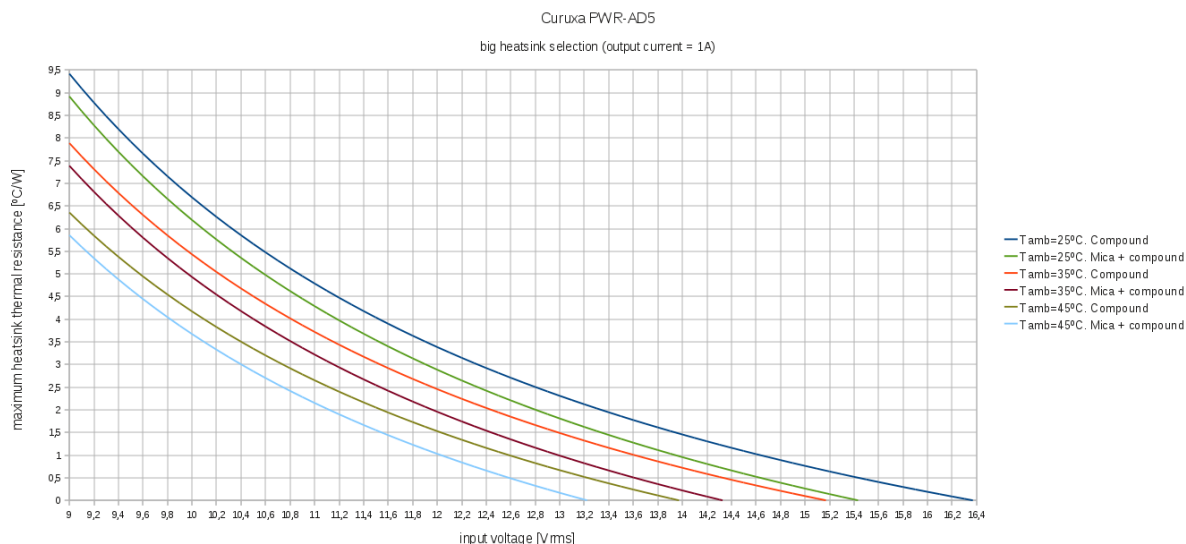


Figura 4.39

Por ejemplo, para alimentar un circuito que requiere 5V y no más de 100mA pero solo se dispone de una fuente de alimentación alterna de 13Vrms a 60Hz, un condensador electrolítico de al menos 400uF y tensión nominal de 16V sería suficiente (25V recomendado). Debe conectarse la fuente de alimentación a la entrada de PWR-AD5, y la salida del Módulo al resto del circuito. No se necesitaría ningún disipador.

En otro ejemplo se necesita alimentar el circuito a 5V y consume hasta 1W, pero se dispone de una fuente de alimentación alterna de 10Vrms y 50Hz. En esta caso se necesitará un condensador electrolítico de al menos 1900mF y 16V. También se necesitará un disipador para TO-220 con una resistencia térmica menor de 6.8°C/W.



Figura 4.40

4.3.13 CMSP-MAX

CMSP-MAX es un Módulo de comunicaciones que sirve para enviar y recibir información desde una Placa Principal a un ordenador y viceversa. El protocolo de comunicaciones se procesa utilizando la circuitería para UART integrada en todos los microcontroladores usados en todas las Placas Principales excepto MBP8.

La comunicación a través del puerto serie es el método más simple y fácil de implementar, tanto a nivel electrónico como de programación, por lo que es una buena

elección para los usuarios que deseen realizar algún tipo de comunicación entre su circuito y un ordenador incluso si no tienen experiencia previa. El puerto serie está mayormente obsoleto hoy en día, por lo que los ordenadores más actuales no suelen integrarlo. No obstante, es muy fácil conseguir adaptadores comerciales serie-USB por muy poco dinero.

Este Módulo no necesita procesar los datos de ninguna manera, ya que las señales pueden enviarse directamente desde el microcontrolador al ordenador y viceversa sin ningún cambio en el tipo de datos. No obstante, los niveles de voltaje utilizados por el microcontrolador y por el puerto serie son totalmente distintos, por lo que se utiliza el circuito integrado MAX232 para adaptar las señales. Además, éste requiere el uso de cuatro condensadores electrolíticos que se utilizan para generar las altas tensiones necesarias para transferir datos desde el MAX232 hasta el puerto serie del ordenador.

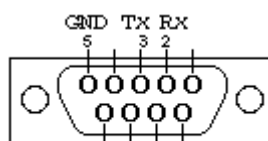


Figura 4.41

El circuito, construido en placa perforada, incluye dos conectores de datos (ver sección 4.1.1) para enviar y recibir datos entre el MAX232 y el microcontrolador, por lo que un extremo de ellos debe conectarse a CMSP-MAX y el otro extremo a la Placa Principal. Por otro lado, se incluye un conector similar de 3 pines que se utiliza para enviar y recibir datos entre el MAX232 y el puerto serie del ordenador. En este conector, el pin N°1 (*SP_RXD*, ver Documento Planos, plano N°17) debe conectarse al pin N°2 de una hembra DB9 (*RX*, ver figura 4.41), el pin N°2 (*SP_TXD*) del conector al pin N°3 (*TX*) de la hembra DB9, y el tercer pin (*GND*) del conector al pin N°5 de esta hembra DB9 (de nuevo, *GND*).

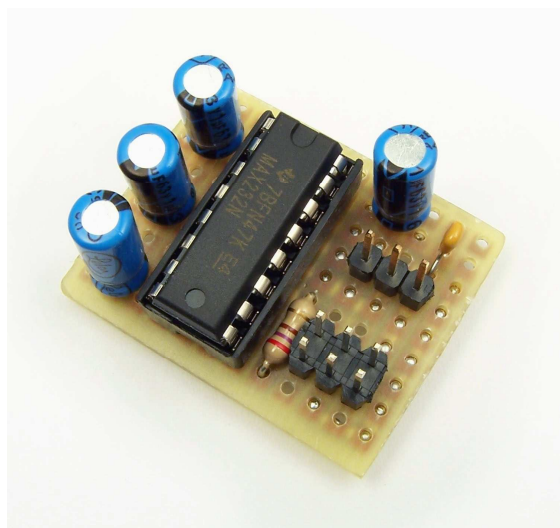


Figura 4.42

Junto con este Módulo se incluyen dos ejemplos de código para aprender a utilizarlo y comprobar que funciona correctamente. El primer ejemplo (ver Anexo XVIII) simplemente envía de manera continuada los números del 0 al 255 consecutivamente desde la Placa Principal hasta el ordenador, útil para comprobar que una transferencia simple de datos puede realizarse adecuadamente. El segundo ejemplo (ver Anexo XIX) puede recibir y enviar datos

desde y hacia el ordenador, y el código puede servir como base para cualquier producto completo que necesite comunicarse con un ordenador.

4.4. Módulos desarrollados por terceros

Independientemente de los Módulos incluidos en este proyecto (ver sección 4.3), varios usuarios ya han diseñado sus propios Módulos personalizados para ser utilizados junto con los Módulos y Placas Principales anteriormente descritas, y siguiendo los estándares internos del proyecto (ver sección 4.1). Estos Módulos personalizados **se incluyen aquí solamente como ejemplo** de las posibilidades de ampliación y libertad que tienen los usuarios de esta plataforma, pero no forman parte del proyecto en sí mismo y, por lo tanto, no se han incluido en el presupuesto.

A continuación se describen muy brevemente los Módulos personalizados que han sido desarrollados por Aarón González Iglesias.

4.4.1 *LTIND-FourSquare*

Módulo con cuatro LEDs rectangulares en una sola placa perforada, incluyendo resistencias. Este Módulo no utiliza el conector estándar de datos en la misma placa, pero sí puede utilizarse el conector de datos en el otro extremo del cable, pudiendo conectarlo a cualquier Placa Principal haciendo que cada uno de estos LEDs sea totalmente compatible con LTIND-A (ver sección 4.3.1).

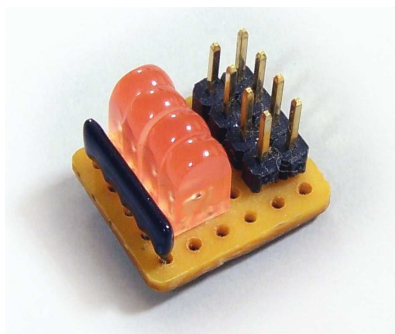


Figura 4.43

4.4.2 *Double SIBW*

Módulo con dos sensores de reflectividad colocados en una sola placa principal. Internamente, cada uno de estos sensores tiene circuitería y cableado independiente, haciéndolos totalmente compatibles con el Módulo SIBW-1Y (ver sección 4.3.4).

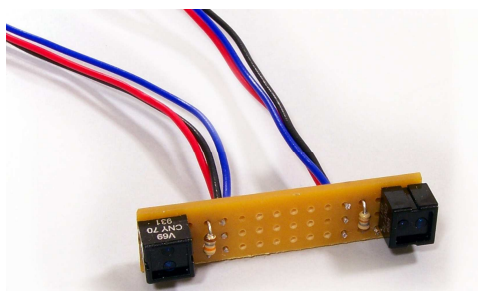


Figura 4.44

4.4.3 AO-SPK-Perfboard

Módulo que incluye un altavoz de baja potencia, eléctricamente idéntico a AO-SPK (ver sección 4.3.5) pero eliminando la resistencia R1, adaptado para poder ser construido en placa perforada. Puede conectarse, utilizarse y probarse de la misma manera que se haría con AO-SPK.

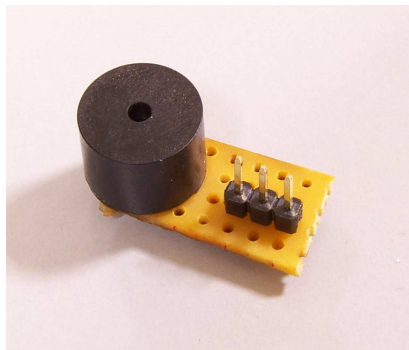


Figura 4.45

4.4.4 PWREG5

Regulador de tensión continua-continua, con salida de 5V, máximo 1A. La entrada puede ser de 7,5-35V, conectando las baterías o fuente de alimentación a los conectores atornillables. Incluye un conmutador que activa y desactiva el circuito completo, un LED indicador de funcionamiento, condensadores para estabilizar las tensiones de entrada y salida, y un conector de alimentación estándar para alimentar el resto del circuito a 5V.

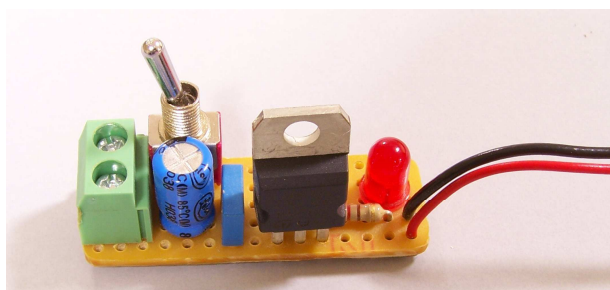


Figura 4.46

4.4.5 No-Wires Module Adapter

Módulo adaptador para poder conectar ciertos Módulos a una Placa Principal sin necesidad de utilizar cables, construido sobre placa perforada.

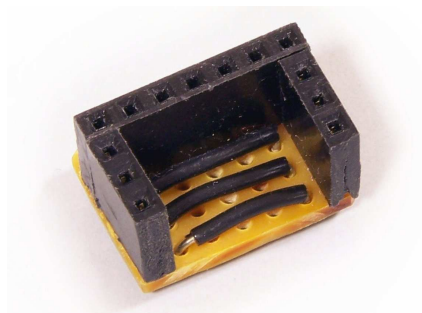


Figura 4.47

5. Herramientas informáticas

Aparte de los circuitos electrónicos que forman parte de la plataforma, descritos en el capítulo 4, también se han desarrollado un conjunto de herramientas informáticas diseñadas para facilitar al usuario el desarrollo de aplicaciones utilizando las Placas Principales y Módulos. Estas herramientas son tres: un entorno de desarrollo multiplataforma, un conjunto de librerías para microcontroladores, y una serie de *scripts* para trabajar con las Placas Principales desde la línea de comandos.

5.1. Entorno de desarrollo

El entorno de desarrollo es un programa para ordenador con interfaz gráfica y multiplataforma que permite acceder a todos los ejemplos de uso de todas las Placas Principales y Módulos disponibles en la plataforma, o bien crear proyectos de código desde cero para que el usuario desarrolle sus propias aplicaciones para microcontroladores, y las pueda compilar y grabar en el microcontrolador muy fácilmente, para luego alimentar el circuito desde el propio ordenador y así ejecutar el programa en desarrollo sin necesidad de recurrir a fuentes de alimentación externas.

El entorno de desarrollo más habitual para trabajar con microcontroladores PIC es MBLAB, oficial del fabricante. No obstante, MPLAB no soporta SDCC (el compilador utilizado en esta plataforma) y solamente funciona en Microsoft Windows, por lo que los usuarios de otros sistemas operativos no dispondrían de una manera sencilla de programar sus Placas Principales. La única solución encontrada para poder trabajar con esta plataforma fácilmente desde multitud de sistemas operativos es el empleo del entorno de desarrollo *Code::Blocks* junto con *PICKit2app*, el entorno gráfico para grabar los archivos con código máquina en el microcontrolador. No obstante, esta solución resultó ser bastante liosa y problemática.

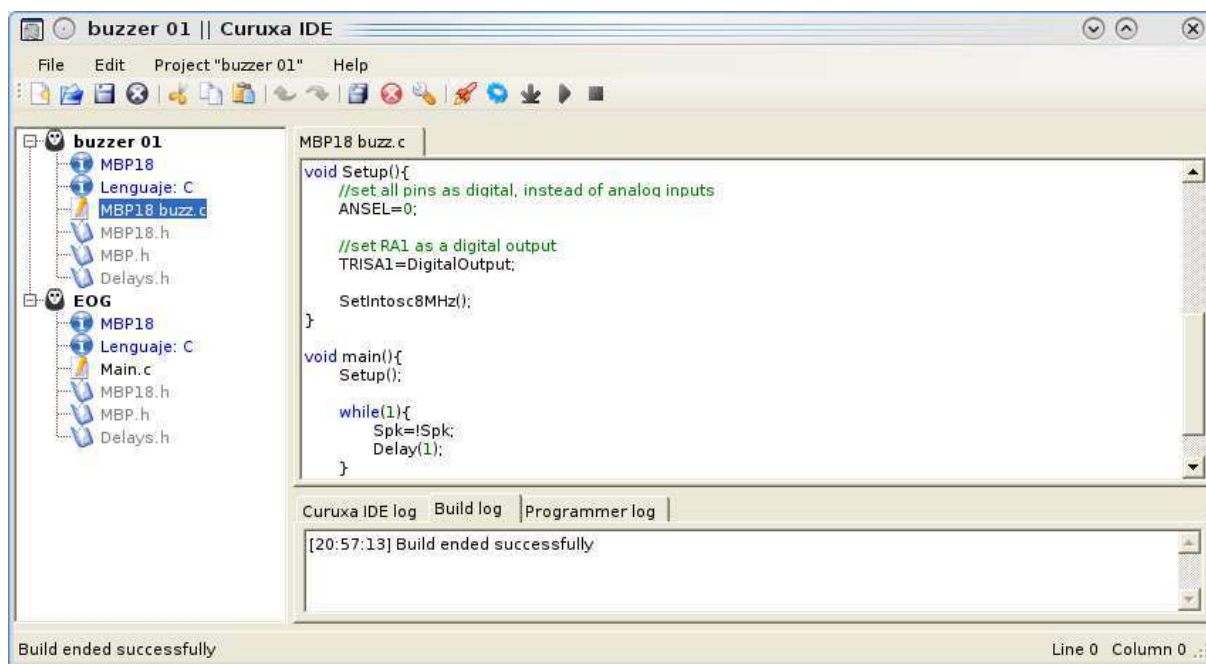


Figura 5.1

Por lo tanto, para poder de una manera muy sencilla programar las Placas Principales con SDCC desde cualquier sistema operativo, al mismo tiempo que se incluyen ejemplos de funcionamiento y plantillas para los circuitos propios de la plataforma, la mejor solución ha sido desarrollar a medida un entorno de desarrollo propio que, en la actualidad, cuenta con las siguientes características:

- Incluye ejemplos para comprobar el correcto funcionamiento de todas las Placas Principales y Módulos, así como algunos programas para aparatos completos contruidos utilizando estos circuitos (ver capítulo 6). Además, varios de estos ejemplos utilizan varios Módulos al mismo tiempo.
- Multiplataforma
 - Se ha probado en Microsoft Windows XP, Debian GNU/Linux y Ubuntu Linux
 - Se incluyen instaladores para Windows, Debian (y derivados), y el código fuente puede utilizarse para compilar y utilizar el programa en muchísimos otros sistemas operativos
 - Este entorno de desarrollo debería funcionar sin cambios en cualquier sistema operativo para el que haya disponible una implementación del CLI/.NET/Mono
- Soporta la programación y alimentación de microcontroladores utilizando el programador Microchip PICKit 2
- Multi-idioma
 - Detecta automáticamente el idioma del sistema
 - Puede cambiarse el idioma del programa sobre la marcha, sin reiniciarlo
 - En la actualidad, está traducido al inglés y al español
- Puede elegirse la Placa Principal utilizada en cada proyecto del usuario, así como el lenguaje de programación en que está desarrollada (lenguaje C o ensamblador) y la tensión a la cual se alimentará el circuito
- Pueden abrirse múltiples proyectos abiertos al mismo tiempo
- Pueden abrirse múltiples archivos de código fuente al mismo tiempo
- Automáticamente detecta las librerías a las que el código fuente hace referencia y permite acceder a ellas desde el árbol del mismo proyecto
- Con sólo tres botones se puede compilar el programa, grabarlo en el microcontrolador y ejecutarlo
 - Además, con un solo botón o pulsación de teclado, pueden ejecutarse estas tres acciones automáticamente

- Resalto (coloreado) de sintaxis del código fuente

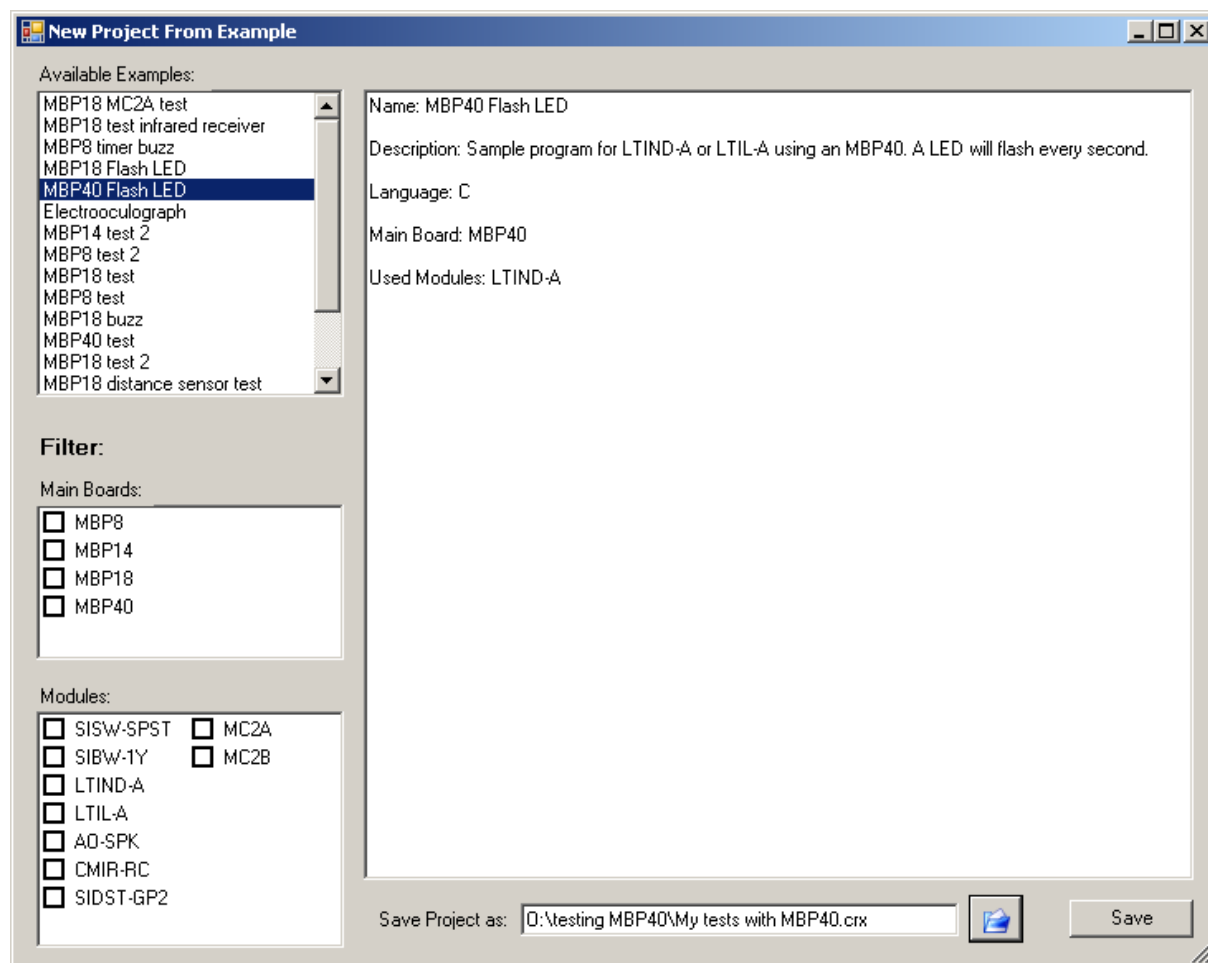


Figura 5.2

El código fuente completo del entorno de desarrollo se encuentra tanto en el Anexo XX como en el CD adjunto, donde también se encuentran los instaladores para Windows y Linux, y un video que ejemplifica la programación y prueba de varios LTIND-A conectados a una MBP40 siendo programada desde este entorno de desarrollo.

5.2. Librerías

Se han desarrollado también una serie de librerías para facilitar que el usuario pueda desarrollar sus propios programas para microcontrolador utilizando los circuitos electrónicos de esta plataforma.

Estas librerías han sido diseñadas para utilizarse con el compilador SDCC, al igual que el resto de código fuente para microcontroladores de esta plataforma. Éstas ofrecen funciones que ejecutan operaciones muy comunes y repetitivas, como por ejemplo configurar las patillas de entrada o salida digital, configurar el conversor analógico-digital, detener la ejecución del programa durante un tiempo determinado, o incluso simplifican la nomenclatura de los distintos tipos de variables soportadas por el compilador.

Cada una de las Placas Principales tiene su propia librería (*MBP8.h*, *MBP14.h*, *MBP40.h...*) y algunos de los Módulos también poseen sus propias librerías para poder

configurarlos o utilizarlos fácilmente. Además, existen varias librerías genéricas que pueden usarse en cualquier caso, como por ejemplo *Delays.h*.

El código fuente completo de estas librerías se encuentra en el Anexo XXI.

5.3. Scripts para línea de comandos

Muchos usuarios de diversos sistemas operativos, especialmente Unix y derivados (como Linux), utilizan a menudo la línea de comandos en sustitución de aplicaciones gráficas para agilizar y simplificar ciertas tareas. Por lo tanto, aunque el entorno de desarrollo incluido como parte de esta plataforma funciona en estos sistemas operativos, también se ofrecen una serie de *scripts* que simplifican las operaciones más habituales durante el desarrollo de aplicaciones para microcontroladores utilizando las Placas Principales.

Los *scripts* disponibles son los siguientes:

- **Burnpic:** Graba un archivo binario que contiene código máquina en el microcontrolador
- **Compile-mbp8:** Compila un programa escrito en C para el microcontrolador utilizado en MBP8.
- **Compile-mbp14:** Compila un programa escrito en C para el microcontrolador utilizado en MBP14.
- **Compile-mbp18:** Compila un programa escrito en C para el microcontrolador utilizado en MBP18.
- **Compile-mb40:** Compila un programa escrito en C para el microcontrolador utilizado en MBP40.
- **Curuxa:** Ejecuta el entorno de desarrollo utilizando Mono como máquina virtual
- **Detectpic:** Devuelve el modelo de microcontrolador conectado actualmente al PICKit 2
- **Runpic:** Alimenta la Placa Principal a 5V desde el PICKit 2, haciendo que el programa en su interior comience a ejecutarse
- **Stoppic:** Quita la alimentación de la Placa Principal, haciendo que el programa se detenga

Estos scripts han sido escritos en *Bash* para la consola de Unix o derivados, normalmente Linux, por lo tanto requieren que *Bash* o una shell equivalente se encuentre instalada en el sistema. Cabe destacar que es posible instalar *Bash* en Windows y utilizar estos *scripts* sin problemas, aunque es probable que casi ningún usuario decida hacerlo.

Todos los *scripts* que se comunican con Placas Principales lo hacen utilizando el programador Microchip PICKit 2, conectado mediante USB.

6. Aplicaciones de ejemplo

Como ejemplo de aplicación de las Placas Principales, Módulos, librerías y entorno de desarrollo, se han desarrollado los siguientes aparatos completos y funcionales.

Estas aplicaciones de ejemplo deben tomarse simplemente como ejemplo de cómo puede utilizarse esta plataforma para desarrollar dispositivos completos, pero estos dispositivos no pretenden ser productos útiles o con salida comercial. Además, esta plataforma solamente cubre circuitos electrónicos y programación de microcontroladores y ordenadores, por lo que la mecánica y el chasis quedan fuera de los objetivos de estas aplicaciones y, por tanto, no se incluyen en el proyecto.

6.1. Electrooculógrafo

Un electrooculógrafo (EOG) es un dispositivo que mide la tensión entre dos electrodos colocados en la cara de un paciente para detectar la posición de sus ojos en cada instante. Este aparato es una versión simple de un electrooculógrafo que puede utilizarse para detectar si el paciente está mirando hacia la derecha o hacia la izquierda. Para su construcción se han utilizado las siguientes Placas Principales y Módulos

- 1x MBP18
- 1x SISW-SPST
 - o Cuando el botón esté pulsado, el paciente debe mirar hacia delante y el EOG entrará en modo calibración.
 - o Debe conectarse a RB3 (pin 3) de MBP18
- 2x LTIND-A
 - o Dos LEDs utilizados para indicar en qué dirección está mirando el paciente en cada momento
 - o Debe conectarse un LTIND-A a RA2 (pin 1) y el otro a RA3 (pin 2) de MBP18
- 1x AO-SPK
 - o Un pequeño altavoz que generará dos tonos diferentes según la dirección en que esté mirando el paciente
 - o Debe conectarse a RA1 (pin 18)
- 1x Módulo de amplificación personalizado,
 - o Éste es un Módulo muy simple diseñado específicamente para esta aplicación, que contiene dos conectores para los electrodos que se colocarán en la cara del paciente, cuya diferencia de tensión será

amplificada por este Módulo y se enviará al microcontrolador a través de un conector de datos.

- o Esquemas, diagramas y fotografías de este Módulo personalizado pueden encontrarse en el CD adjunto
- o Debe conectarse a AN0 (pin 17)

Además de estos circuitos, también se han utilizado electrodos médicos adhesivos y dos pinzas de cocodrilo para unir los electrodos con el módulo amplificador.

El programa desarrollado para el microcontrolador (ver Anexo XXIII) lo primero que hace es configurar las entradas y salidas digitales y analógicas. A continuación, el programa espera a que se pulse el botón (SISW-SPST) para empezar a ser calibrado. Durante el tiempo que el botón está pulsado, el programa se mantiene en modo calibración, eliminando el ruido de la señal y calculando la tensión media que equivale a la tensión límite entre las diferentes tensiones recibidas cuando el paciente está mirando hacia un lado o hacia otro. Una vez que se libera el botón, el programa entra en modo de funcionamiento normal, que consiste en un bucle infinito que en cada pasada toma 200 mediciones, calcula la tensión media (para eliminar ruido, ya que la señal es excesivamente ruidosa) y, comparándola con la tensión media calibrada anteriormente, puede saber si el paciente está mirando hacia la derecha o hacia la izquierda.

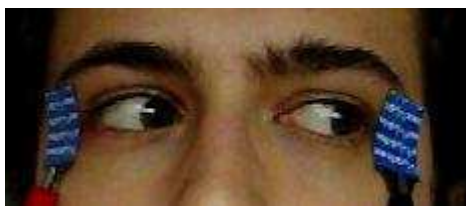


Figura 6.1

6.2. “KETJO” - Octópodo

Este aparato es un robot de ocho patas con una electrónica simple construido sobre un chasis de piezas de K’nex, un producto comercial basado en piezas de plástico interconectables. Las patas del robot están unidas de cuatro en cuatro a dos ejes movidos por dos motores de corriente continua de baja potencia. El circuito completo consta de los siguientes Módulos y Placas Principales:

- 1x MBP18
- 1x MC2A
 - o Este circuito controla los dos motores bidireccionales, y cada uno de ellos mueve cuatro de las ocho patas
 - o El conexionado de este Módulo a MBP18 puede verse en el código fuente (Anexo XXIV)
- 1x CMIR-RC

- o Receptor de señales infrarrojas
- o Debe conectarse a RB0 (pin 6) de MBP18

El chasis ha sido construido utilizando 845 piezas de K'nex y es un diseño original, aunque no forma parte de este proyecto. El movimiento de las patas traza una trayectoria muy realista, similar a la que trazarían las patas de un mamífero real, y está basado en el diseño mecánico de Theo Jansen. El robot completo pesa 1.53kg y está alimentado desde 4 pilas AA recargables colocadas en serie (4,8V en total).

El programa incluido de ejemplo (Anexo XXIV) genera un simple movimiento prepogramado que se inicia tras recibir una señal enviada desde un control remoto.

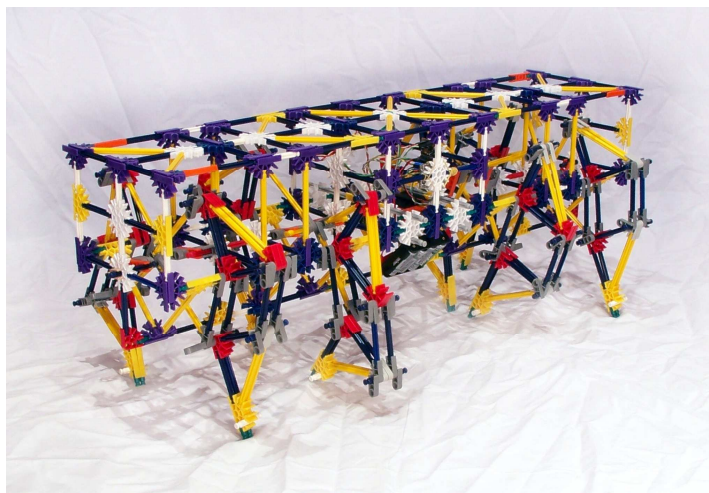


Figura 6.2

Más fotografías, videos y código fuente pueden encontrarse en el CD adjunto.

6.3. Claqueta digital

Este dispositivo es una claqueta común utilizada para indicar el inicio de la toma durante la grabación de videos profesionales. Código fuente y documentación disponible en el Anexo XXV y en el CD adjunto.

La claqueta incluye 6 grandes *displays* de siete segmentos, agrupados de dos en dos, de manera que cada uno de los tres bloques pueda representar los números del 0 al 99, e indiquen el número de escena, plano y toma. Para manejar y alimentar estos *displays* se utilizan los siguientes circuitos:

- 1x MBP40
- 1x PWR-DD5
 - o Disminuye la tensión de 9V a 5V para alimentar la parte digital del circuito
- 1x LTIND-A
 - o Si está encendido, indica funcionamiento normal

- o Si parpadea, indica que se encuentra en modo de bajo consumo
- 1x LT7S-HN
 - o Adaptado para 6 *displays* y 7 segmentos
- 7x SISW-SPST
 - o 6 botones, agrupados de dos en dos, se utilizan para incrementar y decrementar el valor mostrado en cada uno de los tres bloques de dígitos
 - o 1 botón se utiliza para conmutar entre el modo normal y el modo de bajo consumo

Debido a la gran cantidad de segmentos que suman los 6 *displays*, 42 en total, y la necesidad de alimentarlos a 7-10V debido a su gran tamaño, se ha utilizado un Módulo LT7S-HN adaptado para 6 *displays* y 7 segmentos, que permite el control de los 42 segmentos utilizando solamente 13 pines de salida digitales. Para ello, se emplea la técnica de multiplexado explicada en la sección 4.3.7, que consiste en mostrar el valor deseado en un solo *display* en cada instante de tiempo, haciendo que se encienda y se apague cada *display* decenas de veces por segundo y aparente estar siempre encendido, no parpadeando.

La claqueta posee un modo de bajo consumo el cual es activado y desactivado utilizando uno de los botones (SISW-SPST). Este modo de bajo consumo consiste simplemente en apagar todos los *displays* y esperar sin hacer nada hasta que se vuelve a activar el modo normal. Aunque tanto el microcontrolador como varios componentes más siguen consumiendo energía, la mayor parte de la corriente consumida por el circuito se emplea en activar los LEDs que forman los segmentos, por lo que el simple hecho de desactivarlos ahorra suficiente energía y aumenta en gran medida la vida útil de la batería. Cuando la claqueta no se esté usando, un conmutador desactiva la alimentación de todo el circuito. No obstante, el modo de bajo consumo se utiliza para poder desactivar la claqueta entre distintas tomas, ahorrando energía sin perder el valor anterior configurado para ser mostrado en los *displays*.

Al comienzo de funcionamiento de la claqueta, es decir, cada vez que recibe alimentación y el circuito completo comienza a funcionar, la claqueta entra en un modo inicial de comprobación, en el que se mantienen encendidos todos los segmentos de todos los *displays* hasta que un botón es pulsado y se inicia el modo de funcionamiento normal. Esto se hace para que, al comienzo de funcionamiento, pueda comprobarse que todos los segmentos funcionan correctamente.

Todo el circuito es alimentado desde una pila de 9V. Una pila de 9V común suele tener una capacidad de alrededor de 500mAh, por lo que se han calculado los siguientes ciclos de vida útil por cada pila usada en varias situaciones en las que se ha medido su consumo:

Estado	Horas
Todos los segmentos encendidos (modo de prueba inicial)	2,94
Uso normal, mostrando números aleatorios	5,29
Modo de bajo consumo	19,23

6.4. “PCBot”

PCBot es un robot con 2 ruedas motrices y 2 ruedas locas que incluye una cámara y un ordenador completo en su interior, el cual controla todo el circuito mediante una interfaz gráfica que se comunica con el circuito electrónico. Está formado por los siguientes circuitos:

- 1x MBP40
 - o La Placa Principal a la que se conectan todos los Módulos. Es la encargada de enviar los datos de los sensores al ordenador, así como de recibir las instrucciones enviadas por el ordenador y actuar en consecuencia. Por lo tanto, esta Placa Principal no se encarga de procesar ningún dato, sino que actúa como unión entre el ordenador, los sensores y actuadores.
- 2x LTIL-A
 - o Luces de media potencia colocadas en el frontal del ordenador y alumbran lo suficiente para que se pueda maniobrar en salas completamente oscuras utilizando la cámara integrada.
- 4x LTIND-A
 - o Indicadores de estado interno del programa para comprobar distintos estados de funcionamiento
- 2x SISW-SPST
 - o Colocados en las esquinas frontales actúan a modo de *búmper* o final de carrera para detectar choques contra otros objetos
 - o El contacto con otros objetos detiene el movimiento del robot, incluso si el ordenador no lo ha indicado explícitamente
- 1x MC2A
 - o Control bidireccional de los dos motores que mueven las ruedas motrices
- 1x CMSP-MAX
 - o Permite el envío y recepción de datos entre la Placa Principal y el ordenador

Además, se han utilizado 4 pilas AA recargables de NiMH, 1 portapilas para 4 pilas AA, 1 adaptador USB-serie ya que el ordenador que se ha utilizado no incluye puerto serie, 1 cámara conectada por USB al ordenador, 2 motores de corriente continua y un subportátil (*netbook*).

Se ha desarrollado un programa con una pequeña interfaz gráfica que se ejecuta en el ordenador para comunicarse con la Placa Principal (ver figura 6.3). Este programa se ha probado en Debian Linux y Microsoft Windows XP, aunque debería funcionar en muchos otros sistemas operativos sin ningún cambio. El establecimiento de la comunicación entre el programa ejecutado en el ordenador y la Placa Principal se realiza automáticamente. Una vez

conectado, el ordenador pide un código de autenticación al microcontrolador, de manera que si el MCU no responde con el código correcto la comunicación no puede continuar. Esto se hace para comprobar que la comunicación se esté realizando con el dispositivo adecuado y se esté llevando a cabo correctamente, sin fallos. El programa de ordenador puede encender y apagar las luces frontales individualmente o ambas al mismo tiempo, y puede establecer el movimiento de la base del robot (controlado por las ruedas motrices) en 7 posibles estados:

- **Movimiento frontal:** el robot se mueve hacia delante en línea recta
- **Girar hacia la izquierda:** leve giro hacia la izquierda, tomando como centro de giro el punto de contacto entre la rueda izquierda y el suelo
- **Girar hacia la derecha:** similar al anterior pero hacia la derecha
- **Rotar hacia la izquierda:** giro brusco hacia la izquierda, tomando como centro de rotación el punto medio del eje que une las dos ruedas motrices
- **Rotar hacia la derecha:** similar al anterior pero hacia la derecha
- **Movimiento hacia atrás:** el robot se mueve hacia atrás en línea recta
- **Detención:** la base del robot se detiene

No se han implementado giros ni rotaciones con movimiento hacia atrás para evitar colisiones

El programa de control que se ejecuta en el ordenador también recibe en tiempo real el estado de los bumpers, mostrando en pantalla imágenes que indican la colisión contra objetos.

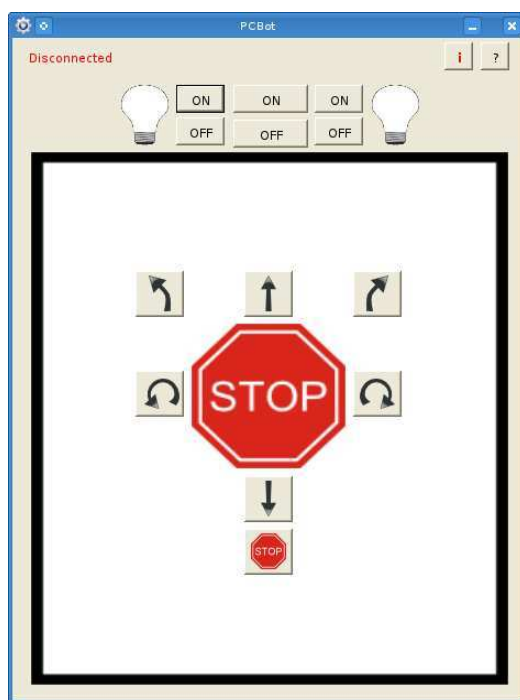


Figura 6.3

El objetivo de este robot es poder controlarlo de manera remota. Para ello, tanto la interfaz gráfica como la imagen ofrecida por la cámara integrada pueden ejecutarse en otros

ordenadores a través de redes inalámbricas mediante diversas técnicas existentes, como por ejemplo escritorios remotos (RDC, VNC...) o redirección por SSH, para poder mostrar esta interfaz gráfica e imagen de la cámara en otro ordenador y controlar el robot inalámbricamente.

Gracias al uso de un ordenador completo dentro del propio robot, las posibilidades de uso son enormes. Por ejemplo, la imagen de la cámara integrada puede grabarse en tiempo real para reproducirla en otro momento desde otro ordenador. También se ha podido sintetizar voz desde el propio ordenador y simular una conversación desde el propio robot. También se ha podido controlar el robot completo desde un teléfono móvil, comunicándose por *bluetooth* con el ordenador y éste enviando las instrucciones de movimiento a la Placa Principal. Además, aunque no se ha implementado ni probado, este robot podría utilizarse para aplicaciones mucho más complejas como visión artificial.

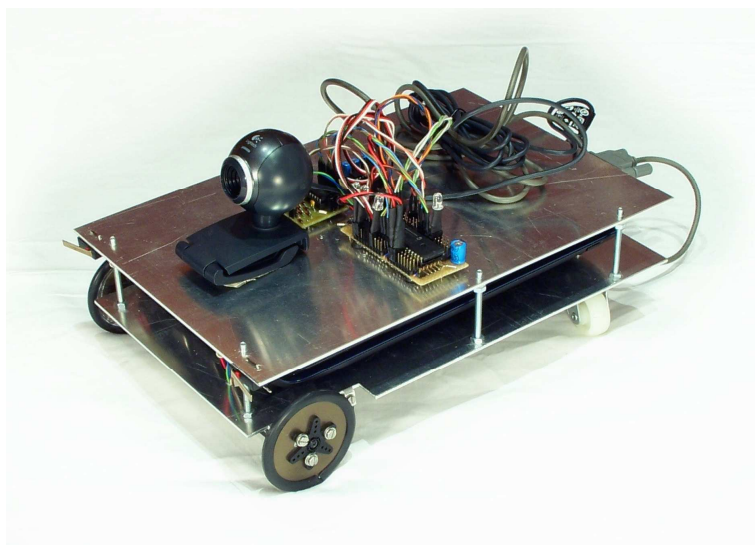


Figura 6.4

6.5. Aplicaciones desarrolladas por terceros

Aparte de las aplicaciones de ejemplo desarrolladas como parte de la plataforma para ejemplificar el uso de los circuitos diseñados, otros usuarios han desarrollado sus propios aparatos utilizando las Placas Principales y Módulos explicados en los capítulos 4.2 y 4.3, y utilizando para su implementación el entorno de desarrollo y librerías explicadas en el capítulo 5.

De manera similar a los Módulos desarrollados por terceros, a los que se hace referencia en la sección 4.4, estas aplicaciones no se han desarrollado como parte de la plataforma, sino que son construcciones propias de varios usuarios, se muestran aquí sólo como ejemplos de uso de la plataforma y no se han incluido en el presupuesto.

6.5.1 “POV” – Beltrán Fernández

Un POV (*Persistence of Vision display*) es un circuito electrónico basado en una serie de LEDs que, al hacer parpadear los LEDs rápidamente y mover el conjunto rápidamente, pueden “dibujarse” letras, números, e incluso dibujos simples “en el aire”.



Figura 6.5

Esta aplicación ha sido construida y programada por Beltrán Fernández utilizando un MBP18, ocho LTIND-A que se utilizan para generar la imagen virtual, y cuatro SISW-SPST utilizados para conmutar entre los distintos textos e imágenes disponibles.

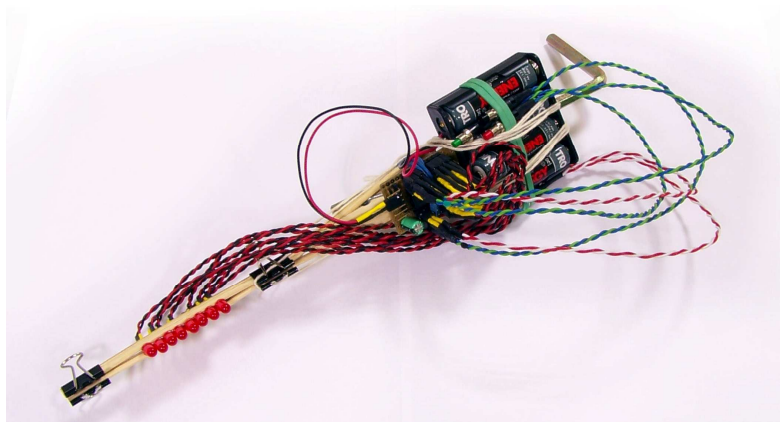


Figura 6.6

6.5.2 “Manu-fakture” – Manuel Fernández

Manu-fakture es un robot con dos ruedas motrices construido por Manuel Fernández utilizando un MBP18, un MC2A y seis SIBW-1Y y se puede utilizar para participar en las habituales competiciones de robots rastreadores o velocistas que se organizan en España. Este robot ha sido desarrollado como parte de los talleres de robótica explicados en el capítulo 7.

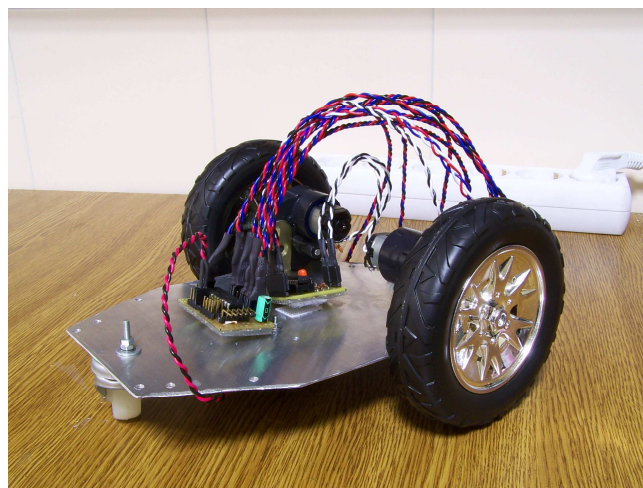


Figura 6.7

6.5.3 “Octapas” – Jorge Aparicio

Al igual que *Manu-fakture*, éste es un robot con dos ruedas motrices, un MBP18, un MC2A, seis SIBW-1Y y un SISW-SPST para conmutar entre distintos modos de funcionamiento, construido por Jorge Aparicio como parte de los talleres de robótica explicados en el capítulo 7.

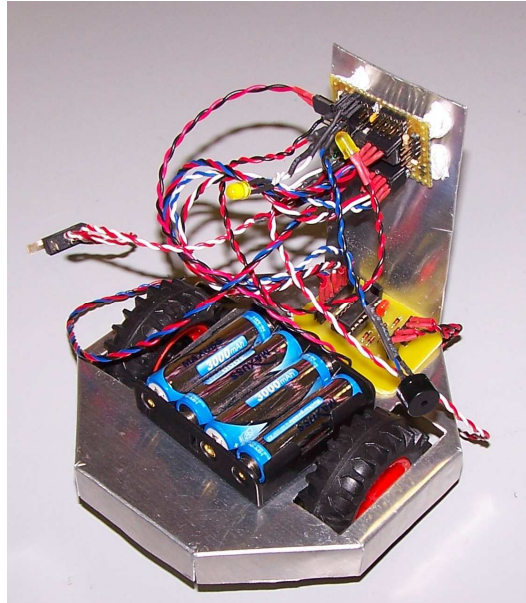


Figura 6.8

6.5.4 “CoyáN 2.0” – Rodrigo Fernández

De nuevo, éste es un robot construido por Rodrigo Fernández como parte de los talleres de robótica descritos en el capítulo 7, utilizando la misma circuitería que los robots anteriores pero, en este caso, se ha diseñado un pequeño Módulo personalizado que integra seis sensores de reflectividad en una sola placa principal, utilizando el mismo esquema electrónico, componentes y cableado que SIBW-1Y, por lo que este Módulo personalizado es totalmente compatible con el Módulo SIBW-1Y.

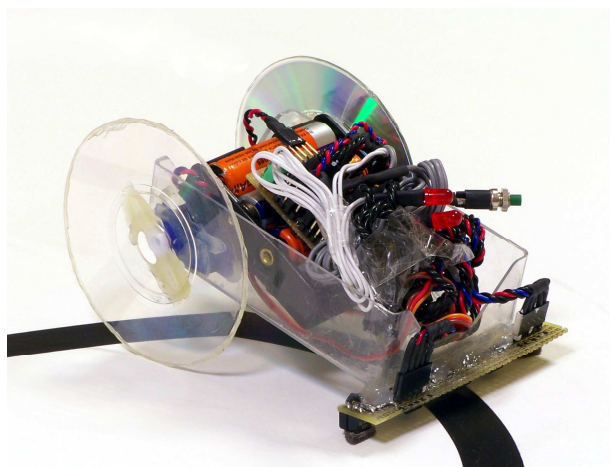


Figura 6.9

7. Publicaciones, Concursos, Talleres

7.1. IV CUSL

Esta *plataforma modular educativa para aprendizaje de microcontroladores* se ha presentado al **IV Concurso Universitario de Software Libre**, con el nombre de *Curuxa*, en el cual participaban 141 estudiantes de toda España repartidos entre 100 proyectos. Este concurso tuvo lugar entre octubre de 2009 y abril de 2010, periodo en que se desarrollaron los proyectos, para finalmente seleccionar 8 proyectos como finalistas del concurso. *Curuxa* fue uno de los ocho finalistas, por lo que junto con los otros siete finalistas, Adrián Bulnes (el autor de este proyecto) fue invitado a la Fase Final celebrada en la Universidad de Cádiz los días 13 y 14 de mayo de 2010, donde dio una charla de presentación del proyecto y fue agraciado con el **primer premio** como **Mejor Proyecto Innovador**.

En la figura 7.1 se muestran los ocho finalistas. Adrián Bulnes es el tercero por la derecha.



Figura 7.1

7.2. Artículo en Linux Magazine

El desarrollo de este proyecto ha dado lugar a la posibilidad de publicar el artículo “**Curuxa: Hardware libre para el aprendizaje**” en la edición N°62 de julio de 2010 en la **revista Linux Magazine**, para el cual se reservaron dos páginas completas de la revista en las que se explicaron los objetivos de la plataforma, su organización, las Placas Principales, los Módulos y las herramientas informáticas desarrolladas.

La versión PDF del artículo publicado se ha incluido en el CD adjunto.

7.3. Talleres de robótica

Además, durante el curso completo 2009/2010, esta plataforma se ha utilizado para los Talleres de Robótica organizados por la Rama de Estudiantes del IEEE de la Universidad de Oviedo.

Estos talleres, organizados por cuatro alumnos de Ingeniería Técnica Industrial, han contado con la participación de otros 42 estudiantes de diversas ingenierías. En ellos, entre los meses de octubre de 2009 y mayo de 2010, los participantes se han reunido entre 3 y 6 horas semanales y han recibido los componentes necesarios para construir una Placa Principal MBP18, tres Módulos LTIND-A, dos SISW-SPST, un AO-SPK, un MC2A (completo, con PCB construido por ellos mismos), seis SIBW-1Y, un SIDST-GP2, un PWR-DD5, un CMIR-RC y otros componentes como motores, aluminio y tornillos para poder construir sus propios robots siguelíneas desde cero. Además, los participantes han recibido las versiones imprimibles de cada una de las Placas Principales y Módulos construidos (disponibles en el CD adjunto), las cuales han resultado enormemente útiles para que no fuese necesario el uso de ordenadores durante la construcción de los circuitos.



Figura 7.2



Figura 7.3

8. Calendario

En esta sección se muestra el diagrama de Gantt (ver Figura 8.1) que describe las tareas llevadas a cabo durante el desarrollo completo del proyecto.

Muchas Placas Principales y Módulos dependen de MBP18 y LTIND-A debido a que se utilizaron como circuitos de referencia para probar el correcto funcionamiento de casi todos los demás Módulos.

Existen temporadas en las que no se trabajó directamente en este proyecto debido a que el tiempo de desarrollo del mismo tuvo que compaginarse con estudios y otros proyectos. Por la misma razón, el presupuesto (ver Documento Presupuesto) no se ha calculado como un trabajo a jornada completa, sino que se han facturado las horas reales empleadas en completar la tarea.

Al comienzo del desarrollo del proyecto se han agrupado multitud de tareas debido a que fue necesario desarrollar una cierta cantidad de circuitos antes del comienzo de los talleres de robótica en los que se iban a utilizar. El resto de tareas pudieron desarrollarse durante los siguientes meses.

Los Módulos con componentes importantes comunes (como PWR-DD5 y PWR-AD5) suelen haber sido diseñados uno a continuación del otro, habiendo empezado por diseñar el más simple.

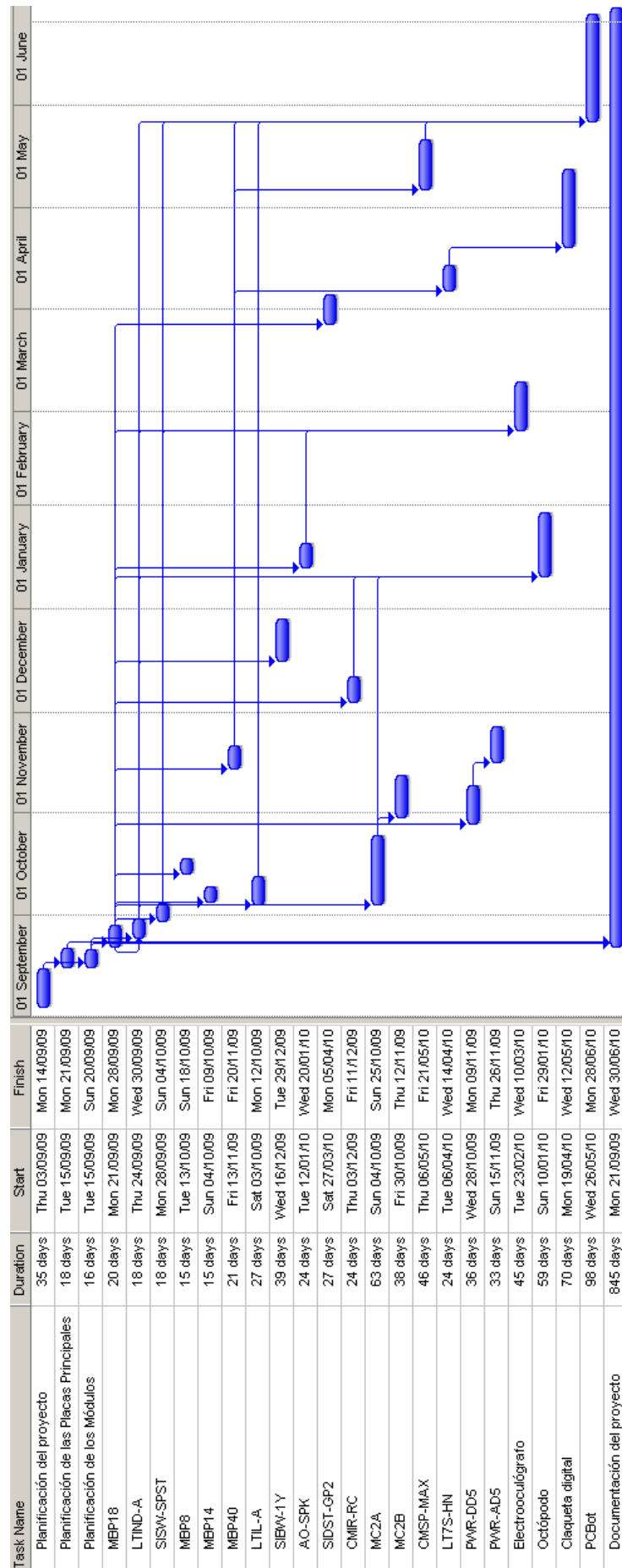


Figura 8.1

9. Bibliografía

- http://www.microchip.com.tw/PDF/2004_spring/PIC16F%20seminar%20presentation.pdf - "PICmicro Family Tree"
- "The C Programming Language" - Dennis M. Ritchie. Prentice Hall. 1988.
- <http://sdcc.sourceforge.net/> - Documentación y descargas de SDCC (ver sección 3.4.1)
- <http://msdn.microsoft.com/> - Documentación sobre C#, .NET y Visual Studio (entorno de desarrollo para C#, entre otros)
- <http://www.mcmanis.com/chuck/robotics/tutorial/h-bridge/> - "H-Bridges: Theory and Practice"
- <http://www.dprg.org/tutorials/1998-04a/> - "BRIEF H-BRIDGE THEORY OF OPERATION"
- http://www.x-robotics.com/sensores_ing.htm - "Sensor CNY70"
- <http://www.sbprojects.com/knowledge/ir/ir.htm> - "IR Remote Control"
- <http://www.taltech.com/resources/intro-sc.html> - "Introduction to Serial Port communications"
- <http://www.db9-pinout.com/> - "DB9 Pinout". Conector para puerto serie.
- <http://www.thaiio.com/prog-cgi/VBnetSerialPort.htm> - "Serial Port Communication in VB.NET Programming"
- "EIA Standard RS-232-C Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Data Interchange" - Electronics Industries Association. Agosto 1969.
- ...*datasheets* de todos los circuitos integrados utilizados (ver CD adjunto).

Anexo I

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo MBP8 #1, sin usar librerías propias*

Lenguaje: C

Archivos de código: 1

```
1: /*=====
2:  * Programa de ejemplo para MBP8 usando un PIC12F683
3:  *
4:  * Este programa configura todos los pines disponibles como
5:  * salidas digitales. Puedes comprobar que todos los pines
6:  * digitales están a la misma tensión que la alimentación del
7:  * circuito utilizando un voltímetro o el módulo LTIND-A
8:  * (el LED debería encenderse).
9:  * Nota: GP3 (pin 4) es un pin sólo de entrada.
10:  *
11:  * Adrián Bulnes Parra
12:  *=====*/
13:
14: #include <pic12f683.h>
15:
16: typedef unsigned int config;
17: config at 0x2007 __CONFIG = _CPD_OFF & _CP_OFF & _BOD_OFF & _MCLRE_OFF &
_PWRTE_ON & _WDT_OFF & _INTOSCIO;
18:
19: void main() {
20:     //configura todos los pines como digitales, no analógicos
21:     ANSEL=0;
22:
23:     //configura todos los pines digitales como salidas
24:     TRISIO=0;
25:
26:     while(1){
27:         GPIO=0b11111111;
28:     }
29: }
```

Anexo II

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo MBP8 #2, usando librerías propias*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Programa de ejemplo para MBP8 usando un PIC12F683
3:  *
4:  * Este programa configura todos los pines disponibles como
5:  * salidas digitales. Puedes comprobar que todos los pines
6:  * digitales están a la misma tensión que la alimentación del
7:  * circuito utilizando un voltímetro o el módulo LTIND-A
8:  * (el LED debería encenderse).
9:  *
10: * Este programa utiliza librerías propias.
11: *
12: * Nota: GP3 (pin 4) es un pin sólo de entrada.
13: *
14: * Adrián Bulnes Parra
15: *===== */
16:
17: #include <MBP8.h>
18:
19: ConfigBits(_CPD_OFF & _CP_OFF & _BOD_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT_
_OFF & _INTOSCIO);
20:
21: void main() {
22:     AdcDisable();
23:
24:     //configura todos los pines digitales como salidas
25:     TRISIO=AllDigitalOutputs;
26:
27:     while(1) {
28:         GPIO=0b11111111;
29:     }
30: }
```

Anexo III

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo MBP14 #1, sin usar librerías propias*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Programa de ejemplo para MBP14 usando un PIC16F688
3:  *
4:  * Este programa configura todos los pines disponibles como
5:  * salidas digitales. Puedes comprobar que todos los pines
6:  * digitales están a la misma tensión que la alimentación del
7:  * circuito utilizando un voltímetro o el módulo LTIND-A
8:  * (el LED debería encenderse).
9:  *
10:  * Nota: RA3 (pin 4) es un pin sólo de entrada.
11:  *
12:  * Adrián Bulnes Parra
13:  *=====*/
14:
15:  #include <pic16f688.h>
16:
17:  typedef unsigned int config;
18:  config at 0x2007 __CONFIG = _CPD_OFF & _CP_OFF & _BOD_OFF & _MCLRE_OFF &
_PWRTE_ON & _WDT_OFF & _INTOSCIO;
19:
20:  void main(){
21:      //configura todos los pines como digitales, no analógicos
22:      ANSEL=0;
23:
24:      //configura todos los pines digitales como salidas
25:      TRISA=0;
26:      TRISC=0;
27:
28:      while(1){
29:          PORTA=0b11111111;
30:          PORTC=0b11111111;
31:      }
32:  }
```

Anexo IV

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo MBP14 #2, usando librerías propias*

Lenguaje: C

Archivos de código: 1

```
1: /*=====
2: * Programa de ejemplo para MBP14 usando un PIC16F688
3: *
4: * Este programa configura todos los pines disponibles como
5: * salidas digitales. Puedes comprobar que todos los pines
6: * digitales están a la misma tensión que la alimentación del
7: * circuito utilizando un voltímetro o el módulo LTIND-A
8: * (el LED debería encenderse).
9: *
10: * Este programa utiliza librerías propias.
11: *
12: * Nota: RA3 (pin 4) es un pin sólo de entrada.
13: *
14: * Adrián Bulnes Parra
15: *=====*/
16:
17: #include <MBP14.h>
18:
19: ConfigBits( _CPD_OFF & _CP_OFF & _BOD_OFF & _MCLRE_OFF & _PWRTE_ON & _WD
T_OFF & _INTOSCIO);
20:
21: void main(){
22:     AdcDisable();
23:
24:     //configura todos los pines digitales como salidas
25:     TRISA=AllDigitalOutputs;
26:     TRISC=AllDigitalOutputs;
27:
28:     while(1){
29:         PORTA=0b11111111;
30:         PORTC=0b11111111;
31:     }
32: }
```


Anexo V

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo MBP18 #1, sin usar librerías propias*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Programa de ejemplo para MBP18 usando un PIC16F88
3:  *
4:  * Este programa configura todos los pines disponibles como
5:  * salidas digitales. Puedes comprobar que todos los pines
6:  * digitales están a la misma tensión que la alimentación del
7:  * circuito utilizando un voltímetro o el módulo LTIND-A
8:  * (el LED debería encenderse).
9:  *
10:  * Nota: RA5 es un pin sólo de entrada.
11:  *
12:  * Adrián Bulnes Parra
13:  *=====*/
14:
15:  #include <pic16f88.h>
16:
17:  typedef unsigned int config;
18:  config at _CONFIG1 __CONFIG = _CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _
CPD_OFF & _LVP_OFF & _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO;
19:
20:  void main(){
21:      //configura todos los pines como digitales, no analógicos
22:      ANSEL=0;
23:
24:      //configura todos los pines digitales como salidas
25:      TRISA=0;
26:      TRISB=0;
27:
28:      while(1){
29:          PORTA=0b11111111;
30:          PORTB=0b11111111;
31:      }
32:  }
```

Anexo VI

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo MBP18 #2, usando librerías propias*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Programa de ejemplo para MBP18 usando un PIC16F88
3:  *
4:  * Este programa configura todos los pines disponibles como
5:  * salidas digitales. Puedes comprobar que todos los pines
6:  * digitales están a la misma tensión que la alimentación del
7:  * circuito utilizando un voltímetro o el módulo LTIND-A
8:  * (el LED debería encenderse).
9:  *
10: * Este programa utiliza librerías propias.
11: *
12: * Nota: RA5 es un pin sólo de entrada.
13: *
14: * Adrián Bulnes Parra
15: *===== */
16:
17: #include <MBP18.h>
18:
19: ConfigBits1(_CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF &
& _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO);
20:
21: void main() {
22:     //configura todos los pines como digitales, no analógicos
23:     AdcDisable();
24:
25:     //configura todos los pines digitales como salidas
26:     TRISA=AllDigitalOutputs;
27:     TRISB=AllDigitalOutputs;
28:
29:     while(1) {
30:         PORTA=0b11111111;
31:         PORTB=0b11111111;
32:     }
33: }
```

Anexo VII

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo MBP40 #1, sin usar librerías propias*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Programa de ejemplo para MBP40 usando un PIC16F887
3:  *
4:  * Este programa configura todos los pines disponibles como
5:  * salidas digitales. Puedes comprobar que todos los pines
6:  * digitales están a la misma tensión que la alimentación del
7:  * circuito utilizando un voltímetro o el módulo LTIND-A
8:  * (el LED debería encenderse).
9:  *
10: * Este programa utiliza librerías propias.
11: *
12: * Nota: RE3 (pin 1) es un pin sólo de entrada.
13: *
14: * Adrián Bulnes Parra
15: *=====*/
16:
17: #include <MBP40.h>
18:
19: ConfigBits1(_CP_OFF & _DEBUG_OFF & _FCMEN_OFF & _IESO_OFF & _CPD_OFF & _
LVP_OFF & _BOR_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT_OFF & _INTOSCIO);
20:
21: void main() {
22:     //configura todos los pines como digitales, no analógicos
23:     AdcDisable();
24:
25:     //configura todos los pines digitales como salidas
26:     TRISA=AllDigitalOutputs;
27:     TRISB=AllDigitalOutputs;
28:     TRISC=AllDigitalOutputs;
29:     TRISD=AllDigitalOutputs;
30:     TRISE=AllDigitalOutputs;
31:
32:     while(1) {
33:         PORTA=0b11111111;
34:         PORTB=0b11111111;
35:         PORTC=0b11111111;
36:         PORTD=0b11111111;
37:         PORTE=0b11111111;
38:     }
39: }
```

Anexo VIII

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo para LTIND-A usando MBP18*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Programa de ejemplo para LTIND-A usando MBP18
3:  *
4:  * Este programa configura todos los pines digitales como salida.
5:  * Conecta el Módulo LTIND-A a cualquier pin. El LED se encenderá
6:  * y después se apagará cada segundo. RA5 es un pin sólo de entrada,
7:  * así que el LED no funcionará en él.
8:  * Nota: El PIC16F887 funciona por defecto a 31.25kHz (osc. interno)
9:  *
10:  * Adrián Bulnes Parra
11:  *=====*/
12:
13:  #include <pic16f88.h>
14:
15:  typedef unsigned int config;
16:  config at _CONFIG1 __CONFIG = _CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _
CPD_OFF & _LVP_OFF & _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO;
17:
18:  //Retraso simple durante un tiempo indeterminado
19:  void Delay(){
20:      long i;
21:
22:      for (i=0; i<200; i++) {
23:          //no hacer nada
24:      }
25:  }
26:
27:  void main(){
28:      //configura todos los pines como digitales, no analógicos
29:      ANSEL=0;
30:
31:      //configura todos los pines digitales como salidas
32:      TRISA=0;
33:      TRISB=0;
34:
35:      while(1){
36:          //encender LED
37:          PORTA=0b11111111;
38:          PORTB=0b11111111;
39:          Delay();
40:
41:          //apagar LED
42:          PORTA=0;
43:          PORTB=0;
44:          Delay();
45:      }
46:  }
```


Anexo IX

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo para LTIND-A y SISW-SPST usando MBP18*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Programa de ejemplo para LTIND-A y SISW-SPST usando MBP18
3:  *
4:  * Conecta SISW-SPST a RB3 (pin 9) y LTIND-A a RB4 (pin 10).
5:  * El LED se encenderá mientras el botón está pulsado.
6:  *
7:  * Adrián Bulnes Parra
8:  *===== */
9:
10: #include <pic16f88.h>
11:
12: typedef unsigned int config;
13: config at _CONFIG1 __CONFIG = _CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _
CPD_OFF & _LVP_OFF & _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO;
14:
15: #define Button RB3
16: #define Pressed 0
17: #define Released 1
18:
19: #define LED RB4
20: #define LedON 1
21: #define LedOFF 0
22:
23: void main(){
24:     //configura RB3 como entrada digital y RB4 como salida
25:     TRISB3=1;
26:     TRISB4=0;
27:
28:     while(1){
29:         if(Button==Pressed) LED=LedON;
30:         else LED=LedOFF;
31:     }
32: }
33:
34:
```

Anexo X

Tipo: Código fuente (programa para microcontrolador)

Programa: *Ejemplo para LTIND-A y SIBW-1Y usando MBP18*

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Ejemplo para LTIND-A y SIBW-1Y usando MBP18
3:  *
4:  * Conecta SIBW-1Y a RB3 (pin 9) y LTIND-A a RB4 (pin 10).
5:  * El LED se encenderá cuando hay un objeto blanco o reflectivo
6:  * próximo al sensor (aprox. menos de 1cm)
7:  *
8:  * Adrián Bulnes Parra
9:  *=====*/
10:
11: #include <MBP18.h>
12:
13: ConfigBits1(_CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF &
& _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO);
14:
15: #define Sensor RB3
16: #define White 0
17: #define Black 1
18:
19: #define LED RB4
20: #define LedON 1
21: #define LedOFF 0
22:
23: void main() {
24:     TRISB3=DigitalInput;
25:     TRISB4=DigitalOutput;
26:
27:     while(1){
28:         if(Sensor==White) LED=LedON;
29:         else LED=LedOFF;
30:     }
31: }
32:
33:
```

Anexo XI

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para AO-SPK usando MBP8

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  *  Ejemplo para AO-SPK usando MBP8
3:  *
4:  *  Conecta AO-SPK a GP5 (pin 2). Sonará un pitido.
5:  *
6:  *  Adrián Bulnes Parra
7:  *=====*/
8:
9:  #define OSC_8MHz
10:
11:  #include <MBP8.h>
12:  #include <Delays.h>
13:
14:  ConfigBits(_CPD_OFF & _CP_OFF & _BOD_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT
_OFF & _INTOSCIO);
15:
16:  #define Spk GP5
17:
18:  void Setup() {
19:      AdcDisable();
20:      SetDigitalOutput(5);
21:      SetIntosc8MHz();
22:  }
23:
24:
25:  void main() {
26:      Setup();
27:
28:      while(1) {
29:          Spk=!Spk;
30:          Delay(1);
31:      }
32:  }
33:
34:
35:
```

Anexo XII

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para AO-SPK usando MBP8 con temporizador

Lenguaje: C

Archivos de código: 1

```

1:  /*=====
2:  *  Ejemplo para AO-SPK usando MBP8 con temporizador
3:  *
4:  *  Conecta AO-SPK a GP5 (pin 2). Sonará un pitido.
5:  *
6:  *  Adrián Bulnes Parra
7:  *=====*/
8:
9:  #include <MBP8.h>
10:
11:  ConfigBits(_CPD_OFF & _CP_OFF & _BOD_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT_
_OFF & _INTOSCIO);
12:
13:  #define Spk GP5
14:
15:  Interrupt {
16:      Spk=!Spk;
17:      TMR0=114;
18:      T0IF=0;
19:  }
20:
21:  void Setup() {
22:      AdcDisable();
23:      SetDigitalOutput(5);
24:
25:      //configurar temporizador
26:      //PIC12F683 funciona por defecto a 4MHz
27:      //Fosc=4MHz, Tosc=250ns, Tcy=1us
28:      //para un tono de 440Hz necesitamor timer a 880Hz, osea 1136 inst
r/ciclo
29:      //...1:8 prescaler (PS=0b010). 142 cuentas de TMR0 por ciclo. pre
carga=114
30:      PSA=0; //asignar prescaler Timer0
31:      T0CS=0; //origen reloj=oscilador interno
32:      PS2=0;
33:      //prescaler...
34:      PS1=1;
35:      PS0=0;
36:      T0IF=0;
37:      GIE=1; //activar interrupciones globales
38:      T0IE=1; //activar int. TMR0
39:  }
40:
41:
42:  void main() {
43:      Setup();
44:
45:      while(1) {
46:          //no hacer nada
47:      }
48:  }
49:

```


Anexo XIII

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para AO-SPK usando MBP18

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  *  Ejemplo para AO-SPK usando MBP18
3:  *
4:  *  Conecta AO-SPK a RA1 (pin 18). Sonará un pitido.
5:  *
6:  *  Adrián Bulnes Parra
7:  *=====*/
8:
9:  #define OSC_8MHz
10:
11:  #include <MBP18.h>
12:  #include <Delays.h>
13:
14:  ConfigBits1(_CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF &
& _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO);
15:
16:  #define Spk RA1
17:
18:  void Setup(){
19:      ANSEL=0;
20:
21:      TRISA1=DigitalOutput;
22:
23:      SetIntosc8MHz();
24:  }
25:
26:  void main(){
27:      Setup();
28:
29:      while(1){
30:          Spk=!Spk;
31:          Delay(1);
32:      }
33:  }
34:
```

Anexo XIV

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para SIDST-GP2 usando MBP18

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  *  Ejemplo para SIDST-GP2 usando MBP18
3:  *
4:  *  Conecta SIDST-GP2 a RA0/AN0 (pin 17) y LTIND-A a RB4 (pin 10).
5:  *  El LED se encenderá cuando hay un objeto a menos de 25cm.
6:  *
7:  *  Adrián Bulnes Parra
8:  *===== */
9:
10: #include <MBP18.h>
11: #include <GP2.h>
12:
13: ConfigBits1(_CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF &
& _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO);
14:
15: #define LED RB4
16: #define LedON 1
17: #define LedOFF 0
18:
19: void main() {
20:     TRISB4=0;
21:     EnableAN0();
22:
23:     while(1) {
24:         //nota: distancia y valones del ADC son
25:         //inversamente proporcionales, así que la
26:         //comparación mayor-menor se hace al revés
27:         if(AdcMeasure() > GP2_25cm) LED=LedON;
28:         else LED=LedOFF;
29:     }
30: }
```

Anexo XV

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para LT7S-HN usando MBP40

Lenguaje: C

Archivos de código: 1

```

1:  /*=====
2:  *  Ejemplo para LT7S-HN usando MBP40
3:  *
4:  *  Este programa utiliza un LT7S-HN con 6 displays y 7 segmentos sin punto
5:  *  además de un SISW-SPST. Todos los displays y segmentos están encendido
6:  *  mientras el botón NO está pulsado, y se apagan al pulsarlo.
7:  *
8:  *  Adrián Bulnes Parra
9:  *=====*/
10:
11: #include <MBP40.h>
12:
13: ConfigBits1(_CP_OFF & _DEBUG_OFF & _FCMEN_OFF & _IESO_OFF & _CPD_OFF & _
LVP_OFF & _BOR_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT_OFF & _INTOSCIO);
14:
15: //displays
16: #define D1 RE2
17: #define D2 RE1
18: #define D3 RE0
19: #define D4 RA5
20: #define D5 RA4
21: #define D6 RA3
22:
23: //segmentos
24: #define SA RD1
25: #define SB RD0
26: #define SC RC3
27: #define SD RC2
28: #define SE RC1
29: #define SF RC0
30: #define SG RA6
31:
32: #define ON 1
33: #define OFF 0
34:
35: #define Button RB0
36: #define Pressed 0
37: #define Released 1
38:
39: void main() {
40:     AdcDisable();
41:
42:     TRISE2=DigitalOutput;
43:     TRISE1=DigitalOutput;
44:     TRISE0=DigitalOutput;
45:     TRISA5=DigitalOutput;
46:     TRISA4=DigitalOutput;
47:     TRISA3=DigitalOutput;
48:     TRISD1=DigitalOutput;
49:     TRISD0=DigitalOutput;
50:     TRISC3=DigitalOutput;
51:     TRISC2=DigitalOutput;
52:     TRISC1=DigitalOutput;
53:     TRISC0=DigitalOutput;
54:     TRISA6=DigitalOutput;
55:
56:     TRISB0=DigitalInput;
57:
58:     while(1) {

```

```
59:             if(Button==Released){
60:                 D1=ON;
61:                 D2=ON;
62:                 D3=ON;
63:                 D4=ON;
64:                 D5=ON;
65:                 D6=ON;
66:                 SA=ON;
67:                 SB=ON;
68:                 SC=ON;
69:                 SD=ON;
70:                 SE=ON;
71:                 SF=ON;
72:                 SG=ON;
73:             } else {
74:                 D1=OFF;
75:                 D2=OFF;
76:                 D3=OFF;
77:                 D4=OFF;
78:                 D5=OFF;
79:                 D6=OFF;
80:                 SA=OFF;
81:                 SB=OFF;
82:                 SC=OFF;
83:                 SD=OFF;
84:                 SE=OFF;
85:                 SF=OFF;
86:                 SG=OFF;
87:             }
88:         }
89: }
```

Anexo XVI

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para CMIR-RC y LTIND-A usando MBP18

Lenguaje: C

Archivos de código: 1


```
1:  /*=====
2:  * Ejemplo para CMIR-RC y LTIND-A usando MBP18
3:  *
4:  * Conecta CMIR-RC a RB3 (pin 9) y LTIND-A a RB4 (pin 10).
5:  * El LED parpadeará mientras se pulsa cualquier botón desde un
6:  * control remoto de televisión o similar.
7:  *
8:  * Adrián Bulnes Parra
9:  *=====*/
10:
11: #include <MBP18.h>
12:
13: ConfigBits1(_CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF &
& _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO);
14:
15: #define IrReceiver RB3
16:
17: #define LED RB4
18: #define LedON 1
19: #define LedOFF 0
20:
21: void main(){
22:     //configura RB3 como entrada digital y RB4 como salida
23:     TRISB3=DigitalInput;
24:     TRISB4=DigitalOutput;
25:
26:     while(1){
27:         if(IrReceiver==0) LED=LedON;
28:         else LED=LedOFF;
29:     }
30: }
```

Anexo XVII

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para MC2A o MC2B usando MBP18

Lenguaje: C

Archivos de código: 1

```
1:  /*=====
2:  * Ejemplo para MC2A usando MBP18
3:  *
4:  * El Motor 1 se mueve hacia delante, después hacia atrás, después
5:  * se detiene y entonces el Motor comienza a moverse hacia delante,
6:  * atrás, se detiene y se repite todo el proceso.
7:  *
8:  * Nota: PIC16F88 funciona por defecto a 31.25kHz (oscilador interno)
9:  *
10:  * Adrián Bulnes Parra
11:  *=====*/
12:
13:  #include <pic16f88.h>
14:
15:  typedef unsigned int config;
16:  config at _CONFIG1 __CONFIG = _CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _
CPD_OFF & _LVP_OFF & _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO;
17:
18:  //pines (sin configurar aún)
19:  #define M1Enable Rxx
20:  #define M1In1 Rxx
21:  #define M1In2 Rxx
22:  #define M2Enable Rxx
23:  #define M2In1 Rxx
24:  #define M2In2 Rxx
25:
26:  //Mover motor 1 hacia delante
27:  void M1Fwd(){
28:      M1Enable=1;
29:      M1In1=1;
30:      M1In2=0;
31:  }
32:
33:  //Mover motor 1 hacia atrás
34:  void M1Bck(){
35:      M1Enable=1;
36:      M1In1=0;
37:      M1In2=1;
38:  }
39:
40:  //Frenar motor 1
41:  void M1Brake(){
42:      M1Enable=1;
43:      M1In1=0;
44:      M1In2=0;
45:  }
46:
47:  //Liberar motor 1
48:  void M1Free(){
49:      M1Enable=0;
50:  }
51:
52:  //Mover motor 2 hacia delante
53:  void M2Fwd(){
54:      M2Enable=1;
55:      M2In1=1;
56:      M2In2=0;
57:  }
58:
59:  //Mover motor 2 hacia atrás
60:  void M2Bck(){
```

```
61:         M2Enable=1;
62:         M2In1=0;
63:         M2In2=1;
64:     }
65:
66:     //Frenar motor 2
67:     void M2Brake(){
68:         M2Enable=1;
69:         M2In1=0;
70:         M2In2=0;
71:     }
72:
73:     //Liberar motor 2
74:     void M2Free(){
75:         M2Enable=0;
76:     }
77:
78:     //Retraso simple durante un tiempo indeterminado
79:     void Delay(){
80:         long i;
81:
82:         for (i=0; i<200; i++) {
83:             //no hacer nada
84:         }
85:     }
86:
87:     void main(){
88:         //configurar pines como salidas digitales
89:         ANSEL=0;
90:         TRISA=(CAMBIAR ESTO SEGUN CONFIGURACION);
91:         TRISB=(CAMBIAR ESTO SEGUN CONFIGURACION);
92:
93:         while(1){
94:             M1Fwd();
95:             Delay();
96:             M1Bck();
97:             Delay();
98:             M1Brake();
99:
100:            M2Fwd();
101:            Delay();
102:            M2Bck();
103:            Delay();
104:            M2Brake();
105:
106:        }
107:    }
```

Anexo XVIII

Tipo: Código fuente (programa para microcontrolador)

**Programa: Ejemplo para CMSP-MAX (sólo transmisión)
usando MBP40**

Lenguaje: C

Archivos de código: 1

```

1:  /*=====
2:  *  Ejemplo para CMSP-MAX (sólo transmisión) usando MBP40
3:  *
4:  *  Envía números consecutivos a través del puerto serie.
5:  *  Ésta es la manera más simple de comprobar que se pueden transmitir
6:  *  datos desde el microcontrolador hasta el ordenador.
7:  *
8:  *  Conecta MB_TX de CMSP-MAX a TX/CK (pin 25) de MBP40.
9:  *
10: *  Adrián Bulnes Parra
11: *===== */
12:
13: #include <MBP40.h>
14:
15: #define OSC_8MHz
16: #include <Delays.h>
17:
18: ConfigBits1(_CP_OFF & _DEBUG_OFF & _FCMEN_OFF & _IESO_OFF & _CPD_OFF & _
LVP_OFF & _BOR_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT_OFF & _INTOSCIO);
19:
20: //Envía un byte a través del puerto serie
21: void WriteSP(unsigned int8 value){
22:     //espera a que termine la transmisión anterior
23:     while(!TXIF){
24:         //no hacer nada
25:     }
26:     TXREG=value;
27:     Delay10us();
28: }
29:
30: void main() {
31:     unsigned int8 i=13;
32:
33:     SetIntosc8MHz();
34:     AdcDisable();
35:
36:     //configura UART para transmisión
37:     //Baud rate timer: SPBRG=((Fosc/(16 * 9600) -1))
38:     //Baud rate = Fosc/(16*(SPBRG+1))
39:     //baud rate real para Fosc=8MHz: 9615.384615 bauds
40:     // SPRG=51
41:     SPBRGH=0;
42:     SPBRG=51;
43:     BRGH=1; //alta velocidad
44:     TXEN=1; // (registro TXSTA) activa transmisor
45:     SYNC=0; // (registro TXSTA) operación asíncrona
46:     SPEN=1; // (registro RCSTA) configura el pin TX/CK como salida
47:     //...por defecto: 8 bits, sin interrupciones
48:
49:     WriteSP('h');
50:     WriteSP('o');
51:     WriteSP('l');
52:     WriteSP('a');
53:
54:     while(1){
55:         WriteSP(i);
56:         i++;
57:     }
58: }
59:

```

Anexo XIX

Tipo: Código fuente (programa para microcontrolador)

Programa: Ejemplo para CMSP-MAX (recepción y transmisión) usando MBP40

Lenguaje: C

Archivos de código: 1

```

1:  /*=====
2:  *  Ejemplo para CMSP-MAX usando MBP40
3:  *
4:  *  Lee un byte desde el puerto serie y envía de vuelta 3 bytes:
5:  *  1- El mismo byte que se recibió
6:  *  2- El byte recibido más uno
7:  *  3- El byte recibido más dos
8:  *  Por ejemplo, si se envía el byte "57" desde el ordenador, se
9:  *  enviará de vuelta al ordenador "57", "58" y "59".
10: *
11: *  Conecta MB_TX de CMSP-MAX a TX/CK (pin 25) en MBP40
12: *  Conecta MB_RX de CMSP-MAX a RX/DT (pin 26) en MBP40
13: *
14: *  Adrián Bulnes Parra
15: *===== */
16:
17: #include <MBP40.h>
18:
19: #define OSC_8MHz
20: #include <Delays.h>
21:
22: ConfigBits1(_CP_OFF & _DEBUG_OFF & _FCMEN_OFF & _IESO_OFF & _CPD_OFF & _
LVP_OFF & _BOR_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT_OFF & _INTOSCIO);
23:
24: //Envía un byte a través del puerto serie
25: void WriteSP(unsigned int8 value) {
26:     //espera a que termine la transmisión anterior
27:     while(!TXIF) {
28:         //no hacer nada
29:     }
30:     TXREG=value;
31:     Delay10us();
32: }
33:
34: //Lee un byte desde el puerto serie. Si no hay datos en el buffer de
35: //recepción, espera indefinidamente hasta el próximo byte recibido
36: unsigned int8 ReadSPWait(){
37:     unsigned int8 dummy;
38:     while(!RCIF) {
39:         if (OERR) {
40:             TXEN=0;
41:             TXEN=1;
42:             CREN=0;
43:             CREN=1;
44:         }
45:         if (FERR) {
46:             dummy=RCREG;
47:             TXEN=0;
48:             TXEN=1;
49:         }
50:     }
51:     return RCREG;
52: }
53:
54: //Lee un byte desde el puerto serie. Si no hay datos en el buffer de
55: //recepción, devuelve 0xFF
56: unsigned int8 ReadSP() {
57:     if(RCIF) return RCREG;
58:     else return 0xFF;
59: }
60:

```



```
61: void main() {
62:     unsigned int8 i=0;
63:
64:     SetIntosc8MHz();
65:     AdcDisable();
66:
67:     //configurar UART
68:     //Baud rate timer: SPBRG=((Fosc/(16 * 9600) -1))
69:     //Baud rate = Fosc/(16*(SPBRG+1))
70:     //baud rate real para Fosc=8MHz: 9615.384615 bauds
71:     // SPRG=51,
72:     SPBRGH=0;
73:     SPBRG=51;
74:     BRGH=1; //alta velocidad
75:     CREN=1; //activar receptor
76:     TXEN=1; // (registro TXSTA) activar transmisor
77:     SYNC=0; // (registro TXSTA) operación asíncrona
78:     SPEN=1; // (registro RCSTA) config TX/CK como salida, RX/DT entrada
da
79:     //...por defecto: 8 bits, sin interrupciones
80:
81:     WriteSP('\n');
82:     WriteSP('i');
83:     WriteSP('n');
84:     WriteSP('i');
85:     WriteSP('t');
86:
87:     while(1) {
88:         i=ReadSP();
89:         if(i!=0xFF) {
90:             WriteSP(i);
91:             i++;
92:             WriteSP(i);
93:             i++;
94:             WriteSP(i);
95:         }
96:     }
97: }
```

Anexo XX

Tipo: Código fuente (programa para ordenador)

Programa: Entorno de desarrollo

Lenguaje: C#

Archivos de código: 42

```
1: using System;
2:
3: namespace CuruxaIDE {
4:     public struct CursorLocation {
5:         public int Line;
6:         public int Column;
7:
8:         public CursorLocation(int Line, int Column) {
9:             this.Line = Line;
10:            this.Column = Column;
11:        }
12:    }
13: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.IO;
6: using System.Xml.Serialization;
7:
8: namespace CuruxaIDE {
9:     /// <summary>
10:    /// Representa un proyecto de ejemplo
11:    /// </summary>
12:    [Serializable]
13:    public class Example {
14:        public const string ExampleExtension = "CuruxaExample";
15:
16:        /// <summary>
17:        /// Proyecto de Curuxa IDE que contiene a este ejemplo
18:        /// </summary>
19:        public Project Project = new Project();
20:
21:        /// <summary>
22:        /// Lista de Módulos de Curuxa usados en este Ejemplo
23:        /// </summary>
24:        public List<Module> UsedModules = new List<Module>();
25:
26:        /// <summary>
27:        /// Guardar Ejemplo
28:        /// </summary>
29:        public void Save(string ExFile) {
30:            XmlSerializer xs = new XmlSerializer(this.GetType());
31:            StreamWriter sw = new StreamWriter(ExFile);
32:            xs.Serialize(sw, this);
33:            sw.Close();
34:        }
35:
36:        #region static members
37:        public static Example[] AvailableExamples {
38:            get {
39:                if(_AvailableExamples == null) {
```

```
40:         DirectoryInfo di = new DirectoryInfo(Settings.ExamplesDir);
41:         FileInfo[] files = di.GetFiles("*. " + ExampleExtension);
42:         Example[] examples = new Example[files.Length];
43:         for(int i = 0; i < examples.Length; i++) {
44:             examples[i] = Load(files[i].Name.Replace(".", " " + ExampleExtension, "
"));
45:         }
46:         _AvailableExamples = examples;
47:     }
48:     return _AvailableExamples;
49: }
50: }
51: protected static Example[] _AvailableExamples;
52:
53: /// <summary>
54: /// Cargar ejemplo desde archivo
55: /// </summary>
56: /// <param name="PrjFile">Path to the example file</param>
57: /// <returns>Example object if found. Null if it doesn't exist</returns>
58: public static Example Load(string Name) {
59:     string file = string.Format("{0}/{1}.{2}", Settings.ExamplesDir, Name, ExampleExtension);
60:     Example NewEx = null;
61:     FileInfo fi = new FileInfo(file);
62:     if(fi.Exists) {
63:         Globals.Debug("Loading example file: " + Name);
64:         XmlSerializer xs = new XmlSerializer(typeof(Example));
65:         NewEx = (Example)xs.Deserialize(new StreamReader(file));
66:         NewEx.Project.Path = Settings.ExamplesDir;
67:     } else Globals.LogIDE(i18n.str("ExFileNotExist", file));
68:     return NewEx;
69: }
70: #endregion
71: }
72: }
```

```
1: // Archivo generado automáticamente por Visual Studio
2:
3: namespace CuruxaIDE {
4:     partial class FrmAbout {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        protected override void Dispose(bool disposing) {
14:            if(disposing && (components != null)) {
15:                components.Dispose();
16:            }
17:            base.Dispose(disposing);
18:        }
19:
20:        #region Windows Form Designer generated code
21:
22:        /// <summary>
23:        /// Required method for Designer support - do not modify
24:        /// the contents of this method with the code editor.
25:        /// </summary>
26:        private void InitializeComponent() {
27:            this.tableLayoutPanel = new System.Windows.Forms.TableLayoutPanel();
28:            this.Logo = new System.Windows.Forms.PictureBox();
29:            this.labelProductName = new System.Windows.Forms.Label();
30:            this.LblVersion = new System.Windows.Forms.Label();
31:            this.TxtDescription = new System.Windows.Forms.TextBox();
32:            this.okButton = new System.Windows.Forms.Button();
33:            this.LblLicense = new System.Windows.Forms.Label();
34:            this.linkLabel1 = new System.Windows.Forms.LinkLabel();
35:            this.tableLayoutPanel.SuspendLayout();
36:            ((System.ComponentModel.ISupportInitialize)(this.Logo)).BeginInit();
37:            this.SuspendLayout();
38:            //
39:            // tableLayoutPanel
```

```
40:          //
41:          this.tableLayoutPanel.ColumnCount = 2;
42:          this.tableLayoutPanel.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows
.Forms.SizeType.Percent, 46.81275F));
43:          this.tableLayoutPanel.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows
.Forms.SizeType.Percent, 53.18725F));
44:          this.tableLayoutPanel.Controls.Add(this.Logo, 0, 0);
45:          this.tableLayoutPanel.Controls.Add(this.labelProductName, 1, 0);
46:          this.tableLayoutPanel.Controls.Add(this.LblVersion, 1, 1);
47:          this.tableLayoutPanel.Controls.Add(this.TxtDescription, 1, 4);
48:          this.tableLayoutPanel.Controls.Add(this.okButton, 1, 5);
49:          this.tableLayoutPanel.Controls.Add(this.LblLicense, 1, 2);
50:          this.tableLayoutPanel.Controls.Add(this.linkLabel1, 1, 3);
51:          this.tableLayoutPanel.Dock = System.Windows.Forms.DockStyle.Fill;
52:          this.tableLayoutPanel.Location = new System.Drawing.Point(9, 9);
53:          this.tableLayoutPanel.Name = "tableLayoutPanel";
54:          this.tableLayoutPanel.RowCount = 6;
55:          this.tableLayoutPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms
.SizeType.Percent, 10F));
56:          this.tableLayoutPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms
.SizeType.Percent, 10F));
57:          this.tableLayoutPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms
.SizeType.Percent, 10F));
58:          this.tableLayoutPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms
.SizeType.Percent, 10F));
59:          this.tableLayoutPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms
.SizeType.Percent, 50F));
60:          this.tableLayoutPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms
.SizeType.Percent, 10F));
61:          this.tableLayoutPanel.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms
.SizeType.Absolute, 20F));
62:          this.tableLayoutPanel.Size = new System.Drawing.Size(502, 265);
63:          this.tableLayoutPanel.TabIndex = 0;
64:          //
65:          // Logo
66:          //
67:          this.Logo.Dock = System.Windows.Forms.DockStyle.Fill;
68:          this.Logo.Location = new System.Drawing.Point(3, 3);
69:          this.Logo.Name = "Logo";
```

```
70:         this.tableLayoutPanel.SetRowSpan(this.Logo, 6);
71:         this.Logo.Size = new System.Drawing.Size(228, 259);
72:         this.Logo.SizeMode = System.Windows.Forms.PictureBoxSizeMode.Zoom;
73:         this.Logo.TabIndex = 12;
74:         this.Logo.TabStop = false;
75:         //
76:         // labelProductName
77:         //
78:         this.labelProductName.Dock = System.Windows.Forms.DockStyle.Fill;
79:         this.labelProductName.Location = new System.Drawing.Point(240, 0);
80:         this.labelProductName.Margin = new System.Windows.Forms.Padding(6, 0, 3, 0);
81:         this.labelProductName.MaximumSize = new System.Drawing.Size(0, 17);
82:         this.labelProductName.Name = "labelProductName";
83:         this.labelProductName.Size = new System.Drawing.Size(259, 17);
84:         this.labelProductName.TabIndex = 19;
85:         this.labelProductName.Text = "Curuxa IDE";
86:         this.labelProductName.TextAlign = System.Drawing.ContentAlignment.MiddleLeft;
87:         //
88:         // LblVersion
89:         //
90:         this.LblVersion.Dock = System.Windows.Forms.DockStyle.Fill;
91:         this.LblVersion.Location = new System.Drawing.Point(240, 26);
92:         this.LblVersion.Margin = new System.Windows.Forms.Padding(6, 0, 3, 0);
93:         this.LblVersion.MaximumSize = new System.Drawing.Size(0, 17);
94:         this.LblVersion.Name = "LblVersion";
95:         this.LblVersion.Size = new System.Drawing.Size(259, 17);
96:         this.LblVersion.TabIndex = 0;
97:         this.LblVersion.Text = "NS (Version)";
98:         this.LblVersion.TextAlign = System.Drawing.ContentAlignment.MiddleLeft;
99:         //
100:        // TxtDescription
101:        //
102:        this.TxtDescription.Dock = System.Windows.Forms.DockStyle.Fill;
103:        this.TxtDescription.Location = new System.Drawing.Point(240, 107);
104:        this.TxtDescription.Margin = new System.Windows.Forms.Padding(6, 3, 3, 3);
105:        this.TxtDescription.Multiline = true;
106:        this.TxtDescription.Name = "TxtDescription";
107:        this.TxtDescription.ReadOnly = true;
108:        this.TxtDescription.ScrollBars = System.Windows.Forms.ScrollBars.Both;
```



```
109:         this.TxtDescription.Size = new System.Drawing.Size(259, 126);
110:         this.TxtDescription.TabIndex = 23;
111:         this.TxtDescription.TabStop = false;
112:         this.TxtDescription.Text = "NS (Description)";
113:         //
114:         // okButton
115:         //
116:         this.okButton.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStylesBottom | System.Windows.Forms.AnchorStylesRight)));
117:         this.okButton.DialogResult = System.Windows.Forms.DialogResult.Cancel;
118:         this.okButton.Location = new System.Drawing.Point(424, 239);
119:         this.okButton.Name = "okButton";
120:         this.okButton.Size = new System.Drawing.Size(75, 23);
121:         this.okButton.TabIndex = 24;
122:         this.okButton.Text = "&OK";
123:         this.okButton.Click += new System.EventHandler(this.okButton_Click);
124:         //
125:         // LblLicense
126:         //
127:         this.LblLicense.Dock = System.Windows.Forms.DockStyle.Fill;
128:         this.LblLicense.Location = new System.Drawing.Point(240, 52);
129:         this.LblLicense.Margin = new System.Windows.Forms.Padding(6, 0, 3, 0);
130:         this.LblLicense.MaximumSize = new System.Drawing.Size(0, 17);
131:         this.LblLicense.Name = "LblLicense";
132:         this.LblLicense.Size = new System.Drawing.Size(259, 17);
133:         this.LblLicense.TabIndex = 21;
134:         this.LblLicense.Text = "GNU General Public License v3.0";
135:         this.LblLicense.TextAlign = System.Drawing.ContentAlignment.MiddleLeft;
136:         //
137:         // linkLabel1
138:         //
139:         this.linkLabel1.AutoSize = true;
140:         this.linkLabel1.Location = new System.Drawing.Point(237, 78);
141:         this.linkLabel1.Name = "linkLabel1";
142:         this.linkLabel1.Size = new System.Drawing.Size(88, 13);
143:         this.linkLabel1.TabIndex = 25;
144:         this.linkLabel1.TabStop = true;
145:         this.linkLabel1.Text = "http://curuxa.org";
146:         this.linkLabel1.LinkClicked += new System.Windows.Forms.LinkLabelLinkClickedEventHandler(t
```

```
this.linkLabel1_LinkClicked);
147:                                     //
148:                                     // FrmAbout
149:                                     //
150:                                     this.AcceptButton = this.okButton;
151:                                     this.AutoScaleMode = new System.Drawing.SizeF(6F, 13F);
152:                                     this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
153:                                     this.ClientSize = new System.Drawing.Size(520, 283);
154:                                     this.Controls.Add(this.tableLayoutPanel);
155:                                     this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;
156:                                     this.MaximizeBox = false;
157:                                     this.MinimizeBox = false;
158:                                     this.Name = "FrmAbout";
159:                                     this.Padding = new System.Windows.Forms.Padding(9);
160:                                     this.ShowIcon = false;
161:                                     this.ShowInTaskbar = false;
162:                                     this.StartPosition = System.Windows.Forms.FormStartPosition.CenterParent;
163:                                     this.Text = "NS (About Curuxa IDE)";
164:                                     this.Load += new System.EventHandler(this.FrmAbout_Load);
165:                                     this.tableLayoutPanel.ResumeLayout(false);
166:                                     this.tableLayoutPanel.PerformLayout();
167:                                     ((System.ComponentModel.ISupportInitialize)(this.Logo)).EndInit();
168:                                     this.ResumeLayout(false);
169:
170:                                 }
171:
172:                             #endregion
173:
174:                             private System.Windows.Forms.TableLayoutPanel tableLayoutPanel;
175:                             private System.Windows.Forms.PictureBox Logo;
176:                             private System.Windows.Forms.Label labelProductName;
177:                             private System.Windows.Forms.Label lblVersion;
178:                             private System.Windows.Forms.Label lblLicense;
179:                             private System.Windows.Forms.TextBox TxtDescription;
180:                             private System.Windows.Forms.Button okButton;
181:                             private System.Windows.Forms.LinkLabel linkLabel1;
182:
183:     }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Drawing;
5: using System.Linq;
6: using System.Reflection;
7: using System.Windows.Forms;
8:
9: namespace CuruxaIDE {
10:     partial class FrmAbout:Form {
11:         public FrmAbout() {
12:             InitializeComponent();
13:         }
14:
15:         private void okButton_Click(object sender, EventArgs e) {
16:             this.Close();
17:         }
18:
19:         private void linkLabel1_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e) {
20:             System.Diagnostics.Process.Start("http://curuxa.org/");
21:         }
22:
23:         private void FrmAbout_Load(object sender, EventArgs e) {
24:             UpdateLangStrings();
25:             Logo.ImageLocation = Settings.ImagesDir + "/CuruxaLogo.png";
26:         }
27:
28:         private void UpdateLangStrings() {
29:             this.Text = i18n.str("About");
30:             LblVersion.Text = i18n.str("Version:", Settings.Version);
31:             TxtDescription.Text = i18n.str("CuruxaDescription").Replace("\\n", Environment.NewLine);
32:         }
33:     }
34: }
```

```
1: // Archivo generado automáticamente por Visual Studio
2:
3: namespace CuruxaIDE {
4:     partial class FrmCrash {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
>
14:        protected override void Dispose(bool disposing) {
15:            if(disposing && (components != null)) {
16:                components.Dispose();
17:            }
18:            base.Dispose(disposing);
19:        }
20:
21:        #region Windows Form Designer generated code
22:
23:        /// <summary>
24:        /// Required method for Designer support - do not modify
25:        /// the contents of this method with the code editor.
26:        /// </summary>
27:        private void InitializeComponent() {
28:            this.TxtError = new System.Windows.Forms.TextBox();
29:            this.BtnClose = new System.Windows.Forms.Button();
30:            this.SuspendLayout();
31:            //
32:            // TxtError
33:            //
34:            this.TxtError.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles
Styles.Top | System.Windows.Forms.AnchorStyles.Bottom)
35:                | System.Windows.Forms.AnchorStyles.Left)
36:                | System.Windows.Forms.AnchorStyles.Right)));
37:            this.TxtError.BackColor = System.Drawing.SystemColors.Window;
```

```
38:         this.TxtError.Location = new System.Drawing.Point(12, 12);
39:         this.TxtError.Multiline = true;
40:         this.TxtError.Name = "TxtError";
41:         this.TxtError.ReadOnly = true;
42:         this.TxtError.ScrollBars = System.Windows.Forms.ScrollBars.Both;
43:         this.TxtError.Size = new System.Drawing.Size(742, 220);
44:         this.TxtError.TabIndex = 0;
45:         //
46:         // BtnClose
47:         //
48:         this.BtnClose.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStylesBottom | System.Windows.Forms.AnchorStylesRight)));
49:         this.BtnClose.Location = new System.Drawing.Point(678, 238);
50:         this.BtnClose.Name = "BtnClose";
51:         this.BtnClose.Size = new System.Drawing.Size(75, 23);
52:         this.BtnClose.TabIndex = 1;
53:         this.BtnClose.Text = "Close";
54:         this.BtnClose.UseVisualStyleBackColor = true;
55:         this.BtnClose.Click += new System.EventHandler(this.BtnClose_Click);
56:         //
57:         // FrmCrash
58:         //
59:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
60:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
61:         this.ClientSize = new System.Drawing.Size(766, 273);
62:         this.Controls.Add(this.BtnClose);
63:         this.Controls.Add(this.TxtError);
64:         this.Name = "FrmCrash";
65:         this.Text = "Application crash";
66:         this.Load += new System.EventHandler(this.FrmCrash_Load);
67:         this.ResumeLayout(false);
68:         this.PerformLayout();
69:
70:     }
71:
72: #endregion
73:
74: private System.Windows.Forms.TextBox TxtError;
75: private System.Windows.Forms.Button BtnClose;
```

```
76:      }  
77: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Data;
5: using System.Drawing;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9:
10: namespace CuruxaIDE {
11:     public partial class FrmCrash:Form {
12:         public readonly object Error;
13:
14:         public FrmCrash(object Error) {
15:             this.Error = Error;
16:             InitializeComponent();
17:         }
18:
19:         private void FrmCrash_Load(object sender, EventArgs e) {
20:             TxtError.Clear();
21:             TxtError.AppendText(Error.ToString());
22:         }
23:
24:         private void BtnClose_Click(object sender, EventArgs e) {
25:             Environment.Exit(1);
26:         }
27:     }
28: }
```

```
1: // Archivo generado automáticamente por Visual Studio
2:
3: namespace CuruxaIDE {
4:     partial class FrmFileName {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
>
14:        protected override void Dispose(bool disposing) {
15:            if(disposing && (components != null)) {
16:                components.Dispose();
17:            }
18:            base.Dispose(disposing);
19:        }
20:
21:        #region Windows Form Designer generated code
22:
23:        /// <summary>
24:        /// Required method for Designer support - do not modify
25:        /// the contents of this method with the code editor.
26:        /// </summary>
27:        private void InitializeComponent() {
28:            this.BtnOk = new System.Windows.Forms.Button();
29:            this.tableLayoutPanel1 = new System.Windows.Forms.TableLayoutPanel();
30:            this.LblFileName = new System.Windows.Forms.Label();
31:            this.TxtFileName = new System.Windows.Forms.TextBox();
32:            this.tableLayoutPanel1.SuspendLayout();
33:            this.SuspendLayout();
34:            //
35:            // BtnOk
36:            //
37:            this.BtnOk.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Right)));
```



```
38:         this.BtnOk.Location = new System.Drawing.Point(241, 50);
39:         this.BtnOk.Name = "BtnOk";
40:         this.BtnOk.Size = new System.Drawing.Size(75, 23);
41:         this.BtnOk.TabIndex = 1;
42:         this.BtnOk.Text = "NS (OK)";
43:         this.BtnOk.UseVisualStyleBackColor = true;
44:         this.BtnOk.Click += new System.EventHandler(this.BtnOk_Click);
45:         //
46:         // tableLayoutPanel1
47:         //
48:         this.tableLayoutPanel1.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Left | System.Windows.Forms.AnchorStyles.Right)));
49:         this.tableLayoutPanel1.ColumnCount = 2;
50:         this.tableLayoutPanel1.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle());
51:         this.tableLayoutPanel1.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 100F));
52:         this.tableLayoutPanel1.Controls.Add(this.LblFileName, 0, 0);
53:         this.tableLayoutPanel1.Controls.Add(this.TxtFileName, 1, 0);
54:         this.tableLayoutPanel1.Location = new System.Drawing.Point(13, 12);
55:         this.tableLayoutPanel1.Name = "tableLayoutPanel1";
56:         this.tableLayoutPanel1.RowCount = 1;
57:         this.tableLayoutPanel1.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 100F));
58:         this.tableLayoutPanel1.Size = new System.Drawing.Size(303, 25);
59:         this.tableLayoutPanel1.TabIndex = 2;
60:         //
61:         // LblFileName
62:         //
63:         this.LblFileName.AutoSize = true;
64:         this.LblFileName.Dock = System.Windows.Forms.DockStyle.Fill;
65:         this.LblFileName.Location = new System.Drawing.Point(3, 0);
66:         this.LblFileName.Name = "LblFileName";
67:         this.LblFileName.Size = new System.Drawing.Size(78, 25);
68:         this.LblFileName.TabIndex = 0;
69:         this.LblFileName.Text = "NS (File Name)";
70:         this.LblFileName.TextAlign = System.Drawing.ContentAlignment.MiddleRight;
71:         //
72:         // TxtFileName
73:
```

```
74:          //
75:          this.TxtFileName.Dock = System.Windows.Forms.DockStyle.Fill;
76:          this.TxtFileName.Location = new System.Drawing.Point(87, 3);
77:          this.TxtFileName.Name = "TxtFileName";
78:          this.TxtFileName.Size = new System.Drawing.Size(213, 20);
79:          this.TxtFileName.TabIndex = 1;
80:          //
81:          // FrmFileName
82:          //
83:          this.AcceptButton = this.BtnOk;
84:          this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
85:          this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
86:          this.ClientSize = new System.Drawing.Size(328, 85);
87:          this.Controls.Add(this.tableLayoutPanel1);
88:          this.Controls.Add(this.BtnOk);
89:          this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedToolWindow;
90:          this.Name = "FrmFileName";
91:          this.Text = "NS (File Name)";
92:          this.Load += new System.EventHandler(this.FrmFileName_Load);
93:          this.tableLayoutPanel1.ResumeLayout(false);
94:          this.tableLayoutPanel1.PerformLayout();
95:          this.ResumeLayout(false);
96:
97:      }
98:
99:      #endregion
100:
101:      private System.Windows.Forms.Button BtnOk;
102:      private System.Windows.Forms.TableLayoutPanel tableLayoutPanel1;
103:      private System.Windows.Forms.Label lblFileName;
104:      public System.Windows.Forms.TextBox TxtFileName;
105:
106:    }
107: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Data;
5: using System.Drawing;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9:
10: namespace CuruxaIDE {
11:     public partial class FrmFileName:Form {
12:         public FrmFileName(string FileName) {
13:             InitializeComponent();
14:             TxtFileName.Text = FileName;
15:         }
16:
17:         private void FrmFileName_Load(object sender, EventArgs e) {
18:             UpdateLang();
19:         }
20:
21:         private void UpdateLang() {
22:             this.Text = i18n.str("SaveMainFileAs");
23:             LblFileName.Text = i18n.str("FileName") + ":";
24:             BtnOk.Text = i18n.str("OK");
25:         }
26:
27:         private void BtnOk_Click(object sender, EventArgs e) {
28:             DialogResult = DialogResult.OK;
29:             this.Close();
30:         }
31:     }
32: }
```

```

1: // Archivo generado automáticamente por Visual Studio
2:
3: namespace CuruxaIDE {
4:     partial class FrmIdeSettings {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
>
14:        protected override void Dispose(bool disposing) {
15:            if(disposing && (components != null)) {
16:                components.Dispose();
17:            }
18:            base.Dispose(disposing);
19:        }
20:
21:        #region Windows Form Designer generated code
22:
23:        /// <summary>
24:        /// Required method for Designer support - do not modify
25:        /// the contents of this method with the code editor.
26:        /// </summary>
27:        private void InitializeComponent() {
28:            this.tableLayoutPanel1 = new System.Windows.Forms.TableLayoutPanel();
29:            this.lblLang = new System.Windows.Forms.Label();
30:            this.cmbLang = new System.Windows.Forms.ComboBox();
31:            this.BtnOk = new System.Windows.Forms.Button();
32:            this.BtnRestart = new System.Windows.Forms.Button();
33:            this.BtnCancel = new System.Windows.Forms.Button();
34:            this.cbEnbSynHgl = new System.Windows.Forms.CheckBox();
35:            this.tableLayoutPanel1.SuspendLayout();
36:            this.SuspendLayout();
37:            //
38:            // tableLayoutPanel1

```

```
39:          //
40:          this.tableLayoutPanel1.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Left)
s.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Left)
41:          | System.Windows.Forms.AnchorStyles.Right));
42:          this.tableLayoutPanel1.ColumnCount = 2;
43:          this.tableLayoutPanel1.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle());
44:          this.tableLayoutPanel1.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 100F));
s.Forms.SizeType.Percent, 100F));
45:          this.tableLayoutPanel1.Controls.Add(this.lblLang, 0, 0);
46:          this.tableLayoutPanel1.Controls.Add(this.cmbLang, 1, 0);
47:          this.tableLayoutPanel1.Location = new System.Drawing.Point(12, 26);
48:          this.tableLayoutPanel1.Name = "tableLayoutPanel1";
49:          this.tableLayoutPanel1.RowCount = 1;
50:          this.tableLayoutPanel1.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 100F));
s.SizeType.Percent, 100F));
51:          this.tableLayoutPanel1.Size = new System.Drawing.Size(325, 28);
52:          this.tableLayoutPanel1.TabIndex = 0;
53:          //
54:          // lblLang
55:          //
56:          this.lblLang.AutoSize = true;
57:          this.lblLang.Dock = System.Windows.Forms.DockStyle.Fill;
58:          this.lblLang.Location = new System.Drawing.Point(3, 0);
59:          this.lblLang.Name = "lblLang";
60:          this.lblLang.Size = new System.Drawing.Size(82, 28);
61:          this.lblLang.TabIndex = 0;
62:          this.lblLang.Text = "NS (Language:)";
63:          this.lblLang.TextAlign = System.Drawing.ContentAlignment.MiddleLeft;
64:          //
65:          // cmbLang
66:          //
67:          this.cmbLang.Dock = System.Windows.Forms.DockStyle.Fill;
68:          this.cmbLang.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
69:          this.cmbLang.FormattingEnabled = true;
70:          this.cmbLang.Location = new System.Drawing.Point(91, 3);
71:          this.cmbLang.Name = "cmbLang";
72:          this.cmbLang.Size = new System.Drawing.Size(231, 21);
73:          this.cmbLang.TabIndex = 1;
74:          //
```

```
75:                // BtnOk
76:                //
77:                this.BtnOk.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles
s.Bottom | System.Windows.Forms.AnchorStyles.Right)));
78:                this.BtnOk.Location = new System.Drawing.Point(169, 148);
79:                this.BtnOk.Name = "BtnOk";
80:                this.BtnOk.Size = new System.Drawing.Size(75, 23);
81:                this.BtnOk.TabIndex = 1;
82:                this.BtnOk.Text = "NS (Ok)";
83:                this.BtnOk.UseVisualStyleBackColor = true;
84:                this.BtnOk.Click += new System.EventHandler(this.BtnOk_Click);
85:                //
86:                // BtnRestart
87:                //
88:                this.BtnRestart.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.Anchor
Styles.Bottom | System.Windows.Forms.AnchorStyles.Right)));
89:                this.BtnRestart.Location = new System.Drawing.Point(76, 148);
90:                this.BtnRestart.Name = "BtnRestart";
91:                this.BtnRestart.Size = new System.Drawing.Size(75, 23);
92:                this.BtnRestart.TabIndex = 2;
93:                this.BtnRestart.Text = "NS (Restart)";
94:                this.BtnRestart.UseVisualStyleBackColor = true;
95:                this.BtnRestart.Click += new System.EventHandler(this.BtnRestart_Click);
96:                //
97:                // BtnCancel
98:                //
99:                this.BtnCancel.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorS
tyles.Bottom | System.Windows.Forms.AnchorStyles.Right)));
100:                this.BtnCancel.Location = new System.Drawing.Point(262, 148);
101:                this.BtnCancel.Name = "BtnCancel";
102:                this.BtnCancel.Size = new System.Drawing.Size(75, 23);
103:                this.BtnCancel.TabIndex = 3;
104:                this.BtnCancel.Text = "NS (Cancel)";
105:                this.BtnCancel.UseVisualStyleBackColor = true;
106:                this.BtnCancel.Click += new System.EventHandler(this.BtnCancel_Click);
107:                //
108:                // cbEnbSynHgl
109:                //
110:                this.cbEnbSynHgl.AutoSize = true;
```

```
111:         this.cbEnbSynHgl.Location = new System.Drawing.Point(18, 83);
112:         this.cbEnbSynHgl.Name = "cbEnbSynHgl";
113:         this.cbEnbSynHgl.Size = new System.Drawing.Size(158, 17);
114:         this.cbEnbSynHgl.TabIndex = 4;
115:         this.cbEnbSynHgl.Text = "NS (Enable syntax highlight)";
116:         this.cbEnbSynHgl.UseVisualStyleBackColor = true;
117:         //
118:         // FrmIdeSettings
119:         //
120:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
121:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
122:         this.ClientSize = new System.Drawing.Size(349, 183);
123:         this.Controls.Add(this.cbEnbSynHgl);
124:         this.Controls.Add(this.BtnCancel);
125:         this.Controls.Add(this.BtnRestart);
126:         this.Controls.Add(this.BtnOk);
127:         this.Controls.Add(this.tableLayoutPanel1);
128:         this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.SizableToolWindow;
129:         this.MinimumSize = new System.Drawing.Size(300, 192);
130:         this.Name = "FrmIdeSettings";
131:         this.Text = "FrmIdeSettings";
132:         this.Load += new System.EventHandler(this.FrmIdeSettings_Load);
133:         this.tableLayoutPanel1.ResumeLayout(false);
134:         this.tableLayoutPanel1.PerformLayout();
135:         this.ResumeLayout(false);
136:         this.PerformLayout();
137:     }
138: }
139:
140: #endregion
141:
142: private System.Windows.Forms.TableLayoutPanel tableLayoutPanel1;
143: private System.Windows.Forms.Label lblLang;
144: private System.Windows.Forms.Button BtnOk;
145: private System.Windows.Forms.ComboBox cmbLang;
146: private System.Windows.Forms.Button BtnRestart;
147: private System.Windows.Forms.Button BtnCancel;
148: private System.Windows.Forms.CheckBox cbEnbSynHgl;
149: }
```

150: }


```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Data;
5: using System.Drawing;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9:
10: namespace CuruxaIDE {
11:     public partial class FrmIdeSettings:Form {
12:         public FrmIdeSettings() {
13:             InitializeComponent();
14:         }
15:
16:         private void FrmIdeSettings_Load(object sender, EventArgs e) {
17:             UpdateLang();
18:             LoadCurrentSettings();
19:         }
20:
21:         private void LoadCurrentSettings() {
22:             //idiomas
23:             cmbLang.Items.Clear();
24:             foreach(string lang in i18n.AvailableLanguages) {
25:                 cmbLang.Items.Add(i18n.GetNativeName(lang));
26:             }
27:             cmbLang.SelectedItem = i18n.GetNativeName(i18n.CurrentLanguage);
28:
29:             //activar resalto de sintaxis
30:             if(Settings.EnableSyntaxHighlight) cbEnbSynHgl.Checked = true;
31:             else cbEnbSynHgl.Checked = false;
32:         }
33:
34:         private void UpdateLang() {
35:             Text = i18n.str("Preferences");
36:             BtnRestart.Text = i18n.str("Restart");
37:             BtnOk.Text = i18n.str("Ok");
38:             BtnCancel.Text = i18n.str("Cancel");
39:             lblLang.Text = i18n.str("HumanLanguage:");
```

```
40:                cbEnbSynHgl.Text = i18n.str("EnableSyntaxHighlight");
41:            }
42:
43:            private void BtnRestart_Click(object sender, EventArgs e) {
44:                LoadCurrentSettings();
45:            }
46:
47:            private void BtnCancel_Click(object sender, EventArgs e) {
48:                Close();
49:            }
50:
51:            private void BtnOk_Click(object sender, EventArgs e) {
52:                SavePreferences();
53:                Close();
54:            }
55:
56:            private void SavePreferences() {
57:                i18n.CurrentLanguage = i18n.GetLangFromNativeName((string)cmbLang.SelectedItem);
58:                ((FrmMainWindow)Owner).UpdateLang();
59:                Settings.EnableSyntaxHighlight = cbEnbSynHgl.Checked;
60:            }
61:        }
62:    }
```

```
1: // Archivo generado automáticamente por Visual Studio
2:
3: namespace CuruxaIDE {
4:     partial class FrmMainWindow {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
>
14:        protected override void Dispose(bool disposing) {
15:            if(disposing && (components != null)) {
16:                components.Dispose();
17:            }
18:            base.Dispose(disposing);
19:        }
20:
21:        #region Windows Form Designer generated code
22:
23:        /// <summary>
24:        /// Required method for Designer support - do not modify
25:        /// the contents of this method with the code editor.
26:        /// </summary>
27:        private void InitializeComponent() {
28:            this.components = new System.ComponentModel.Container();
29:            System.ComponentModel.ComponentResourceManager resources = new System.ComponentModel.Compo
nentResourceManager(typeof(FrmMainWindow));
30:            this.StatusStrip = new System.Windows.Forms.StatusStrip();
31:            this.LblStatus = new System.Windows.Forms.ToolStripStatusLabel();
32:            this.StatusProgrammer = new System.Windows.Forms.ToolStripStatusLabel();
33:            this.StatusLin = new System.Windows.Forms.ToolStripStatusLabel();
34:            this.StatusCol = new System.Windows.Forms.ToolStripStatusLabel();
35:            this.menuStrip1 = new System.Windows.Forms.MenuStrip();
36:            this.MiFile = new System.Windows.Forms.ToolStripMenuItem();
37:            this.MiNewPrj = new System.Windows.Forms.ToolStripMenuItem();
```

```
38:         this.MiOpenExpl = new System.Windows.Forms.ToolStripItem();
39:         this.MiOpenPrj = new System.Windows.Forms.ToolStripItem();
40:         this.toolStripMenuItem1 = new System.Windows.Forms.ToolStripItemSeparator();
41:         this.MiPrint = new System.Windows.Forms.ToolStripItem();
42:         this.MiPrintPreview = new System.Windows.Forms.ToolStripItem();
43:         this.MiFileSave = new System.Windows.Forms.ToolStripItem();
44:         this.MiFileClose = new System.Windows.Forms.ToolStripItem();
45:         this.toolStripMenuItem2 = new System.Windows.Forms.ToolStripItemSeparator();
46:         this.MiExit = new System.Windows.Forms.ToolStripItem();
47:         this.MiEdit = new System.Windows.Forms.ToolStripItem();
48:         this.MiUndo = new System.Windows.Forms.ToolStripItem();
49:         this.MiRedo = new System.Windows.Forms.ToolStripItem();
50:         this.toolStripSeparator7 = new System.Windows.Forms.ToolStripItemSeparator();
51:         this.MiCut = new System.Windows.Forms.ToolStripItem();
52:         this.MiCopy = new System.Windows.Forms.ToolStripItem();
53:         this.MiPaste = new System.Windows.Forms.ToolStripItem();
54:         this.toolStripSeparator8 = new System.Windows.Forms.ToolStripItemSeparator();
55:         this.MiSelectAll = new System.Windows.Forms.ToolStripItem();
56:         this.MiPreferences = new System.Windows.Forms.ToolStripItem();
57:         this.MiProject = new System.Windows.Forms.ToolStripItem();
58:         this.MiNewFile = new System.Windows.Forms.ToolStripItem();
59:         this.MiPrjAddFile = new System.Windows.Forms.ToolStripItem();
60:         this.MiPrjSaveAll = new System.Windows.Forms.ToolStripItem();
61:         this.MiPrjClose = new System.Windows.Forms.ToolStripItem();
62:         this.toolStripMenuItem3 = new System.Windows.Forms.ToolStripItemSeparator();
63:         this.MiPrjBuildBurnRun = new System.Windows.Forms.ToolStripItem();
64:         this.MiPrjBuild = new System.Windows.Forms.ToolStripItem();
65:         this.MiPrjProgram = new System.Windows.Forms.ToolStripItem();
66:         this.MiPrjRun = new System.Windows.Forms.ToolStripItem();
67:         this.MiPrjStop = new System.Windows.Forms.ToolStripItem();
68:         this.toolStripMenuItem4 = new System.Windows.Forms.ToolStripItemSeparator();
69:         this.MiCheckMB = new System.Windows.Forms.ToolStripItem();
70:         this.MiPrjSettings = new System.Windows.Forms.ToolStripItem();
71:         this.MiHelp = new System.Windows.Forms.ToolStripItem();
72:         this.MiLanguage = new System.Windows.Forms.ToolStripItem();
73:         this.MiLangSpanish = new System.Windows.Forms.ToolStripItem();
74:         this.MiLangEnglish = new System.Windows.Forms.ToolStripItem();
75:         this.MiWebsite = new System.Windows.Forms.ToolStripItem();
76:         this.MiAbout = new System.Windows.Forms.ToolStripItem();
```

```
77:         this.splitContainer2 = new System.Windows.Forms.SplitContainer();
78:         this.TreePrj = new System.Windows.Forms.TreeView();
79:         this.splitContainer1 = new System.Windows.Forms.SplitContainer();
80:         this.TabsSrc = new CuruxaIDE.SrcTabControl();
81:         this.TabDemo = new System.Windows.Forms.TabPage();
82:         this.TabsMsg = new System.Windows.Forms.TabControl();
83:         this.TabMsgIDE = new System.Windows.Forms.TabPage();
84:         this.TxtLogIDE = new System.Windows.Forms.RichTextBox();
85:         this.TabBuildLog = new System.Windows.Forms.TabPage();
86:         this.TxtLogBuild = new System.Windows.Forms.RichTextBox();
87:         this.TabProgLog = new System.Windows.Forms.TabPage();
88:         this.TxtLogProgrammer = new System.Windows.Forms.RichTextBox();
89:         this.MenuTabs = new System.Windows.Forms.ContextMenuStrip(this.components);
90:         this.MiTabSave = new System.Windows.Forms.ToolStripMenuItem();
91:         this.MiTabClose = new System.Windows.Forms.ToolStripMenuItem();
92:         this.StripPrj = new System.Windows.Forms.ToolStrip();
93:         this.BtnNewFile = new System.Windows.Forms.ToolStripButton();
94:         this.BtnAddFilePrj = new System.Windows.Forms.ToolStripButton();
95:         this.BtnSaveFile = new System.Windows.Forms.ToolStripButton();
96:         this.BtnFileClose = new System.Windows.Forms.ToolStripButton();
97:         this.toolStripSeparator1 = new System.Windows.Forms.ToolStripSeparator();
98:         this.BtnCut = new System.Windows.Forms.ToolStripButton();
99:         this.BtnCopy = new System.Windows.Forms.ToolStripButton();
100:        this.BtnPaste = new System.Windows.Forms.ToolStripButton();
101:        this.toolStripSeparator4 = new System.Windows.Forms.ToolStripSeparator();
102:        this.BtnUndo = new System.Windows.Forms.ToolStripButton();
103:        this.BtnRedo = new System.Windows.Forms.ToolStripButton();
104:        this.toolStripSeparator = new System.Windows.Forms.ToolStripSeparator();
105:        this.BtnSaveAll = new System.Windows.Forms.ToolStripButton();
106:        this.BtnClosePrj = new System.Windows.Forms.ToolStripButton();
107:        this.BtnPrjSettings = new System.Windows.Forms.ToolStripButton();
108:        this.toolStripSeparator2 = new System.Windows.Forms.ToolStripSeparator();
109:        this.BtnPrjBuildBurnRun = new System.Windows.Forms.ToolStripButton();
110:        this.BtnBuild = new System.Windows.Forms.ToolStripButton();
111:        this.BtnProgramMCU = new System.Windows.Forms.ToolStripButton();
112:        this.BtnRun = new System.Windows.Forms.ToolStripButton();
113:        this.BtnStop = new System.Windows.Forms.ToolStripButton();
114:        this.StatusStrip.SuspendLayout();
115:        this.menuStrip1.SuspendLayout();
```

```
116:         this.splitContainer2.Panel1.SuspendLayout();
117:         this.splitContainer2.Panel2.SuspendLayout();
118:         this.splitContainer2.SuspendLayout();
119:         this.splitContainer1.Panel1.SuspendLayout();
120:         this.splitContainer1.Panel2.SuspendLayout();
121:         this.splitContainer1.SuspendLayout();
122:         this.TabsSrc.SuspendLayout();
123:         this.TabsMsg.SuspendLayout();
124:         this.TabMsgIDE.SuspendLayout();
125:         this.TabBuildLog.SuspendLayout();
126:         this.TabProgLog.SuspendLayout();
127:         this.MenuTabs.SuspendLayout();
128:         this.StripPrj.SuspendLayout();
129:         this.SuspendLayout();
130:         //
131:         // StatusStrip
132:         //
133:         this.StatusStrip.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
134:         this.LblStatus,
135:         this.StatusProgrammer,
136:         this.StatusLin,
137:         this.StatusCol});
138:         this.StatusStrip.Location = new System.Drawing.Point(0, 551);
139:         this.StatusStrip.Name = "StatusStrip";
140:         this.StatusStrip.Size = new System.Drawing.Size(792, 22);
141:         this.StatusStrip.TabIndex = 1;
142:         this.StatusStrip.Text = "statusStrip1";
143:         //
144:         // LblStatus
145:         //
146:         this.LblStatus.Name = "LblStatus";
147:         this.LblStatus.Size = new System.Drawing.Size(552, 17);
148:         this.LblStatus.Spring = true;
149:         this.LblStatus.Text = "NS (Status text)";
150:         this.LblStatus.TextAlign = System.Drawing.ContentAlignment.MiddleLeft;
151:         //
152:         // StatusProgrammer
153:         //
154:         this.StatusProgrammer.BorderSides = System.Windows.Forms.ToolStripStatusLabelBorderSides.L
```

```
eft;
155:         this.StatusProgrammer.BorderStyle = System.Windows.Forms.Border3DStyle.Etched;
156:         this.StatusProgrammer.Name = "StatusProgrammer";
157:         this.StatusProgrammer.Size = new System.Drawing.Size(126, 17);
158:         this.StatusProgrammer.Text = "NS (programmer status)";
159:         //
160:         // StatusLin
161:         //
162:         this.StatusLin.BorderSides = System.Windows.Forms.ToolStripStatusLabelBorderSides.Left;
163:         this.StatusLin.BorderStyle = System.Windows.Forms.Border3DStyle.Etched;
164:         this.StatusLin.Name = "StatusLin";
165:         this.StatusLin.Size = new System.Drawing.Size(51, 17);
166:         this.StatusLin.Text = "NS (line)";
167:         //
168:         // StatusCol
169:         //
170:         this.StatusCol.BorderSides = System.Windows.Forms.ToolStripStatusLabelBorderSides.Left;
171:         this.StatusCol.BorderStyle = System.Windows.Forms.Border3DStyle.Etched;
172:         this.StatusCol.Name = "StatusCol";
173:         this.StatusCol.Size = new System.Drawing.Size(48, 17);
174:         this.StatusCol.Text = "NS (col)";
175:         //
176:         // menuStrip1
177:         //
178:         this.menuStrip1.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
179:             this.MiFile,
180:             this.MiEdit,
181:             this.MiProject,
182:             this.MiHelp});
183:         this.menuStrip1.Location = new System.Drawing.Point(0, 0);
184:         this.menuStrip1.Name = "menuStrip1";
185:         this.menuStrip1.Size = new System.Drawing.Size(792, 24);
186:         this.menuStrip1.TabIndex = 2;
187:         this.menuStrip1.Text = "menuStrip1";
188:         //
189:         // MiFile
190:         //
191:         this.MiFile.DropDownItems.AddRange(new System.Windows.Forms.ToolStripItem[] {
192:             this.MiNewPrj,
```

```
193:         this.MiOpenExpl,
194:         this.MiOpenPrj,
195:         this.toolStripMenuItem1,
196:         this.MiPrint,
197:         this.MiPrintPreview,
198:         this.MiFileSave,
199:         this.MiFileClose,
200:         this.toolStripMenuItem2,
201:         this.MiExit});
202:         this.MiFile.Name = "MiFile";
203:         this.MiFile.Size = new System.Drawing.Size(59, 20);
204:         this.MiFile.Text = "NS (File)";
205:         //
206:         // MiNewPrj
207:         //
208:         this.MiNewPrj.Name = "MiNewPrj";
209:         this.MiNewPrj.ShortcutKeys = ((System.Windows.Forms.Keys)(((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.Shift)
210:         | System.Windows.Forms.Keys.N))));
211:         this.MiNewPrj.Size = new System.Drawing.Size(242, 22);
212:         this.MiNewPrj.Text = "NS (New Project)";
213:         this.MiNewPrj.Click += new System.EventHandler(this.MiNewPrj_Click);
214:         //
215:         // MiOpenExpl
216:         //
217:         this.MiOpenExpl.Name = "MiOpenExpl";
218:         this.MiOpenExpl.Size = new System.Drawing.Size(242, 22);
219:         this.MiOpenExpl.Text = "NS (New prj from example)";
220:         this.MiOpenExpl.Click += new System.EventHandler(this.MiOpenExpl_Click);
221:         //
222:         // MiOpenPrj
223:         //
224:         this.MiOpenPrj.Name = "MiOpenPrj";
225:         this.MiOpenPrj.ShortcutKeys = ((System.Windows.Forms.Keys)(((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.Shift)
226:         | System.Windows.Forms.Keys.O))));
227:         this.MiOpenPrj.Size = new System.Drawing.Size(242, 22);
228:         this.MiOpenPrj.Text = "NS (Open Project)";
229:         this.MiOpenPrj.Click += new System.EventHandler(this.MiOpenPrj_Click);
```



```
230:         //
231:         // toolStripMenuItem1
232:         //
233:         this.toolStripMenuItem1.Name = "toolStripMenuItem1";
234:         this.toolStripMenuItem1.Size = new System.Drawing.Size(239, 6);
235:         //
236:         // MiPrint
237:         //
238:         this.MiPrint.Name = "MiPrint";
239:         this.MiPrint.Size = new System.Drawing.Size(242, 22);
240:         this.MiPrint.Text = "NS (Print)";
241:         this.MiPrint.Click += new System.EventHandler(this.MiPrint_Click);
242:         //
243:         // MiPrintPreview
244:         //
245:         this.MiPrintPreview.Enabled = false;
246:         this.MiPrintPreview.Name = "MiPrintPreview";
247:         this.MiPrintPreview.Size = new System.Drawing.Size(242, 22);
248:         this.MiPrintPreview.Text = "NS (Print Preview)";
249:         this.MiPrintPreview.Click += new System.EventHandler(this.MiPrintPreview_Click);
250:         //
251:         // MiFileSave
252:         //
253:         this.MiFileSave.Name = "MiFileSave";
254:         this.MiFileSave.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.S)));
255:         this.MiFileSave.Size = new System.Drawing.Size(242, 22);
256:         this.MiFileSave.Text = "NS (Save)";
257:         this.MiFileSave.Click += new System.EventHandler(this.MiSave_Click);
258:         //
259:         // MiFileClose
260:         //
261:         this.MiFileClose.Name = "MiFileClose";
262:         this.MiFileClose.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.W)));
263:         this.MiFileClose.Size = new System.Drawing.Size(242, 22);
264:         this.MiFileClose.Text = "NS (Close)";
265:         this.MiFileClose.Click += new System.EventHandler(this.MiClose_Click);
266:         //
```

```
267:         // toolStripMenuItem2
268:         //
269:         this.toolStripMenuItem2.Name = "toolStripMenuItem2";
270:         this.toolStripMenuItem2.Size = new System.Drawing.Size(239, 6);
271:         //
272:         // MiExit
273:         //
274:         this.MiExit.Name = "MiExit";
275:         this.MiExit.Size = new System.Drawing.Size(242, 22);
276:         this.MiExit.Text = "NS (Exit)";
277:         this.MiExit.Click += new System.EventHandler(this.MiExit_Click);
278:         //
279:         // MiEdit
280:         //
281:         this.MiEdit.DropDownItems.AddRange(new System.Windows.Forms.ToolStripItem[] {
282:             this.MiUndo,
283:             this.MiRedo,
284:             this.toolStripSeparator7,
285:             this.MiCut,
286:             this.MiCopy,
287:             this.MiPaste,
288:             this.toolStripSeparator8,
289:             this.MiSelectAll,
290:             this.MiPreferences});
291:         this.MiEdit.Name = "MiEdit";
292:         this.MiEdit.Size = new System.Drawing.Size(61, 20);
293:         this.MiEdit.Text = "NS (Edit)";
294:         //
295:         // MiUndo
296:         //
297:         this.MiUndo.Enabled = false;
298:         this.MiUndo.Name = "MiUndo";
299:         this.MiUndo.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.Z)));
300:         this.MiUndo.Size = new System.Drawing.Size(202, 22);
301:         this.MiUndo.Text = "NS (Undo)";
302:         this.MiUndo.Click += new System.EventHandler(this.MiUndo_Click);
303:         //
304:         // MiRedo
```

```
305:          //
306:          this.MiRedo.Enabled = false;
307:          this.MiRedo.Name = "MiRedo";
308:          this.MiRedo.ShortcutKeys = ((System.Windows.Forms.Keys)((((System.Windows.Forms.Keys.Control
1 | System.Windows.Forms.Keys.Shift)
309:                                     | System.Windows.Forms.Keys.Z)))));
310:          this.MiRedo.Size = new System.Drawing.Size(202, 22);
311:          this.MiRedo.Text = "NS (Redo)";
312:          this.MiRedo.Click += new System.EventHandler(this.MiRedo_Click);
313:          //
314:          // toolStripSeparator7
315:          //
316:          this.toolStripSeparator7.Name = "toolStripSeparator7";
317:          this.toolStripSeparator7.Size = new System.Drawing.Size(199, 6);
318:          //
319:          // MiCut
320:          //
321:          this.MiCut.Image = ((System.Drawing.Image)(resources.GetObject("MiCut.Image")));
322:          this.MiCut.ImageTransparentColor = System.Drawing.Color.Magenta;
323:          this.MiCut.Name = "MiCut";
324:          this.MiCut.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control
| System.Windows.Forms.Keys.X)));
325:          this.MiCut.Size = new System.Drawing.Size(202, 22);
326:          this.MiCut.Text = "NS (Cut)";
327:          this.MiCut.Click += new System.EventHandler(this.MiCut_Click);
328:          //
329:          // MiCopy
330:          //
331:          this.MiCopy.Image = ((System.Drawing.Image)(resources.GetObject("MiCopy.Image")));
332:          this.MiCopy.ImageTransparentColor = System.Drawing.Color.Magenta;
333:          this.MiCopy.Name = "MiCopy";
334:          this.MiCopy.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control
| System.Windows.Forms.Keys.C)));
335:          this.MiCopy.Size = new System.Drawing.Size(202, 22);
336:          this.MiCopy.Text = "NS (Copy)";
337:          this.MiCopy.Click += new System.EventHandler(this.MiCopy_Click);
338:          //
339:          // MiPaste
340:          //
```

```
341:         this.MiPaste.Image = ((System.Drawing.Image)(resources.GetObject("MiPaste.Image")));
342:         this.MiPaste.ImageTransparentColor = System.Drawing.Color.Magenta;
343:         this.MiPaste.Name = "MiPaste";
344:         this.MiPaste.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.V)));
345:         this.MiPaste.Size = new System.Drawing.Size(202, 22);
346:         this.MiPaste.Text = "NS (Paste)";
347:         this.MiPaste.Click += new System.EventHandler(this.MiPaste_Click);
348:         //
349:         // toolStripSeparator8
350:         //
351:         this.toolStripSeparator8.Name = "toolStripSeparator8";
352:         this.toolStripSeparator8.Size = new System.Drawing.Size(199, 6);
353:         //
354:         // MiSelectAll
355:         //
356:         this.MiSelectAll.Name = "MiSelectAll";
357:         this.MiSelectAll.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.A)));
358:         this.MiSelectAll.Size = new System.Drawing.Size(202, 22);
359:         this.MiSelectAll.Text = "NS (Select All)";
360:         this.MiSelectAll.Click += new System.EventHandler(this.MiSelectAll_Click);
361:         //
362:         // MiPreferences
363:         //
364:         this.MiPreferences.Name = "MiPreferences";
365:         this.MiPreferences.Size = new System.Drawing.Size(202, 22);
366:         this.MiPreferences.Text = "NS (Preferences)";
367:         this.MiPreferences.Click += new System.EventHandler(this.MiPreferences_Click);
368:         //
369:         // MiProject
370:         //
371:         this.MiProject.DropDownItems.AddRange(new System.Windows.Forms.ToolStripItem[] {
372:             this.MiNewFile,
373:             this.MiPrjAddFile,
374:             this.MiPrjSaveAll,
375:             this.MiPrjClose,
376:             this.toolStripMenuItem3,
377:             this.MiPrjBuildBurnRun,
```

```
378:         this.MiPrjBuild,
379:         this.MiPrjProgram,
380:         this.MiPrjRun,
381:         this.MiPrjStop,
382:         this.toolStripMenuItem4,
383:         this.MiCheckMB,
384:         this.MiPrjSettings});
385:         this.MiProject.Name = "MiProject";
386:         this.MiProject.Size = new System.Drawing.Size(77, 20);
387:         this.MiProject.Text = "NS (Project)";
388:         //
389:         // MiNewFile
390:         //
391:         this.MiNewFile.Enabled = false;
392:         this.MiNewFile.Name = "MiNewFile";
393:         this.MiNewFile.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.N)));
394:         this.MiNewFile.Size = new System.Drawing.Size(227, 22);
395:         this.MiNewFile.Text = "NS (New File)";
396:         this.MiNewFile.Click += new System.EventHandler(this.MiNewFile_Click);
397:         //
398:         // MiPrjAddFile
399:         //
400:         this.MiPrjAddFile.Enabled = false;
401:         this.MiPrjAddFile.Name = "MiPrjAddFile";
402:         this.MiPrjAddFile.ShortcutKeyDisplayString = "Ctrl+O";
403:         this.MiPrjAddFile.Size = new System.Drawing.Size(227, 22);
404:         this.MiPrjAddFile.Text = "NS (Add Existing File)";
405:         this.MiPrjAddFile.Click += new System.EventHandler(this.MiPrjAddFile_Click);
406:         //
407:         // MiPrjSaveAll
408:         //
409:         this.MiPrjSaveAll.Name = "MiPrjSaveAll";
410:         this.MiPrjSaveAll.ShortcutKeys = ((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control | System.Windows.Forms.Keys.Shift | System.Windows.Forms.Keys.S)));
411:         this.MiPrjSaveAll.Size = new System.Drawing.Size(227, 22);
412:         this.MiPrjSaveAll.Text = "NS (Save All)";
413:         this.MiPrjSaveAll.Click += new System.EventHandler(this.MiPrjSaveAll_Click);
414:
```

```
415:         //
416:         // MiPrjClose
417:         //
418:         this.MiPrjClose.Name = "MiPrjClose";
419:         this.MiPrjClose.Size = new System.Drawing.Size(227, 22);
420:         this.MiPrjClose.Text = "NS (Close)";
421:         this.MiPrjClose.Click += new System.EventHandler(this.MiPrjClose_Click);
422:         //
423:         // toolStripMenuItem3
424:         //
425:         this.toolStripMenuItem3.Name = "toolStripMenuItem3";
426:         this.toolStripMenuItem3.Size = new System.Drawing.Size(224, 6);
427:         //
428:         // MiPrjBuildBurnRun
429:         //
430:         this.MiPrjBuildBurnRun.Name = "MiPrjBuildBurnRun";
431:         this.MiPrjBuildBurnRun.ShortcutKeys = System.Windows.Forms.Keys.F4;
432:         this.MiPrjBuildBurnRun.Size = new System.Drawing.Size(227, 22);
433:         this.MiPrjBuildBurnRun.Text = "NS (Build, Burn, Run)";
434:         this.MiPrjBuildBurnRun.Click += new System.EventHandler(this.MiBuildBurnRun_Click);
435:         //
436:         // MiPrjBuild
437:         //
438:         this.MiPrjBuild.Name = "MiPrjBuild";
439:         this.MiPrjBuild.ShortcutKeys = System.Windows.Forms.Keys.F5;
440:         this.MiPrjBuild.Size = new System.Drawing.Size(227, 22);
441:         this.MiPrjBuild.Text = "NS (Build)";
442:         this.MiPrjBuild.Click += new System.EventHandler(this.MiBuildPrj_Click);
443:         //
444:         // MiPrjProgram
445:         //
446:         this.MiPrjProgram.Name = "MiPrjProgram";
447:         this.MiPrjProgram.ShortcutKeyDisplayString = "";
448:         this.MiPrjProgram.ShortcutKeys = System.Windows.Forms.Keys.F6;
449:         this.MiPrjProgram.Size = new System.Drawing.Size(227, 22);
450:         this.MiPrjProgram.Text = "NS (Program)";
451:         this.MiPrjProgram.Click += new System.EventHandler(this.MiProgramPrj_Click);
452:         //
453:         // MiPrjRun
```

```
454:         //
455:         this.MiPrjRun.Name = "MiPrjRun";
456:         this.MiPrjRun.ShortcutKeys = System.Windows.Forms.Keys.F7;
457:         this.MiPrjRun.Size = new System.Drawing.Size(227, 22);
458:         this.MiPrjRun.Text = "NS (Run)";
459:         this.MiPrjRun.Click += new System.EventHandler(this.MiRunPrj_Click);
460:         //
461:         // MiPrjStop
462:         //
463:         this.MiPrjStop.Name = "MiPrjStop";
464:         this.MiPrjStop.ShortcutKeyDisplayString = "";
465:         this.MiPrjStop.ShortcutKeys = System.Windows.Forms.Keys.F8;
466:         this.MiPrjStop.Size = new System.Drawing.Size(227, 22);
467:         this.MiPrjStop.Text = "NS (Stop)";
468:         this.MiPrjStop.Click += new System.EventHandler(this.MiStopPrj_Click);
469:         //
470:         // toolStripMenuItem4
471:         //
472:         this.toolStripMenuItem4.Name = "toolStripMenuItem4";
473:         this.toolStripMenuItem4.Size = new System.Drawing.Size(224, 6);
474:         //
475:         // MiCheckMB
476:         //
477:         this.MiCheckMB.Name = "MiCheckMB";
478:         this.MiCheckMB.Size = new System.Drawing.Size(227, 22);
479:         this.MiCheckMB.Text = "NS (Check Main Board)";
480:         this.MiCheckMB.Click += new System.EventHandler(this.MiCheckMB_Click);
481:         //
482:         // MiPrjSettings
483:         //
484:         this.MiPrjSettings.Name = "MiPrjSettings";
485:         this.MiPrjSettings.Size = new System.Drawing.Size(227, 22);
486:         this.MiPrjSettings.Text = "NS (Settings)";
487:         this.MiPrjSettings.Click += new System.EventHandler(this.MiPrjSettings_Click);
488:         //
489:         // MiHelp
490:         //
491:         this.MiHelp.DropDownItems.AddRange(new System.Windows.Forms.ToolStripItem[] {
492:             this.MiLanguage,
```

```
493:         this.MiWebsite,
494:         this.MiAbout});
495:         this.MiHelp.Name = "MiHelp";
496:         this.MiHelp.Size = new System.Drawing.Size(64, 20);
497:         this.MiHelp.Text = "NS (Help)";
498:         //
499:         // MiLanguage
500:         //
501:         this.MiLanguage.DropDownItems.AddRange(new System.Windows.Forms.ToolStripItem[] {
502:         this.MiLangSpanish,
503:         this.MiLangEnglish});
504:         this.MiLanguage.Name = "MiLanguage";
505:         this.MiLanguage.Size = new System.Drawing.Size(205, 22);
506:         this.MiLanguage.Text = "NS (Language)";
507:         //
508:         // MiLangSpanish
509:         //
510:         this.MiLangSpanish.Name = "MiLangSpanish";
511:         this.MiLangSpanish.Size = new System.Drawing.Size(122, 22);
512:         this.MiLangSpanish.Text = "Español";
513:         this.MiLangSpanish.Click += new System.EventHandler(this.MiLangSpanish_Click);
514:         //
515:         // MiLangEnglish
516:         //
517:         this.MiLangEnglish.Name = "MiLangEnglish";
518:         this.MiLangEnglish.Size = new System.Drawing.Size(122, 22);
519:         this.MiLangEnglish.Text = "English";
520:         this.MiLangEnglish.Click += new System.EventHandler(this.MiLangEnglish_Click);
521:         //
522:         // MiWebsite
523:         //
524:         this.MiWebsite.Name = "MiWebsite";
525:         this.MiWebsite.ShortcutKeys = System.Windows.Forms.Keys.F1;
526:         this.MiWebsite.Size = new System.Drawing.Size(205, 22);
527:         this.MiWebsite.Text = "NS (Curuxa Website)";
528:         this.MiWebsite.Click += new System.EventHandler(this.MiWebsite_Click);
529:         //
530:         // MiAbout
531:         //
```



```
532:         this.MiAbout.Name = "MiAbout";
533:         this.MiAbout.Size = new System.Drawing.Size(205, 22);
534:         this.MiAbout.Text = "NS (About Curuxa IDE)";
535:         this.MiAbout.Click += new System.EventHandler(this.MiAbout_Click);
536:         //
537:         // splitContainer2
538:         //
539:         this.splitContainer2.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.
AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)
540:         | System.Windows.Forms.AnchorStyles.Left)
541:         | System.Windows.Forms.AnchorStyles.Right)));
542:         this.splitContainer2.Location = new System.Drawing.Point(0, 52);
543:         this.splitContainer2.Name = "splitContainer2";
544:         //
545:         // splitContainer2.Panel1
546:         //
547:         this.splitContainer2.Panel1.Controls.Add(this.TreePrj);
548:         this.splitContainer2.Panel1MinSize = 150;
549:         //
550:         // splitContainer2.Panel2
551:         //
552:         this.splitContainer2.Panel2.Controls.Add(this.splitContainer1);
553:         this.splitContainer2.Size = new System.Drawing.Size(792, 496);
554:         this.splitContainer2.SplitterDistance = 182;
555:         this.splitContainer2.TabIndex = 0;
556:         //
557:         // TreePrj
558:         //
559:         this.TreePrj.Dock = System.Windows.Forms.DockStyle.Fill;
560:         this.TreePrj.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F, System.Drawing.
FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
561:         this.TreePrj.FullRowSelect = true;
562:         this.TreePrj.HideSelection = false;
563:         this.TreePrj.Location = new System.Drawing.Point(0, 0);
564:         this.TreePrj.Name = "TreePrj";
565:         this.TreePrj.PathSeparator = "/";
566:         this.TreePrj.ShowNodeToolTips = true;
567:         this.TreePrj.Size = new System.Drawing.Size(182, 496);
568:         this.TreePrj.TabIndex = 0;
```

```
569:         this.TreePrj.NodeMouseDoubleClick += new System.Windows.Forms.TreeNodeMouseClickEventHandler(
er(this.TreePrj_NodeMouseDoubleClick));
570:         this.TreePrj.NodeMouseClick += new System.Windows.Forms.TreeNodeMouseClickEventHandler(this.
s.TreePrj_NodeMouseClick);
571:         //
572:         // splitContainer1
573:         //
574:         this.splitContainer1.Dock = System.Windows.Forms.DockStyle.Fill;
575:         this.splitContainer1.Location = new System.Drawing.Point(0, 0);
576:         this.splitContainer1.Name = "splitContainer1";
577:         this.splitContainer1.Orientation = System.Windows.Forms.Orientation.Horizontal;
578:         //
579:         // splitContainer1.Panel1
580:         //
581:         this.splitContainer1.Panel1.Controls.Add(this.TabsSrc);
582:         //
583:         // splitContainer1.Panel2
584:         //
585:         this.splitContainer1.Panel2.Controls.Add(this.TabsMsg);
586:         this.splitContainer1.Size = new System.Drawing.Size(606, 496);
587:         this.splitContainer1.SplitterDistance = 361;
588:         this.splitContainer1.TabIndex = 0;
589:         //
590:         // TabsSrc
591:         //
592:         this.TabsSrc.Controls.Add(this.TabDemo);
593:         this.TabsSrc.Dock = System.Windows.Forms.DockStyle.Fill;
594:         this.TabsSrc.Location = new System.Drawing.Point(0, 0);
595:         this.TabsSrc.Name = "TabsSrc";
596:         this.TabsSrc.SelectedIndex = 0;
597:         this.TabsSrc.Size = new System.Drawing.Size(606, 361);
598:         this.TabsSrc.TabIndex = 1;
599:         //
600:         // TabDemo
601:         //
602:         this.TabDemo.Location = new System.Drawing.Point(4, 22);
603:         this.TabDemo.Name = "TabDemo";
604:         this.TabDemo.Padding = new System.Windows.Forms.Padding(3);
605:         this.TabDemo.Size = new System.Drawing.Size(598, 335);
```

```
606:         this.TabDemo.TabIndex = 0;
607:         this.TabDemo.Text = "demo tab page";
608:         this.TabDemo.UseVisualStyleBackColor = true;
609:         //
610:         // TabsMsg
611:         //
612:         this.TabsMsg.Controls.Add(this.TabMsgIDE);
613:         this.TabsMsg.Controls.Add(this.TabBuildLog);
614:         this.TabsMsg.Controls.Add(this.TabProgLog);
615:         this.TabsMsg.Dock = System.Windows.Forms.DockStyle.Fill;
616:         this.TabsMsg.Location = new System.Drawing.Point(0, 0);
617:         this.TabsMsg.Name = "TabsMsg";
618:         this.TabsMsg.SelectedIndex = 0;
619:         this.TabsMsg.Size = new System.Drawing.Size(606, 131);
620:         this.TabsMsg.TabIndex = 0;
621:         //
622:         // TabMsgIDE
623:         //
624:         this.TabMsgIDE.Controls.Add(this.TxtLogIDE);
625:         this.TabMsgIDE.Location = new System.Drawing.Point(4, 22);
626:         this.TabMsgIDE.Name = "TabMsgIDE";
627:         this.TabMsgIDE.Padding = new System.Windows.Forms.Padding(3);
628:         this.TabMsgIDE.Size = new System.Drawing.Size(598, 105);
629:         this.TabMsgIDE.TabIndex = 0;
630:         this.TabMsgIDE.Text = "NS (IDE)";
631:         this.TabMsgIDE.UseVisualStyleBackColor = true;
632:         //
633:         // TxtLogIDE
634:         //
635:         this.TxtLogIDE.BackColor = System.Drawing.SystemColors.Window;
636:         this.TxtLogIDE.Dock = System.Windows.Forms.DockStyle.Fill;
637:         this.TxtLogIDE.Location = new System.Drawing.Point(3, 3);
638:         this.TxtLogIDE.Name = "TxtLogIDE";
639:         this.TxtLogIDE.ReadOnly = true;
640:         this.TxtLogIDE.ScrollBars = System.Windows.Forms.RichTextBoxScrollBars.ForcedVertical;
641:         this.TxtLogIDE.Size = new System.Drawing.Size(592, 99);
642:         this.TxtLogIDE.TabIndex = 0;
643:         this.TxtLogIDE.Text = "";
644:         //
```

```
645:         // TabBuildLog
646:         //
647:         this.TabBuildLog.Controls.Add(this.TxtLogBuild);
648:         this.TabBuildLog.Location = new System.Drawing.Point(4, 22);
649:         this.TabBuildLog.Name = "TabBuildLog";
650:         this.TabBuildLog.Padding = new System.Windows.Forms.Padding(3);
651:         this.TabBuildLog.Size = new System.Drawing.Size(598, 105);
652:         this.TabBuildLog.TabIndex = 1;
653:         this.TabBuildLog.Text = "NS (build log)";
654:         this.TabBuildLog.UseVisualStyleBackColor = true;
655:         //
656:         // TxtLogBuild
657:         //
658:         this.TxtLogBuild.BackColor = System.Drawing.SystemColors.Window;
659:         this.TxtLogBuild.Dock = System.Windows.Forms.DockStyle.Fill;
660:         this.TxtLogBuild.Location = new System.Drawing.Point(3, 3);
661:         this.TxtLogBuild.Name = "TxtLogBuild";
662:         this.TxtLogBuild.ReadOnly = true;
663:         this.TxtLogBuild.ScrollBars = System.Windows.Forms.RichTextBoxScrollBars.ForcedVertical;
664:         this.TxtLogBuild.Size = new System.Drawing.Size(592, 99);
665:         this.TxtLogBuild.TabIndex = 1;
666:         this.TxtLogBuild.Text = "";
667:         //
668:         // TabProgLog
669:         //
670:         this.TabProgLog.Controls.Add(this.TxtLogProgrammer);
671:         this.TabProgLog.Location = new System.Drawing.Point(4, 22);
672:         this.TabProgLog.Name = "TabProgLog";
673:         this.TabProgLog.Padding = new System.Windows.Forms.Padding(3);
674:         this.TabProgLog.Size = new System.Drawing.Size(598, 105);
675:         this.TabProgLog.TabIndex = 2;
676:         this.TabProgLog.Text = "NS (prog log)";
677:         this.TabProgLog.UseVisualStyleBackColor = true;
678:         //
679:         // TxtLogProgrammer
680:         //
681:         this.TxtLogProgrammer.BackColor = System.Drawing.SystemColors.Window;
682:         this.TxtLogProgrammer.Dock = System.Windows.Forms.DockStyle.Fill;
683:         this.TxtLogProgrammer.Location = new System.Drawing.Point(3, 3);
```

```
684:         this.TxtLogProgrammer.Name = "TxtLogProgrammer";
685:         this.TxtLogProgrammer.ReadOnly = true;
686:         this.TxtLogProgrammer.ScrollBars = System.Windows.Forms.RichTextBoxScrollBars.ForcedVertical;
al;
687:         this.TxtLogProgrammer.Size = new System.Drawing.Size(592, 99);
688:         this.TxtLogProgrammer.TabIndex = 2;
689:         this.TxtLogProgrammer.Text = "";
690:         //
691:         // MenuTabs
692:         //
693:         this.MenuTabs.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
694:         this.MiTabSave,
695:         this.MiTabClose});
696:         this.MenuTabs.Name = "MenuTabs";
697:         this.MenuTabs.Size = new System.Drawing.Size(136, 48);
698:         //
699:         // MiTabSave
700:         //
701:         this.MiTabSave.Name = "MiTabSave";
702:         this.MiTabSave.Size = new System.Drawing.Size(135, 22);
703:         this.MiTabSave.Text = "NS (Save)";
704:         //
705:         // MiTabClose
706:         //
707:         this.MiTabClose.Name = "MiTabClose";
708:         this.MiTabClose.Size = new System.Drawing.Size(135, 22);
709:         this.MiTabClose.Text = "NS (Close)";
710:         //
711:         // StripPrj
712:         //
713:         this.StripPrj.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
714:         this.BtnNewFile,
715:         this.BtnAddFilePrj,
716:         this.BtnSaveFile,
717:         this.BtnFileClose,
718:         this.toolStripSeparator1,
719:         this.BtnCut,
720:         this.BtnCopy,
721:         this.BtnPaste,
```

```
722:         this.toolStripSeparator4,
723:         this.BtnUndo,
724:         this.BtnRedo,
725:         this.toolStripSeparator,
726:         this.BtnSaveAll,
727:         this.BtnClosePrj,
728:         this.BtnPrjSettings,
729:         this.toolStripSeparator2,
730:         this.BtnPrjBuildBurnRun,
731:         this.BtnBuild,
732:         this.BtnProgramMCU,
733:         this.BtnRun,
734:         this.BtnStop});
735:         this.StripPrj.Location = new System.Drawing.Point(0, 24);
736:         this.StripPrj.Name = "StripPrj";
737:         this.StripPrj.Size = new System.Drawing.Size(792, 25);
738:         this.StripPrj.TabIndex = 4;
739:         this.StripPrj.Text = "ToolStrip";
740:         //
741:         // BtnNewFile
742:         //
743:         this.BtnNewFile.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
744:         this.BtnNewFile.Image = ((System.Drawing.Image)(resources.GetObject("BtnNewFile.Image")));
745:         this.BtnNewFile.ImageTransparentColor = System.Drawing.Color.Black;
746:         this.BtnNewFile.Name = "BtnNewFile";
747:         this.BtnNewFile.Size = new System.Drawing.Size(23, 22);
748:         this.BtnNewFile.Text = "NS (New File)";
749:         this.BtnNewFile.Click += new System.EventHandler(this.BtnNewFile_Click);
750:         //
751:         // BtnAddFilePrj
752:         //
753:         this.BtnAddFilePrj.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
754:         this.BtnAddFilePrj.Image = ((System.Drawing.Image)(resources.GetObject("BtnAddFilePrj.Imag
e"))));
755:         this.BtnAddFilePrj.ImageTransparentColor = System.Drawing.Color.Black;
756:         this.BtnAddFilePrj.Name = "BtnAddFilePrj";
757:         this.BtnAddFilePrj.Size = new System.Drawing.Size(23, 22);
758:         this.BtnAddFilePrj.Text = "NS (Add Existing File to Project)";
759:         this.BtnAddFilePrj.Click += new System.EventHandler(this.BtnAddFilePrj_Click);
```

```
760:         //
761:         // BtnSaveFile
762:         //
763:         this.BtnSaveFile.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
764:         this.BtnSaveFile.Image = ((System.Drawing.Image)(resources.GetObject("BtnSaveFile.Image")))
);
765:         this.BtnSaveFile.ImageTransparentColor = System.Drawing.Color.Black;
766:         this.BtnSaveFile.Name = "BtnSaveFile";
767:         this.BtnSaveFile.Size = new System.Drawing.Size(23, 22);
768:         this.BtnSaveFile.Text = "NS (Save File)";
769:         this.BtnSaveFile.Click += new System.EventHandler(this.BtnSaveFile_Click);
770:         //
771:         // BtnFileClose
772:         //
773:         this.BtnFileClose.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
774:         this.BtnFileClose.Image = ((System.Drawing.Image)(resources.GetObject("BtnFileClose.Image")))
));
775:         this.BtnFileClose.ImageTransparentColor = System.Drawing.Color.Magenta;
776:         this.BtnFileClose.Name = "BtnFileClose";
777:         this.BtnFileClose.Size = new System.Drawing.Size(23, 22);
778:         this.BtnFileClose.Text = "NS (close file)";
779:         this.BtnFileClose.Click += new System.EventHandler(this.BtnFileClose_Click);
780:         //
781:         // toolStripSeparator1
782:         //
783:         this.toolStripSeparator1.Name = "toolStripSeparator1";
784:         this.toolStripSeparator1.Size = new System.Drawing.Size(6, 25);
785:         //
786:         // BtnCut
787:         //
788:         this.BtnCut.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
789:         this.BtnCut.Image = ((System.Drawing.Image)(resources.GetObject("BtnCut.Image")));
790:         this.BtnCut.ImageTransparentColor = System.Drawing.Color.Magenta;
791:         this.BtnCut.Name = "BtnCut";
792:         this.BtnCut.Size = new System.Drawing.Size(23, 22);
793:         this.BtnCut.Text = "C&ut";
794:         this.BtnCut.Click += new System.EventHandler(this.BtnCut_Click);
795:         //
796:         // BtnCopy
```

```
797: //
798: this.BtnCopy.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
799: this.BtnCopy.Image = ((System.Drawing.Image)(resources.GetObject("BtnCopy.Image")));
800: this.BtnCopy.ImageTransparentColor = System.Drawing.Color.Magenta;
801: this.BtnCopy.Name = "BtnCopy";
802: this.BtnCopy.Size = new System.Drawing.Size(23, 22);
803: this.BtnCopy.Text = "&Copy";
804: this.BtnCopy.Click += new System.EventHandler(this.BtnCopy_Click);
805: //
806: // BtnPaste
807: //
808: this.BtnPaste.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
809: this.BtnPaste.Image = ((System.Drawing.Image)(resources.GetObject("BtnPaste.Image")));
810: this.BtnPaste.ImageTransparentColor = System.Drawing.Color.Magenta;
811: this.BtnPaste.Name = "BtnPaste";
812: this.BtnPaste.Size = new System.Drawing.Size(23, 22);
813: this.BtnPaste.Text = "&Paste";
814: this.BtnPaste.Click += new System.EventHandler(this.BtnPaste_Click);
815: //
816: // toolStripSeparator4
817: //
818: this.toolStripSeparator4.Name = "toolStripSeparator4";
819: this.toolStripSeparator4.Size = new System.Drawing.Size(6, 25);
820: //
821: // BtnUndo
822: //
823: this.BtnUndo.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
824: this.BtnUndo.Enabled = false;
825: this.BtnUndo.Image = ((System.Drawing.Image)(resources.GetObject("BtnUndo.Image")));
826: this.BtnUndo.ImageTransparentColor = System.Drawing.Color.Magenta;
827: this.BtnUndo.Name = "BtnUndo";
828: this.BtnUndo.Size = new System.Drawing.Size(23, 22);
829: this.BtnUndo.Text = "NS (Undo)";
830: this.BtnUndo.Click += new System.EventHandler(this.BtnUndo_Click);
831: //
832: // BtnRedo
833: //
834: this.BtnRedo.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
835: this.BtnRedo.Enabled = false;
```



```
836:         this.BtnRedo.Image = ((System.Drawing.Image)(resources.GetObject("BtnRedo.Image")));
837:         this.BtnRedo.ImageTransparentColor = System.Drawing.Color.Magenta;
838:         this.BtnRedo.Name = "BtnRedo";
839:         this.BtnRedo.Size = new System.Drawing.Size(23, 22);
840:         this.BtnRedo.Text = "NS (Redo)";
841:         this.BtnRedo.Click += new System.EventHandler(this.BtnRedo_Click);
842:         //
843:         // toolStripSeparator
844:         //
845:         this.toolStripSeparator.Name = "toolStripSeparator";
846:         this.toolStripSeparator.Size = new System.Drawing.Size(6, 25);
847:         //
848:         // BtnSaveAll
849:         //
850:         this.BtnSaveAll.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
851:         this.BtnSaveAll.Image = ((System.Drawing.Image)(resources.GetObject("BtnSaveAll.Image")));
852:         this.BtnSaveAll.ImageTransparentColor = System.Drawing.Color.Magenta;
853:         this.BtnSaveAll.Name = "BtnSaveAll";
854:         this.BtnSaveAll.Size = new System.Drawing.Size(23, 22);
855:         this.BtnSaveAll.Text = "NS (Save All)";
856:         this.BtnSaveAll.Click += new System.EventHandler(this.BtnSaveAll_Click);
857:         //
858:         // BtnClosePrj
859:         //
860:         this.BtnClosePrj.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
861:         this.BtnClosePrj.Image = ((System.Drawing.Image)(resources.GetObject("BtnClosePrj.Image")))
);
862:         this.BtnClosePrj.ImageTransparentColor = System.Drawing.Color.Magenta;
863:         this.BtnClosePrj.Name = "BtnClosePrj";
864:         this.BtnClosePrj.Size = new System.Drawing.Size(23, 22);
865:         this.BtnClosePrj.Text = "NS (Close Project)";
866:         this.BtnClosePrj.Click += new System.EventHandler(this.BtnClosePrj_Click);
867:         //
868:         // BtnPrjSettings
869:         //
870:         this.BtnPrjSettings.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
871:         this.BtnPrjSettings.Image = ((System.Drawing.Image)(resources.GetObject("BtnPrjSettings.Im
age"))));
872:         this.BtnPrjSettings.ImageTransparentColor = System.Drawing.Color.Magenta;
```

```
873:         this.BtnPrjSettings.Name = "BtnPrjSettings";
874:         this.BtnPrjSettings.Size = new System.Drawing.Size(23, 22);
875:         this.BtnPrjSettings.Text = "NS (Project Settings)";
876:         this.BtnPrjSettings.Click += new System.EventHandler(this.BtnConfigPrj_Click);
877:         //
878:         // toolStripSeparator2
879:         //
880:         this.toolStripSeparator2.Name = "toolStripSeparator2";
881:         this.toolStripSeparator2.Size = new System.Drawing.Size(6, 25);
882:         //
883:         // BtnPrjBuildBurnRun
884:         //
885:         this.BtnPrjBuildBurnRun.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
e;
886:         this.BtnPrjBuildBurnRun.Image = ((System.Drawing.Image)(resources.GetObject("BtnPrjBuildBu
rnRun.Image"))));
887:         this.BtnPrjBuildBurnRun.ImageTransparentColor = System.Drawing.Color.Magenta;
888:         this.BtnPrjBuildBurnRun.Name = "BtnPrjBuildBurnRun";
889:         this.BtnPrjBuildBurnRun.Size = new System.Drawing.Size(23, 22);
890:         this.BtnPrjBuildBurnRun.Text = "toolStripButton1";
891:         this.BtnPrjBuildBurnRun.Click += new System.EventHandler(this.BtnBuildBurnRun_Click);
892:         //
893:         // BtnBuild
894:         //
895:         this.BtnBuild.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
896:         this.BtnBuild.Image = ((System.Drawing.Image)(resources.GetObject("BtnBuild.Image")));
897:         this.BtnBuild.ImageTransparentColor = System.Drawing.Color.Magenta;
898:         this.BtnBuild.Name = "BtnBuild";
899:         this.BtnBuild.Size = new System.Drawing.Size(23, 22);
900:         this.BtnBuild.Text = "NS (Build)";
901:         this.BtnBuild.Click += new System.EventHandler(this.BtnBuild_Click);
902:         //
903:         // BtnProgramMCU
904:         //
905:         this.BtnProgramMCU.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
906:         this.BtnProgramMCU.Image = ((System.Drawing.Image)(resources.GetObject("BtnProgramMCU.Imag
e"))));
907:         this.BtnProgramMCU.ImageTransparentColor = System.Drawing.Color.Magenta;
908:         this.BtnProgramMCU.Name = "BtnProgramMCU";
```

```
909:         this.BtnProgramMCU.Size = new System.Drawing.Size(23, 22);
910:         this.BtnProgramMCU.Text = "NS (Write App to MCU)";
911:         this.BtnProgramMCU.Click += new System.EventHandler(this.BtnProgramMCU_Click);
912:         //
913:         // BtnRun
914:         //
915:         this.BtnRun.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
916:         this.BtnRun.Image = ((System.Drawing.Image)(resources.GetObject("BtnRun.Image")));
917:         this.BtnRun.ImageTransparentColor = System.Drawing.Color.Magenta;
918:         this.BtnRun.Name = "BtnRun";
919:         this.BtnRun.Size = new System.Drawing.Size(23, 22);
920:         this.BtnRun.Text = "NS (Run Program)";
921:         this.BtnRun.Click += new System.EventHandler(this.BtnRun_Click);
922:         //
923:         // BtnStop
924:         //
925:         this.BtnStop.DisplayStyle = System.Windows.Forms.ToolStripItemDisplayStyle.Image;
926:         this.BtnStop.Image = ((System.Drawing.Image)(resources.GetObject("BtnStop.Image")));
927:         this.BtnStop.ImageTransparentColor = System.Drawing.Color.Magenta;
928:         this.BtnStop.Name = "BtnStop";
929:         this.BtnStop.Size = new System.Drawing.Size(23, 22);
930:         this.BtnStop.Text = "NS (Stop)";
931:         this.BtnStop.Click += new System.EventHandler(this.BtnStop_Click);
932:         //
933:         // FrmMainWindow
934:         //
935:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
936:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
937:         this.ClientSize = new System.Drawing.Size(792, 573);
938:         this.Controls.Add(this.splitContainer2);
939:         this.Controls.Add(this.StripPrj);
940:         this.Controls.Add(this.StatusStrip);
941:         this.Controls.Add(this.menuStrip1);
942:         this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.Icon")));
943:         this.MainMenuStrip = this.menuStrip1;
944:         this.MinimumSize = new System.Drawing.Size(400, 350);
945:         this.Name = "FrmMainWindow";
946:         this.Text = "NS (Title)";
947:         this.Load += new System.EventHandler(this.MainWindow_Load);
```

```
948:         this.FormClosing += new System.Windows.Forms.FormClosingEventHandler(this.FrmMainWindow_Fo
rmClosing);
949:         this.StatusStrip.ResumeLayout(false);
950:         this.StatusStrip.PerformLayout();
951:         this.menuStrip1.ResumeLayout(false);
952:         this.menuStrip1.PerformLayout();
953:         this.splitContainer2.Panel1.ResumeLayout(false);
954:         this.splitContainer2.Panel2.ResumeLayout(false);
955:         this.splitContainer2.ResumeLayout(false);
956:         this.splitContainer1.Panel1.ResumeLayout(false);
957:         this.splitContainer1.Panel2.ResumeLayout(false);
958:         this.splitContainer1.ResumeLayout(false);
959:         this.TabsSrc.ResumeLayout(false);
960:         this.TabsMsg.ResumeLayout(false);
961:         this.TabMsgIDE.ResumeLayout(false);
962:         this.TabBuildLog.ResumeLayout(false);
963:         this.TabProgLog.ResumeLayout(false);
964:         this.MenuTabs.ResumeLayout(false);
965:         this.StripPrj.ResumeLayout(false);
966:         this.StripPrj.PerformLayout();
967:         this.ResumeLayout(false);
968:         this.PerformLayout();
969:
970:     }
971:
972:     #endregion
973:
974:     private System.Windows.Forms.StatusStrip StatusStrip;
975:     private System.Windows.Forms.ToolStripStatusLabel LblStatus;
976:     private System.Windows.Forms.MenuStrip menuStrip1;
977:     private System.Windows.Forms.ToolStripMenuItem MiFile;
978:     private System.Windows.Forms.ToolStripMenuItem MiNewPrj;
979:     private System.Windows.Forms.ToolStripMenuItem MiOpenExpl;
980:     private System.Windows.Forms.ToolStripSeparator toolStripMenuItem1;
981:     private System.Windows.Forms.ToolStripMenuItem MiFileSave;
982:     private System.Windows.Forms.ToolStripSeparator toolStripMenuItem2;
983:     private System.Windows.Forms.ToolStripMenuItem MiExit;
984:     private System.Windows.Forms.ToolStripMenuItem MiProject;
985:     private System.Windows.Forms.ToolStripMenuItem MiNewFile;
```

```
986:         private System.Windows.Forms.ToolStripItem MiPrjAddFile;
987:         private System.Windows.Forms.ToolStripItemSeparator toolStripMenuItem3;
988:         private System.Windows.Forms.ToolStripItem MiPrjSettings;
989:         private System.Windows.Forms.ToolStripItem MiHelp;
990:         private System.Windows.Forms.ToolStripItem MiPrjBuild;
991:         private System.Windows.Forms.ToolStripItem MiPrjProgram;
992:         private System.Windows.Forms.ToolStripItem MiPrjRun;
993:         private System.Windows.Forms.ToolStripItem MiPrjStop;
994:         private System.Windows.Forms.ToolStripItem MiWebsite;
995:         private System.Windows.Forms.ToolStripItem MiAbout;
996:         private System.Windows.Forms.ToolStripItemSeparator toolStripMenuItem4;
997:         private System.Windows.Forms.SplitContainer splitContainer2;
998:         private System.Windows.Forms.TreeView TreePrj;
999:         private System.Windows.Forms.RichTextBox TxtLogIDE;
1000:        private System.Windows.Forms.ToolStripItem MiPrjClose;
1001:        private System.Windows.Forms.ToolStripItem StripPrj;
1002:        private System.Windows.Forms.ToolStripButton BtnNewFile;
1003:        private System.Windows.Forms.ToolStripButton BtnAddFilePrj;
1004:        private System.Windows.Forms.ToolStripButton BtnSaveFile;
1005:        private System.Windows.Forms.ToolStripItemSeparator toolStripSeparator1;
1006:        private System.Windows.Forms.ToolStripItemSeparator toolStripSeparator2;
1007:        private System.Windows.Forms.ToolStripButton BtnBuild;
1008:        private System.Windows.Forms.ToolStripButton BtnProgramMCU;
1009:        private System.Windows.Forms.ToolStripButton BtnRun;
1010:        private System.Windows.Forms.ToolStripButton BtnStop;
1011:        private System.Windows.Forms.ToolStripButton BtnClosePrj;
1012:        private System.Windows.Forms.ToolStripButton BtnPrjSettings;
1013:        private System.Windows.Forms.ToolStripItem MiOpenPrj;
1014:        private System.Windows.Forms.SplitContainer splitContainer1;
1015:        private System.Windows.Forms.TabPage TabMsgIDE;
1016:        public System.Windows.Forms.RichTextBox TxtLogBuild;
1017:        public System.Windows.Forms.RichTextBox TxtLogProgrammer;
1018:        public System.Windows.Forms.TabPage TabBuildLog;
1019:        public System.Windows.Forms.TabPage TabProgLog;
1020:        public System.Windows.Forms.TabControl TabsMsg;
1021:        private System.Windows.Forms.ToolStripStatusLabel StatusLin;
1022:        private System.Windows.Forms.ToolStripStatusLabel StatusProgrammer;
1023:        private SrcTabControl TabsSrc;
1024:        private System.Windows.Forms.TabPage TabDemo;
```

```
1025:         private System.Windows.Forms.ToolStripStatusLabel StatusCol;
1026:         private System.Windows.Forms.ToolStripMenuItem MiFileClose;
1027:         private System.Windows.Forms.ToolStripMenuItem MiTabSave;
1028:         private System.Windows.Forms.ToolStripMenuItem MiTabClose;
1029:         public System.Windows.Forms.ContextMenuStrip MenuTabs;
1030:         private System.Windows.Forms.ToolStripSeparator toolStripSeparator;
1031:         private System.Windows.Forms.ToolStripButton BtnCut;
1032:         private System.Windows.Forms.ToolStripButton BtnCopy;
1033:         private System.Windows.Forms.ToolStripButton BtnPaste;
1034:         private System.Windows.Forms.ToolStripMenuItem MiEdit;
1035:         private System.Windows.Forms.ToolStripMenuItem MiUndo;
1036:         private System.Windows.Forms.ToolStripMenuItem MiRedo;
1037:         private System.Windows.Forms.ToolStripSeparator toolStripSeparator7;
1038:         private System.Windows.Forms.ToolStripMenuItem MiCut;
1039:         private System.Windows.Forms.ToolStripMenuItem MiCopy;
1040:         private System.Windows.Forms.ToolStripMenuItem MiPaste;
1041:         private System.Windows.Forms.ToolStripSeparator toolStripSeparator8;
1042:         private System.Windows.Forms.ToolStripMenuItem MiSelectAll;
1043:         private System.Windows.Forms.ToolStripMenuItem MiPrjSaveAll;
1044:         private System.Windows.Forms.ToolStripButton BtnFileClose;
1045:         private System.Windows.Forms.ToolStripMenuItem MiLanguage;
1046:         private System.Windows.Forms.ToolStripMenuItem MiLangSpanish;
1047:         private System.Windows.Forms.ToolStripMenuItem MiLangEnglish;
1048:         private System.Windows.Forms.ToolStripMenuItem MiPrint;
1049:         private System.Windows.Forms.ToolStripButton BtnSaveAll;
1050:         private System.Windows.Forms.ToolStripSeparator toolStripSeparator4;
1051:         private System.Windows.Forms.ToolStripButton BtnUndo;
1052:         private System.Windows.Forms.ToolStripButton BtnRedo;
1053:         private System.Windows.Forms.ToolStripMenuItem MiPrjBuildBurnRun;
1054:         private System.Windows.Forms.ToolStripButton BtnPrjBuildBurnRun;
1055:         private System.Windows.Forms.ToolStripMenuItem MiCheckMB;
1056:         private System.Windows.Forms.ToolStripMenuItem MiPreferences;
1057:         private System.Windows.Forms.ToolStripMenuItem MiPrintPreview;
1058:     }
1059: }
1060:
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Data;
5: using System.Drawing;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9: using System.Xml.Serialization;
10: using System.IO;
11: using System.Text.RegularExpressions;
12: using System.Drawing.Printing;
13:
14: namespace CuruxaIDE {
15:     public partial class FrmMainWindow:Form {
16:         FrmAbout AboutWindow = new FrmAbout();
17:         Timer TmrHighlight = new Timer();
18:
19:         /// <summary>
20:         /// Indica si el formulario se está cerrando
21:         /// </summary>
22:         bool IsClosing = false;
23:
24:         /// <summary>
25:         /// Cantidad de línea informativas mostradas en el árbol de proyectos bajo cada proyecto, antes de
la lista de archivos de código
26:         /// </summary>
27:         const uint LinesInfoPrjTree = 2;
28:
29:         public FrmMainWindow() {
30:             InitializeComponent();
31:             this.WindowState = Settings.MainWindowState;
32:         }
33:
34:         private void MainWindow_Load(object sender, EventArgs e) {
35:             UpdateLang();
36:             SetTitle("");
37:             LogIDE(i18n.str("AppInitialized"));
38:             if(Project.OpenProjects.Length > 0) Globals.ActiveProject = Project.OpenProjects[0];
```

```
39:         TreePrj.ImageList = new ImageList();
40:         TreePrj.ImageList.Images.Add("Project", Globals.LoadImage("CuruxaLogo_16x16.png"));
41:         TreePrj.ImageList.Images.Add("Source", Globals.LoadImage("edit.png"));
42:         TreePrj.ImageList.Images.Add("Library", Globals.LoadImage("contents.png"));
43:         TreePrj.ImageList.Images.Add("Info", Globals.LoadImage("info.png"));
44:         UpdatePrjList();
45:         MiNewFile.Image = BtnNewFile.Image = Globals.LoadImage("filenew.png");
46:         MiPrjAddFile.Image = BtnAddFilePrj.Image = Globals.LoadImage("fileopen.png");
47:         MiFileSave.Image = BtnSaveFile.Image = Globals.LoadImage("filesave.png");
48:         MiNewPrj.Image = Globals.LoadImage("window_new.png");
49:         MiPrint.Image = Globals.LoadImage("fileprint.png");
50:         MiOpenPrj.Image = Globals.LoadImage("project_open.png");
51:         MiPrjBuild.Image = BtnBuild.Image = Globals.LoadImage("exec.png");
52:         MiPrjProgram.Image = BtnProgramMCU.Image = Globals.LoadImage("bottom.png");
53:         MiPrjRun.Image = BtnRun.Image = Globals.LoadImage("play.png");
54:         MiPrjStop.Image = BtnStop.Image = Globals.LoadImage("stop.png");
55:         MiPrjClose.Image = BtnClosePrj.Image = Globals.LoadImage("cancel.png");
56:         MiPrjSettings.Image = BtnPrjSettings.Image = Globals.LoadImage("configure.png");
57:         MiCut.Image = BtnCut.Image = Globals.LoadImage("editcut.png");
58:         MiCopy.Image = BtnCopy.Image = Globals.LoadImage("editcopy.png");
59:         MiPaste.Image = BtnPaste.Image = Globals.LoadImage("editpaste.png");
60:         MiPrjSaveAll.Image = BtnSaveAll.Image = Globals.LoadImage("save_all.png");
61:         MiFileClose.Image = BtnFileClose.Image = Globals.LoadImage("fileclose.png");
62:         MiPrjBuildBurnRun.Image = BtnPrjBuildBurnRun.Image = Globals.LoadImage("launch.png");
63:         MiOpenExpl.Image = Globals.LoadImage("wizard.png");
64:         MiUndo.Image = BtnUndo.Image = Globals.LoadImage("undo.png");
65:         MiRedo.Image = BtnRedo.Image = Globals.LoadImage("redo.png");
66:         MiAbout.Image = Globals.LoadImage("CuruxaLogo_16x16.png");
67:         MiWebsite.Image = Globals.LoadImage("gohome.png");
68:         MiExit.Image = Globals.LoadImage("exit.png");
69:         TabsSrc.TabPages.Clear();
70:
71:         CheckForIllegalCrossThreadCalls = false;
72:
73:         Globals.CheckVersion();
74:     }
75:
76:     public void UpdateLang() {
77:         MiFile.Text = i18n.str("File");
```



```
78:         MiNewPrj.Text = i18n.str("NewProject");
79:         MiOpenExpl.Text = i18n.str("NewPrjFromEx");
80:         MiFileSave.Text = i18n.str("Save");
81:         MiExit.Text = i18n.str("Exit");
82:         MiNewFile.Text = BtnNewFile.Text = i18n.str("NewFile");
83:         MiPrjAddFile.Text = BtnAddFilePrj.Text = i18n.str("AddExistingFile");
84:         MiPrjBuild.Text = BtnBuild.Text = i18n.str("Build");
85:         MiPrjProgram.Text = BtnProgramMCU.Text = i18n.str("Program");
86:         MiPrjRun.Text = BtnRun.Text = i18n.str("Run");
87:         MiPrjStop.Text = BtnStop.Text = i18n.str("Stop");
88:         MiPrjSettings.Text = i18n.str("Settings");
89:         MiHelp.Text = i18n.str("Help");
90:         MiWebsite.Text = i18n.str("Website");
91:         MiAbout.Text = i18n.str("About");
92:         MiPrjClose.Text = i18n.str("Close");
93:         MiFileClose.Text = BtnFileClose.Text = i18n.str("CloseFile");
94:         MiPrint.Text = i18n.str("MenuPrint");
95:         MiUndo.Text = BtnUndo.Text = i18n.str("MenuUndo");
96:         MiRedo.Text = BtnRedo.Text = i18n.str("MenuRedo");
97:         MiCut.Text = BtnCut.Text = i18n.str("MenuCut");
98:         MiCopy.Text = BtnCopy.Text = i18n.str("MenuCopy");
99:         MiPaste.Text = BtnPaste.Text = i18n.str("MenuPaste");
100:        MiSelectAll.Text = i18n.str("MenuSelectAll");
101:        MiPrjSaveAll.Text = BtnSaveAll.Text = i18n.str("MenuSaveAll");
102:        MiLanguage.Text = i18n.str("MenuLanguage");
103:        BtnSaveFile.Text = i18n.str("SaveFile");
104:        BtnClosePrj.Text = i18n.str("ClosePrj");
105:        BtnPrjSettings.Text = i18n.str("PrjSettings");
106:        MiOpenPrj.Text = i18n.str("OpenPrj");
107:        MiCheckMB.Text = i18n.str("CheckMB");
108:        UpdateActivePrj(Globals.ActiveProject);
109:        TabMsgIDE.Text = i18n.str("IdeLog");
110:        TabBuildLog.Text = i18n.str("BuildLog");
111:        TabProgLog.Text = i18n.str("ProgLog");
112:        MiEdit.Text = i18n.str("MenuEdit");
113:        MiPrjBuildBurnRun.Text = BtnPrjBuildBurnRun.Text = i18n.str("BuildBurnRun");
114:        CursorLocationChanged(new CursorLocation(0, 0));
115:        StatusProgrammer.Text = "";
116:        MiPreferences.Text = i18n.str("Preferences");
```

```
117:             MiPrintPreview.Text = i18n.str("PrintPreview");
118:         }
119:
120:         string FormatLogText(string Text) {
121:             return string.Format("[{0:00}:{1:00}:{2:00}] {3}\n", DateTime.Now.Hour, DateTime.Now.Minute,
e, DateTime.Now.Second, Text);
122:         }
123:
124:         /// <summary>
125:         /// Muestra un texto en el log de Curuxa IDE
126:         /// </summary>
127:         public void LogIDE(string Text) {
128:             LogIDE(Text, true);
129:         }
130:
131:         /// <summary>
132:         /// Muestra un texto en el log de Curuxa IDE
133:         /// </summary>
134:         public void LogIDE(string Text, bool LogDebug) {
135:             if(LogDebug) Globals.Debug("[IDE] " + Text);
136:
137:             //evita textos nulos, vacíos o que sólo contengan espacios
138:             if(string.IsNullOrEmpty(Text)) return;
139:             for(int i = 0; i <= Text.Length; i++) {
140:                 if(i == Text.Length) return;
141:                 if(Text[i] != ' ') break;
142:             }
143:
144:             if(IsClosing) return;
145:
146:             TxtLogIDE.AppendText(FormatLogText(Text));
147:             TxtLogIDE.SelectionStart = TxtLogIDE.Text.Length - 1;
148:             TxtLogIDE.ScrollToCaret();
149:             LblStatus.Text = Text;
150:         }
151:
152:         /// <summary>
153:         /// Muestra un texto en el log de Curuxa IDE
154:         /// </summary>
```

```
155:         public void LogIDE(string Text, params object[] p) {
156:             LogIDE(string.Format(Text, p));
157:         }
158:
159:         bool LogBuildLocker = false;
160:
161:         /// <summary>
162:         /// Muestra un texto en el log de compilación
163:         /// </summary>
164:         public void LogBuild(string Text) {
165:             Globals.Debug("[Build] " + Text);
166:
167:             //evita textos nulos, vacíos o que sólo contengan espacios
168:             if(string.IsNullOrEmpty(Text)) return;
169:             for(int i = 0; i <= Text.Length; i++) {
170:                 if(i == Text.Length) return;
171:                 if(Text[i] != ' ') break;
172:             }
173:
174:             if(IsClosing) return;
175:
176:             TxtLogBuild.AppendText(FormatLogText(Text));
177:             TxtLogBuild.SelectionStart = TxtLogIDE.Text.Length - 1;
178:             TxtLogBuild.ScrollToCaret();
179:         }
180:
181:         /// <summary>
182:         /// Muestra un texto en el log del programador
183:         /// </summary>
184:         public void LogProgrammer(string Text, params object[] p) {
185:             LogProgrammer(string.Format(Text, p));
186:         }
187:
188:         /// <summary>
189:         /// Muestra un texto en el log del programador
190:         /// </summary>
191:         public void LogProgrammer(string Text) {
192:             Globals.Debug("[Programmer] " + Text);
193:
```

```

194:                //evita textos nulos, vacíos o que sólo contengan espacios
195:                if(string.IsNullOrEmpty(Text)) return;
196:                for(int i = 0; i <= Text.Length; i++) {
197:                    if(i == Text.Length) return;
198:                    if(Text[i] != ' ') break;
199:                }
200:
201:                if(IsClosing) return;
202:
203:                TxtLogProgrammer.AppendText(FormatLogText(Text));
204:                TxtLogProgrammer.SelectionStart = TxtLogIDE.Text.Length - 1;
205:                TxtLogProgrammer.ScrollToCaret();
206:            }
207:
208:            /// <summary>
209:            /// Muestra un texto en el log del compilador
210:            /// </summary>
211:            public void LogBuild(string Text, params object[] p) {
212:                LogBuild(string.Format(Text, p));
213:            }
214:
215:            /// <summary>
216:            /// Configura el título de la ventana
217:            /// </summary>
218:            /// <param name="Text"></param>
219:            public void SetTitle(string Text) {
220:                string tail = "Curuxa IDE";
221:                if(string.IsNullOrEmpty(Text)) this.Text = tail;
222:                else this.Text = Text + " || " + tail;
223:            }
224:
225:            /// <summary>
226:            /// Actualiza la lista de proyectos
227:            /// </summary>
228:            ///NOTA PARA DESARROLLADORES: si ponemos fuentes comunes por defecto en el treeview y luego forzamos al nodo padre a que sea negrita, el texto no se mostrará correctamente
229:            public void UpdatePrjList() {
230:                if(IsClosing) return;
231:

```

```
232:         Font ChildrenFont = new Font(TreePrj.Font, FontStyle.Regular);
233:
234:         TreeNode SelectedNode = TreePrj.SelectedNode;
235:         TreePrj.Nodes.Clear();
236:         TreePrj.BackColor = Settings.PrjListBackColor;
237:         foreach(Project prj in Project.OpenProjects) {
238:             TreeNode Parent = new TreeNode(prj.Name);
239:             Parent.ToolTipText = prj.Description;
240:             Parent.ImageKey = Parent.SelectedImageKey = "Project";
241:
242:             //Placa Principal
243:             TreeNode TnMB = new TreeNode(prj.MainBoard.ToString());
244:             TnMB.NodeFont = ChildrenFont;
245:             TnMB.ForeColor = Settings.PrjListSettingsColor;
246:             TnMB.ToolTipText = i18n.str("PrjTreeTooltipMB", prj.MainBoard, prj.MainBoard.GetMC
U());
247:             TnMB.ImageKey = TnMB.SelectedImageKey = "Info";
248:             Parent.Nodes.Add(TnMB);
249:
250:             //Lenguaje de programación
251:             TreeNode TnLang = new TreeNode(i18n.str("Language:") + " " + prj.Language.ToString
());
252:             TnLang.NodeFont = ChildrenFont;
253:             TnLang.ForeColor = Settings.PrjListSettingsColor;
254:             TnLang.ToolTipText = i18n.str("PrjTreeTooltipLang", prj.Language);
255:             TnLang.ImageKey = TnLang.SelectedImageKey = "Info";
256:             Parent.Nodes.Add(TnLang);
257:
258:             //AVISO: si añades más líneas informativas aquí, acuérdate de actualizar la consta
nte "LinesInfoPrjTree"
259:
260:             //archivos de código fuente
261:             foreach(SrcFile src in prj.SrcFiles) {
262:                 //TreeNode TnSrc = new TreeNode(src.FileName + ((src.Modified) ? " *" : ""));
263:                 TreeNode TnSrc = new TreeNode(src.FileName);
264:                 TnSrc.NodeFont = ChildrenFont;
265:                 TnSrc.ForeColor = Settings.PrjListSrcColor;
266:                 TnSrc.ImageKey = TnSrc.SelectedImageKey = "Source";
```

```

267:         TnSrc.ToolTipText = src.FullPath;
268:         Parent.Nodes.Add(TnSrc);
269:     }
270:
271:     //librerías
272:     foreach(SrcFile lib in prj.Libs) {
273:         TreeNode TnLib = new TreeNode(lib.FileName);
274:         TnLib.NodeFont = ChildrenFont;
275:         if(lib.IsReadOnly) TnLib.ForeColor = Settings.PrjListLibsColor;
276:         else TnLib.ForeColor = Settings.PrjListSrcColor;
277:         TnLib.ImageKey = TnLib.SelectedImageKey = "Library";
278:         TnLib.ToolTipText = lib.FullPath;
279:         Parent.Nodes.Add(TnLib);
280:     }
281:
282:     TreePrj.Nodes.Add(Parent);
283: }
284: TreePrj.ExpandAll();
285: if(Project.OpenProjects.Length > 0) {
286:     TreePrj.Enabled = true;
287:     if(SelectedNode != null) {
288:         //select previous node
289:         switch(SelectedNode.Level) {
290:             case 0:
291:                 TreePrj.SelectedNode = TreePrj.Nodes[SelectedNode.Text];
292:                 break;
293:             case 1:
294:                 if(TreePrj.Nodes.ContainsKey(SelectedNode.Parent.Text)) {
295:                     if(TreePrj.Nodes[SelectedNode.Parent.Text].Nodes.ContainsKey(SelectedNode.Text)) TreePrj.SelectedNode = TreePrj.Nodes[SelectedNode.Parent.Text].Nodes[SelectedNode.Text];
296:                     else if(TreePrj.Nodes[SelectedNode.Parent.Text].Nodes.ContainsKey(SelectedNode.Text.Replace(" *", ""))) TreePrj.SelectedNode = TreePrj.Nodes[SelectedNode.Parent.Text].Nodes[SelectedNode.Text.Replace(" *", "")];
297:                     else TreePrj.SelectedNode = TreePrj.Nodes[0];
298:                 } else TreePrj.SelectedNode = TreePrj.Nodes[0];
299:                 break;
300:             default:
301:                 throw new NotImplementedException();
302:         }

```

```
303:                } else {
304:                    //seleccionar primer nodo
305:                    TreePrj.SelectedNode = TreePrj.Nodes[0];
306:                }
307:            } else {
308:                TreePrj.Enabled = false;
309:                TreePrj.Nodes.Add(i18n.str("NoPrjAvail"));
310:            }
311:        }
312:
313:        private void MiAbout_Click(object sender, EventArgs e) {
314:            AboutWindow.ShowDialog(this);
315:        }
316:
317:        private void MiExit_Click(object sender, EventArgs e) {
318:            this.Close();
319:        }
320:
321:        private void MiNewPrj_Click(object sender, EventArgs e) {
322:            Project NewProject = new Project();
323:
324:            //Guardar Proyecto como...
325:            SaveFileDialog SaveProjectDialog = new SaveFileDialog();
326:            SaveProjectDialog.AddExtension = true;
327:            SaveProjectDialog.FileName = i18n.str("NewPrjName");
328:            SaveProjectDialog.Filter = i18n.str("CrxPrjFilter");
329:            SaveProjectDialog.OverwritePrompt = true;
330:            SaveProjectDialog.RestoreDirectory = false;
331:            SaveProjectDialog.ShowHelp = false;
332:            SaveProjectDialog.Title = i18n.str("SavePrjAs");
333:            SaveProjectDialog.ValidateNames = true;
334:            if(SaveProjectDialog.ShowDialog(this) == DialogResult.Cancel) return;
335:            NewProject.PrjFilePath = SaveProjectDialog.FileName;
336:
337:            //Opciones del proyecto
338:            FrmProjectSettings SettingsWindow = new FrmProjectSettings(NewProject);
339:            if(SettingsWindow.ShowDialog(this) == DialogResult.Cancel) return;
340:
341:            FrmFileName FN = new FrmFileName(i18n.str("NewMainFileName") + "." + NewProject.Language.G
```

```
etExtension());
342:         FN.ShowInTaskbar = true;
343:         if(FN.ShowDialog() == DialogResult.Cancel) return;
344:         string FileName = FN.TxtFileName.Text;
345:         if(string.IsNullOrEmpty(FileName)) FileName = "Undefined";
346:         if(!FileName.EndsWith(".") + NewProject.Language.GetExtension())) FileName += "." + NewProj
ect.Language.GetExtension();
347:         SrcFile MainFile = SrcFile.GetFromTemplate(NewProject, FileName, NewProject.MainBoard, New
Project.Language);
348:         NewProject.AddSrcFile(MainFile);
349:
350:         //Añadir a la lista de proyectos abiertos
351:         Project.Add(NewProject);
352:         NewProject.SaveAll();
353:         MainFile.SaveFile();
354:         Globals.LogIDE(i18n.str("NewEmptyPrj", NewProject.Path));
355:     }
356:
357:     private void MiOpenExpl_Click(object sender, EventArgs e) {
358:         FrmNewPrjFromExample win = new FrmNewPrjFromExample();
359:         if(win.ShowDialog(this) == DialogResult.OK) {
360:             Example ex = win.ChosenExample;
361:             Globals.LogIDE(i18n.str("GeneratingExample", ex.Project.Name));
362:
363:             //cargar contenidos a memoria para que luego puedan ser copiados
364:             string temp = "";
365:             foreach(SrcFile src in ex.Project.SrcFiles) temp = src.Content;
366:
367:             Project NewPrj = ex.Project;
368:             NewPrj.PrjFilePath = win.TxtPrjFile.Text;
369:             NewPrj.SaveAll();
370:             Project.Add(NewPrj);
371:         }
372:     }
373:
374:     private void MiOpenPrj_Click(object sender, EventArgs e) {
375:         //Abrir Proyecto...
376:         OpenFileDialog OpenProjectDialog = new OpenFileDialog();
377:         OpenProjectDialog.AddExtension = true;
```



```
378:         OpenProjectDialog.Filter = i18n.str("CrxPrjFilter");
379:         OpenProjectDialog.RestoreDirectory = false;
380:         OpenProjectDialog.ShowHelp = false;
381:         OpenProjectDialog.Title = i18n.str("OpenPrj");
382:         OpenProjectDialog.ValidateNames = true;
383:         OpenProjectDialog.ShowDialog(this);
384:
385:         string PrjFile = OpenProjectDialog.FileName;
386:         Globals.LogIDE(i18n.str("OpeningPrj", PrjFile));
387:         Project.Open(PrjFile);
388:     }
389:
390:     private void MiWebsite_Click(object sender, EventArgs e) {
391:         System.Diagnostics.Process.Start("http://curuxa.org/");
392:     }
393:
394:     private void FrmMainWindow_FormClosing(object sender, FormClosingEventArgs e) {
395:         IsClosing = true;
396:         foreach(Project prj in Project.OpenProjects) {
397:             CheckSavePrj(prj);
398:         }
399:         Settings.Save();
400:     }
401:
402:     private void TreePrj_NodeMouseClick(object sender, TreeNodeMouseClickEventArgs e) {
403:         if(e.Node.Level == 0) {
404:             Globals.ActiveProject = Project.OpenProjects.GetByName(e.Node.Text);
405:         } else if(e.Node.Level == 1) {
406:             Globals.ActiveProject = Project.OpenProjects.GetByName(e.Node.Parent.Text);
407:         }
408:     }
409:
410:     private void TreePrj_NodeMouseDoubleClick(object sender, TreeNodeMouseClickEventArgs e) {
411:         if(e.Node.Level == 1) {
412:             //un archivo acaba de ser seleccionado
413:             Globals.ActiveProject = Project.OpenProjects.GetByName(e.Node.Parent.Text);
414:
415:             //doble click en archivos de código fuente
416:             if(e.Node.Index >= LinesInfoPrjTree) {
```

[illegible]

```
452:                                     if(TabsSrc.Tabs.ContainsKey(Globals.ActiveSrcFile)) TabsSrc.Tabs[Globals.ActiveSrc ↵
File].UpdateTitle();
453:                                     }
454:                                 }
455:
456:         private void MiPrjClose_Click(object sender, EventArgs e) {
457:             Project p = Globals.ActiveProject;
458:             CheckSavePrj(p);
459:             TabsSrc.RemoveAllFromPrj(p);
460:             p.Close();
461:         }
462:
463:         /// <summary>
464:         /// Comprueba si un proyecto y su código fuente necesita ser guardado, y le pregunta al usuario
465:         /// </summary>
466:         /// <param name="p"></param>
467:         private void CheckSavePrj(Project p) {
468:             //guardando archivos
469:             foreach(SrcFile src in p.SrcFiles) {
470:                 if(src.Modified) {
471:                     DialogResult srcresult = MessageBox.Show(this, i18n.str("WantSaveFilePrj", ↵
src.FileName, p.Name), i18n.str("FileNotSaved"), MessageBoxButtons.YesNoCancel, MessageBoxIcon.Question, MessageBoxDefau ↵
ltButton.Button1);
472:                     if(srcresult == DialogResult.Yes) src.SaveFile();
473:                     if(srcresult == DialogResult.Cancel) return;
474:                 }
475:             }
476:
477:             //guardando proyecto
478:             p.Save();
479:         }
480:
481:         private void BtnNewFile_Click(object sender, EventArgs e) {
482:             MiNewFile.PerformClick();
483:         }
484:
485:         private void BtnAddFilePrj_Click(object sender, EventArgs e) {
486:             MiPrjAddFile.PerformClick();
487:         }
```

```
488:
489:     private void BtnSaveFile_Click(object sender, EventArgs e) {
490:         MiFileSave.PerformClick();
491:     }
492:
493:     private void BtnBuild_Click(object sender, EventArgs e) {
494:         MiPrjBuild.PerformClick();
495:     }
496:
497:     private void BtnRun_Click(object sender, EventArgs e) {
498:         MiPrjRun.PerformClick();
499:     }
500:
501:     private void BtnStop_Click(object sender, EventArgs e) {
502:         MiPrjStop.PerformClick();
503:     }
504:
505:     private void BtnClosePrj_Click(object sender, EventArgs e) {
506:         MiPrjClose.PerformClick();
507:     }
508:
509:     private void BtnConfigPrj_Click(object sender, EventArgs e) {
510:         MiPrjSettings.PerformClick();
511:     }
512:
513:     private void MiBuildPrj_Click(object sender, EventArgs e) {
514:         if(Globals.ActiveProject == null) {
515:             LogIDE(i18n.str("NoActivePrj"));
516:         } else {
517:             Globals.ActiveProject.SaveAll();
518:
519:             if(Globals.ActiveProject.Build() == 0) {
520:                 LogIDE(i18n.str("BuildOk"));
521:                 LogBuild(i18n.str("BuildOk"));
522:             } else {
523:                 LogIDE(i18n.str("BuildFail"));
524:                 LogBuild(i18n.str("BuildFail"));
525:             }
526:         }
```

```
527:         }
528:
529:         private void MiPrjSettings_Click(object sender, EventArgs e) {
530:             if(Globals.ActiveProject != null) {
531:                 FrmProjectSettings frm = new FrmProjectSettings(Globals.ActiveProject);
532:                 frm.ShowDialog(this);
533:                 UpdatePrjList();
534:                 Globals.ActiveProject.SaveAll();
535:             }
536:         }
537:
538:         private void MiProgramPrj_Click(object sender, EventArgs e) {
539:             if(Globals.ActiveProject == null) {
540:                 LogIDE(i18n.str("NoActivePrj"));
541:             } else {
542:                 if(Globals.ActiveProject.Burn() == 0) {
543:                     StatusProgrammer.Text = i18n.str("PrjProgrammed");
544:                 }
545:             }
546:         }
547:
548:         private void BtnProgramMCU_Click(object sender, EventArgs e) {
549:             MiPrjProgram.PerformClick();
550:         }
551:
552:         private void MiRunPrj_Click(object sender, EventArgs e) {
553:             if(Globals.ActiveProject == null) {
554:                 LogIDE(i18n.str("NoActivePrj"));
555:             } else {
556:                 if(Globals.ActiveProject.Run() == 0) {
557:                     StatusProgrammer.Text = i18n.str("PrjRun");
558:                 }
559:             }
560:         }
561:
562:         private void MiStopPrj_Click(object sender, EventArgs e) {
563:             if(Globals.ActiveProject == null) {
564:                 LogIDE(i18n.str("NoActivePrj"));
565:             } else {
```

```
566:                 if(Globals.ActiveProject.Stop() == 0) {
567:                     StatusProgrammer.Text = i18n.str("PrjStopped");
568:                 }
569:             }
570:         }
571:
572:         public void CursorLocationChanged(CursorLocation c) {
573:             StatusLin.Text = i18n.str("LineX", c.Line);
574:             StatusCol.Text = i18n.str("ColX", c.Column);
575:         }
576:
577:         private void MiClose_Click(object sender, EventArgs e) {
578:             TabsSrc.CloseSrc(Globals.ActiveSrcFile);
579:         }
580:
581:         private void MiPrjSaveAll_Click(object sender, EventArgs e) {
582:             Globals.ActiveProject.SaveAll();
583:             foreach(KeyValuePair<SrcFile, SrcTabPage> pair in TabsSrc.Tabs) {
584:                 pair.Value.UpdateTitle();
585:             }
586:         }
587:
588:         private void BtnFileClose_Click(object sender, EventArgs e) {
589:             MiFileClose.PerformClick();
590:         }
591:
592:         private void MiLangSpanish_Click(object sender, EventArgs e) {
593:             i18n.CurrentLanguage = "es";
594:             UpdateLang();
595:         }
596:
597:         private void MiLangEnglish_Click(object sender, EventArgs e) {
598:             i18n.CurrentLanguage = "en";
599:             UpdateLang();
600:         }
601:
602:         private void BtnUndo_Click(object sender, EventArgs e) {
603:             MiUndo.PerformClick();
604:         }
```

```
605:
606:         private void MiUndo_Click(object sender, EventArgs e) {
607:             if(TabsSrc.SelectedTab != null) {
608:                 //(TabsSrc.SelectedTab as SrcTabPage).TxtCode.Undo(); NO FUNCIONA CON RESALTO DE S
609:             }
610:         }
611:
612:         private void MiRedo_Click(object sender, EventArgs e) {
613:             if(TabsSrc.SelectedTab != null) {
614:                 //(TabsSrc.SelectedTab as SrcTabPage).TxtCode.Redo(); NO FUNCIONA CON RESALTO DE S
615:             }
616:         }
617:
618:         private void BtnRedo_Click(object sender, EventArgs e) {
619:             MiRedo.PerformClick();
620:         }
621:
622:         private void BtnSaveAll_Click(object sender, EventArgs e) {
623:             MiPrjSaveAll.PerformClick();
624:         }
625:
626:         private void BtnCut_Click(object sender, EventArgs e) {
627:             MiCut.PerformClick();
628:         }
629:
630:         private void MiCut_Click(object sender, EventArgs e) {
631:             if(TabsSrc.SelectedTab != null) {
632:                 (TabsSrc.SelectedTab as SrcTabPage).TxtCode.Cut();
633:             }
634:         }
635:
636:         private void BtnCopy_Click(object sender, EventArgs e) {
637:             MiCopy.PerformClick();
638:         }
639:
640:         private void MiCopy_Click(object sender, EventArgs e) {
641:             if(TabsSrc.SelectedTab != null) {
```

```
642:                (TabsSrc.SelectedTab as SrcTabPage).TxtCode.Copy();
643:            }
644:        }
645:
646:        private void BtnPaste_Click(object sender, EventArgs e) {
647:            MiPaste.PerformClick();
648:        }
649:
650:        private void MiPaste_Click(object sender, EventArgs e) {
651:            if(TabsSrc.SelectedTab != null) {
652:                (TabsSrc.SelectedTab as SrcTabPage).TxtCode.Paste();
653:            }
654:        }
655:
656:        private void BtnBuildBurnRun_Click(object sender, EventArgs e) {
657:            MiPrjBuildBurnRun.PerformClick();
658:        }
659:
660:        private void MiBuildBurnRun_Click(object sender, EventArgs e) {
661:            Project p = Globals.ActiveProject;
662:            if(p != null) {
663:                MiPrjBuild.PerformClick();
664:                if(p.IsCompiled) MiPrjProgram.PerformClick();
665:                if(p.IsBurnt) MiPrjRun.PerformClick();
666:            }
667:        }
668:
669:        private void MiNewFile_Click(object sender, EventArgs e) {
670:        }
671:
672:
673:        private void MiPrjAddFile_Click(object sender, EventArgs e) {
674:
675:        }
676:
677:        private void MiSelectAll_Click(object sender, EventArgs e) {
678:            if(TabsSrc.SelectedTab != null) {
679:                (TabsSrc.SelectedTab as SrcTabPage).TxtCode.SelectAll();
680:            }
```



```
681:         }
682:
683:         private void MiCheckMB_Click(object sender, EventArgs e) {
684:             Microcontroller? DetectedMCU = Globals.ActiveProject.Programmer.GetProgrammerApp().Detect(
);
685:             TabsMsg.SelectedTab = TabMsgIDE;
686:             if(!DetectedMCU.HasValue) Globals.LogIDE(i18n.str("NoMcuDetected"));
687:             else {
688:                 Globals.LogIDE(i18n.str("DetectedX", DetectedMCU.Value, DetectedMCU.Value.GetMainBoard()));
689:             }
690:         }
691:
692:         #region printing
693:         string[] lines;
694:         private PrintDialog printDialog1;
695:         private PrintDocument printDocument1;
696:         private PageSetupDialog pageSetupDialog1;
697:         private PrintPreviewDialog printPreviewDialog1;
698:         bool PrintSetup = false;
699:
700:         private void SetupPrint() {
701:             printDialog1 = new PrintDialog();
702:             printDocument1 = new PrintDocument();
703:             pageSetupDialog1 = new PageSetupDialog();
704:
705:             printDialog1.Document = this.printDocument1;
706:             printDocument1.BeginPrint += new PrintEventHandler(OnBeginPrint);
707:             printDocument1.PrintPage += new PrintPageEventHandler(OnPrintPage);
708:             pageSetupDialog1.Document = this.printDocument1;
709:
710:             printPreviewDialog1 = new PrintPreviewDialog();
711:
712:             printPreviewDialog1.AutoScrollMargin = new System.Drawing.Size(0, 0);
713:             printPreviewDialog1.AutoScrollMinSize = new System.Drawing.Size(0, 0);
714:             printPreviewDialog1.ClientSize = new System.Drawing.Size(400, 300);
715:             printPreviewDialog1.Document = this.printDocument1;
716:             printPreviewDialog1.Enabled = true;
717:             printPreviewDialog1.Location = new System.Drawing.Point(88, 116);
```

```
718:         printPreviewDialog1.MaximumSize = new System.Drawing.Size(0, 0);
719:         printPreviewDialog1.Name = i18n.str("PrintPreview");
720:         printPreviewDialog1.Opacity = 1;
721:         printPreviewDialog1.TransparencyKey = System.Drawing.Color.Empty;
722:         printPreviewDialog1.Visible = false;
723:
724:         PrintSetup = true;
725:     }
726:
727:     private void MiPrint_Click(object sender, EventArgs e) {
728:         if(TabsSrc.SelectedTab == null) {
729:             LogIDE(i18n.str("NoSrcSelected"));
730:             return;
731:         }
732:
733:         if(!PrintSetup) SetupPrint();
734:
735:         if(printDialog1.ShowDialog() == DialogResult.OK) {
736:             printDocument1.Print();
737:         }
738:     }
739:
740:     private void OnBeginPrint(object sender, System.Drawing.Printing.PrintEventArgs e) {
741:         char[] param = { '\n' };
742:
743:         if(printDialog1.PrinterSettings.PrintRange == PrintRange.Selection) {
744:             lines = ((SrcTabPage)TabsSrc.SelectedTab).TxtCode.SelectedText.Split(param);
745:         } else {
746:             lines = ((SrcTabPage)TabsSrc.SelectedTab).TxtCode.Text.Split(param);
747:         }
748:
749:         int i = 0;
750:         char[] trimParam = { '\r' };
751:         foreach(string s in lines) {
752:             lines[i++] = s.TrimEnd(trimParam);
753:         }
754:     }
755:
756:     private void OnPrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e) {
```

```
757:         int linesPrinted = 0;
758:         int x = e.MarginBounds.Left;
759:         int y = e.MarginBounds.Top;
760:         Brush brush = new SolidBrush(((SrcTabPage)TabsSrc.SelectedTab).TxtCode.ForeColor);
761:
762:         while(linesPrinted < lines.Length) {
763:             e.Graphics.DrawString(lines[linesPrinted++], ((SrcTabPage)TabsSrc.SelectedTab).Txt
Code.Font, brush, x, y);
764:             y += 15;
765:             if(y >= e.MarginBounds.Bottom) {
766:                 e.HasMorePages = true;
767:                 return;
768:             }
769:         }
770:
771:         linesPrinted = 0;
772:         e.HasMorePages = false;
773:     }
774:
775:     private void MiPrintPreview_Click(object sender, EventArgs e) {
776:         if(TabsSrc.SelectedTab == null) {
777:             LogIDE(i18n.str("NoSrcSelected"));
778:             return;
779:         }
780:
781:         if(!PrintSetup) SetupPrint();
782:
783:         printPreviewDialog1.ShowDialog();
784:     }
785: #endregion
786:
787:     private void MiPreferences_Click(object sender, EventArgs e) {
788:         FrmIdeSettings frm = new FrmIdeSettings();
789:         frm.ShowDialog(this);
790:     }
791: }
792: }
```

```

1: // Archivo generado automáticamente por Visual Studio
2:
3: namespace CuruxaIDE {
4:     partial class FrmNewPrjFromExample {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
>
14:        protected override void Dispose(bool disposing) {
15:            if(disposing && (components != null)) {
16:                components.Dispose();
17:            }
18:            base.Dispose(disposing);
19:        }
20:
21:        #region Windows Form Designer generated code
22:
23:        /// <summary>
24:        /// Required method for Designer support - do not modify
25:        /// the contents of this method with the code editor.
26:        /// </summary>
27:        private void InitializeComponent() {
28:            System.Windows.Forms.ListViewItem listViewItem1 = new System.Windows.Forms.ListViewItem("t
est");
29:            System.Windows.Forms.ListViewItem listViewItem2 = new System.Windows.Forms.ListViewItem("a
nother test");
30:            System.Windows.Forms.ListViewItem listViewItem3 = new System.Windows.Forms.ListViewItem("d
on't remove these tests!");
31:            System.Windows.Forms.ListViewItem listViewItem4 = new System.Windows.Forms.ListViewItem("t
est");
32:            System.Windows.Forms.ListViewItem listViewItem5 = new System.Windows.Forms.ListViewItem("a
nother test");
33:            System.Windows.Forms.ListViewItem listViewItem6 = new System.Windows.Forms.ListViewItem("d

```

```
on\'t remove these tests!");
34:         this.LstExamples = new System.Windows.Forms.ListBox();
35:         this.LblAvailEx = new System.Windows.Forms.Label();
36:         this.TxtDescription = new System.Windows.Forms.TextBox();
37:         this.tableLayoutPanel1 = new System.Windows.Forms.TableLayoutPanel();
38:         this.LblSaveAs = new System.Windows.Forms.Label();
39:         this.TxtPrjFile = new System.Windows.Forms.TextBox();
40:         this.BtnSaveAs = new System.Windows.Forms.Button();
41:         this.BtnSave = new System.Windows.Forms.Button();
42:         this.LstFilterMB = new System.Windows.Forms.ListView();
43:         this.LblFilter = new System.Windows.Forms.Label();
44:         this.LblMainBoard = new System.Windows.Forms.Label();
45:         this.LblModules = new System.Windows.Forms.Label();
46:         this.LstFilterModules = new System.Windows.Forms.ListView();
47:         this.tableLayoutPanel1.SuspendLayout();
48:         this.SuspendLayout();
49:         //
50:         // LstExamples
51:         //
52:         this.LstExamples.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)
53:                                     | System.Windows.Forms.AnchorStyles.Left));
54:         this.LstExamples.FormattingEnabled = true;
55:         this.LstExamples.Location = new System.Drawing.Point(12, 25);
56:         this.LstExamples.Name = "LstExamples";
57:         this.LstExamples.Size = new System.Drawing.Size(186, 212);
58:         this.LstExamples.TabIndex = 0;
59:         this.LstExamples.SelectedValueChanged += new System.EventHandler(this.LstExamples_SelectedValue
ValueChanged);
60:         //
61:         // LblAvailEx
62:         //
63:         this.LblAvailEx.AutoSize = true;
64:         this.LblAvailEx.Location = new System.Drawing.Point(12, 9);
65:         this.LblAvailEx.Name = "LblAvailEx";
66:         this.LblAvailEx.Size = new System.Drawing.Size(86, 13);
67:         this.LblAvailEx.TabIndex = 1;
68:         this.LblAvailEx.Text = "NS (List avail prj)";
69:         //
```

```
70:                // TxtDescription
71:                //
72:                this.TxtDescription.Anchor = ((System.Windows.Forms.AnchorStyles)(System.Windows.Forms.
AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)
73:                | System.Windows.Forms.AnchorStyles.Left)
74:                | System.Windows.Forms.AnchorStyles.Right));
75:                this.TxtDescription.BackColor = System.Drawing.SystemColors.Window;
76:                this.TxtDescription.Location = new System.Drawing.Point(210, 25);
77:                this.TxtDescription.Multiline = true;
78:                this.TxtDescription.Name = "TxtDescription";
79:                this.TxtDescription.ReadOnly = true;
80:                this.TxtDescription.Size = new System.Drawing.Size(570, 497);
81:                this.TxtDescription.TabIndex = 0;
82:                //
83:                // tableLayoutPanel1
84:                //
85:                this.tableLayoutPanel1.Anchor = ((System.Windows.Forms.AnchorStyles)(System.Windows.Forms.
s.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Left)
86:                | System.Windows.Forms.AnchorStyles.Right));
87:                this.tableLayoutPanel1.ColumnCount = 3;
88:                this.tableLayoutPanel1.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle());
89:                this.tableLayoutPanel1.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.
s.Forms.SizeType.Percent, 100F));
90:                this.tableLayoutPanel1.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.
s.Forms.SizeType.Absolute, 39F));
91:                this.tableLayoutPanel1.Controls.Add(this.LblSaveAs, 0, 0);
92:                this.tableLayoutPanel1.Controls.Add(this.TxtPrjFile, 1, 0);
93:                this.tableLayoutPanel1.Controls.Add(this.BtnSaveAs, 2, 0);
94:                this.tableLayoutPanel1.Location = new System.Drawing.Point(210, 528);
95:                this.tableLayoutPanel1.Name = "tableLayoutPanel1";
96:                this.tableLayoutPanel1.RowCount = 1;
97:                this.tableLayoutPanel1.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.
s.SizeType.Percent, 100F));
98:                this.tableLayoutPanel1.Size = new System.Drawing.Size(470, 35);
99:                this.tableLayoutPanel1.TabIndex = 4;
100:                //
101:                // LblSaveAs
102:                //
103:                this.LblSaveAs.AutoSize = true;
```

```
104:         this.LblSaveAs.Dock = System.Windows.Forms.DockStyle.Fill;
105:         this.LblSaveAs.Location = new System.Drawing.Point(3, 0);
106:         this.LblSaveAs.Name = "LblSaveAs";
107:         this.LblSaveAs.Size = new System.Drawing.Size(84, 35);
108:         this.LblSaveAs.TabIndex = 0;
109:         this.LblSaveAs.Text = "NS (Save prj as)";
110:         this.LblSaveAs.TextAlign = System.Drawing.ContentAlignment.MiddleRight;
111:         //
112:         // TxtPrjFile
113:         //
114:         this.TxtPrjFile.BackColor = System.Drawing.SystemColors.Window;
115:         this.TxtPrjFile.Dock = System.Windows.Forms.DockStyle.Fill;
116:         this.TxtPrjFile.Location = new System.Drawing.Point(93, 8);
117:         this.TxtPrjFile.Margin = new System.Windows.Forms.Padding(3, 8, 3, 3);
118:         this.TxtPrjFile.Name = "TxtPrjFile";
119:         this.TxtPrjFile.ReadOnly = true;
120:         this.TxtPrjFile.Size = new System.Drawing.Size(335, 20);
121:         this.TxtPrjFile.TabIndex = 1;
122:         //
123:         // BtnSaveAs
124:         //
125:         this.BtnSaveAs.Dock = System.Windows.Forms.DockStyle.Fill;
126:         this.BtnSaveAs.Location = new System.Drawing.Point(434, 3);
127:         this.BtnSaveAs.Name = "BtnSaveAs";
128:         this.BtnSaveAs.Size = new System.Drawing.Size(33, 29);
129:         this.BtnSaveAs.TabIndex = 2;
130:         this.BtnSaveAs.UseVisualStyleBackColor = true;
131:         this.BtnSaveAs.Click += new System.EventHandler(this.BtnSaveAs_Click);
132:         //
133:         // BtnSave
134:         //
135:         this.BtnSave.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Right)));
136:         this.BtnSave.Location = new System.Drawing.Point(705, 534);
137:         this.BtnSave.Name = "BtnSave";
138:         this.BtnSave.Size = new System.Drawing.Size(75, 23);
139:         this.BtnSave.TabIndex = 5;
140:         this.BtnSave.Text = "NS (Save)";
141:         this.BtnSave.UseVisualStyleBackColor = true;
```

```
142:         this.BtnSave.Click += new System.EventHandler(this.BtnSave_Click);
143:         //
144:         // LstFilterMB
145:         //
146:         this.LstFilterMB.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Left)));
147:         this.LstFilterMB.CheckBoxes = true;
148:         this.LstFilterMB.FullRowSelect = true;
149:         listViewItem1.StateImageIndex = 0;
150:         listViewItem2.StateImageIndex = 0;
151:         listViewItem3.StateImageIndex = 0;
152:         this.LstFilterMB.Items.AddRange(new System.Windows.Forms.ListViewItem[] {
153:             listViewItem1,
154:             listViewItem2,
155:             listViewItem3});
156:         this.LstFilterMB.Location = new System.Drawing.Point(12, 295);
157:         this.LstFilterMB.MultiSelect = false;
158:         this.LstFilterMB.Name = "LstFilterMB";
159:         this.LstFilterMB.Size = new System.Drawing.Size(186, 94);
160:         this.LstFilterMB.TabIndex = 6;
161:         this.LstFilterMB.UseCompatibleStateImageBehavior = false;
162:         this.LstFilterMB.View = System.Windows.Forms.View.List;
163:         this.LstFilterMB.ItemChecked += new System.Windows.Forms.ItemCheckedEventHandler(this.LstFilterMB_ItemChecked);
164:         //
165:         // LblFilter
166:         //
167:         this.LblFilter.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Left)));
168:         this.LblFilter.AutoSize = true;
169:         this.LblFilter.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
170:         this.LblFilter.Location = new System.Drawing.Point(12, 252);
171:         this.LblFilter.Name = "LblFilter";
172:         this.LblFilter.Size = new System.Drawing.Size(83, 17);
173:         this.LblFilter.TabIndex = 7;
174:         this.LblFilter.Text = "NS (Filter)";
175:         //
176:         // LblMainBoard
```



```
177:                //
178:                this.LblMainBoard.Anchor = ((System.Windows.Forms.AnchorStyles) (System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Left));
179:                this.LblMainBoard.AutoSize = true;
180:                this.LblMainBoard.Location = new System.Drawing.Point(12, 279);
181:                this.LblMainBoard.Name = "LblMainBoard";
182:                this.LblMainBoard.Size = new System.Drawing.Size(85, 13);
183:                this.LblMainBoard.TabIndex = 8;
184:                this.LblMainBoard.Text = "NS (Main Board)";
185:                //
186:                // LblModules
187:                //
188:                this.LblModules.Anchor = ((System.Windows.Forms.AnchorStyles) (System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Left));
189:                this.LblModules.AutoSize = true;
190:                this.LblModules.Location = new System.Drawing.Point(12, 404);
191:                this.LblModules.Name = "LblModules";
192:                this.LblModules.Size = new System.Drawing.Size(71, 13);
193:                this.LblModules.TabIndex = 9;
194:                this.LblModules.Text = "NS (Modules)";
195:                //
196:                // LstFilterModules
197:                //
198:                this.LstFilterModules.Anchor = ((System.Windows.Forms.AnchorStyles) (System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Left));
199:                this.LstFilterModules.CheckBoxes = true;
200:                this.LstFilterModules.FullRowSelect = true;
201:                listViewItem4.StateImageIndex = 0;
202:                listViewItem5.StateImageIndex = 0;
203:                listViewItem6.StateImageIndex = 0;
204:                this.LstFilterModules.Items.AddRange(new System.Windows.Forms.ListViewItem[] {
205:                    listViewItem4,
206:                    listViewItem5,
207:                    listViewItem6});
208:                this.LstFilterModules.Location = new System.Drawing.Point(12, 420);
209:                this.LstFilterModules.MultiSelect = false;
210:                this.LstFilterModules.Name = "LstFilterModules";
211:                this.LstFilterModules.Size = new System.Drawing.Size(186, 143);
212:                this.LstFilterModules.TabIndex = 10;
```

```
213:         this.LstFilterModules.UseCompatibleStateImageBehavior = false;
214:         this.LstFilterModules.View = System.Windows.Forms.View.List;
215:         this.LstFilterModules.ItemChecked += new System.Windows.Forms.ItemCheckedEventHandler(this
.LstFilterModules_ItemChecked);
216:         //
217:         // FrmNewPrjFromExample
218:         //
219:         this.AcceptButton = this.BtnSave;
220:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
221:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
222:         this.ClientSize = new System.Drawing.Size(792, 573);
223:         this.Controls.Add(this.LstFilterModules);
224:         this.Controls.Add(this.LblModules);
225:         this.Controls.Add(this.LblMainBoard);
226:         this.Controls.Add(this.LblFilter);
227:         this.Controls.Add(this.LstFilterMB);
228:         this.Controls.Add(this.BtnSave);
229:         this.Controls.Add(this.tableLayoutPanel1);
230:         this.Controls.Add(this.TxtDescription);
231:         this.Controls.Add(this.LblAvailEx);
232:         this.Controls.Add(this.LstExamples);
233:         this.MinimumSize = new System.Drawing.Size(570, 500);
234:         this.Name = "FrmNewPrjFromExample";
235:         this.Text = "NS (New Project From Example)";
236:         this.Load += new System.EventHandler(this.NewPrjFromExample_Load);
237:         this.tableLayoutPanel1.ResumeLayout(false);
238:         this.tableLayoutPanel1.PerformLayout();
239:         this.ResumeLayout(false);
240:         this.PerformLayout();
241:
242:     }
243:
244:     #endregion
245:
246:     private System.Windows.Forms.ListBox LstExamples;
247:     private System.Windows.Forms.Label LblAvailEx;
248:     private System.Windows.Forms.TextBox TxtDescription;
249:     private System.Windows.Forms.TableLayoutPanel tableLayoutPanel1;
250:     private System.Windows.Forms.Label LblSaveAs;
```

```
251:         public System.Windows.Forms.TextBox TxtPrjFile;
252:         private System.Windows.Forms.Button BtnSaveAs;
253:         private System.Windows.Forms.Button BtnSave;
254:         private System.Windows.Forms.ListView LstFilterMB;
255:         private System.Windows.Forms.Label LblFilter;
256:         private System.Windows.Forms.Label LblMainBoard;
257:         private System.Windows.Forms.Label LblModules;
258:         private System.Windows.Forms.ListView LstFilterModules;
259:     }
260: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Data;
5: using System.Drawing;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9:
10: namespace CuruxaIDE {
11:     public partial class FrmNewPrjFromExample:Form {
12:         /// <summary>
13:         /// Ejemplo seleccionado actualmente. Null si no hay ninguno seleccionado
14:         /// </summary>
15:         public Example ChosenExample;
16:
17:         public FrmNewPrjFromExample() {
18:             InitializeComponent();
19:         }
20:
21:         private void NewPrjFromExample_Load(object sender, EventArgs e) {
22:             BtnSaveAs.Image = Globals.LoadImage("fileopen.png");
23:             UpdateLang();
24:             UpdateExamplesList();
25:
26:             //actualizar listas de filtrado
27:             LstFilterMB.Items.Clear();
28:             foreach(string MB in Enum.GetNames(typeof(MainBoard))) {
29:                 LstFilterMB.Items.Add(MB);
30:             }
31:             LstFilterModules.Items.Clear();
32:             foreach(string Module in Enum.GetNames(typeof(Module))) {
33:                 LstFilterModules.Items.Add((((Module)Enum.Parse(typeof(Module), Module)).GetRealName
me()));
34:             }
35:         }
36:
37:         /// <summary>
38:         /// Actualiza la lista de ejemplos disponibles
```

```
39:         /// </summary>
40:         private void UpdateExamplesList() {
41:             LstExamples.Items.Clear();
42:             foreach (Example ex in Example.AvailableExamples) {
43:                 bool show = true;
44:                 foreach (ListViewItem MbItem in LstFilterMB.Items) {
45:                     if (MbItem.Checked && ex.Project.MainBoard != (MainBoard)Enum.Parse(typeof(MainBoard), MbItem.Text)) show = false;
46:                 }
47:                 foreach (ListViewItem ModItem in LstFilterModules.Items) {
48:                     if (ModItem.Checked) {
49:                         bool ExampleUsesMod = false;
50:                         foreach (Module CompatMod in ModuleExtensions.GetCompatibleModules(ModuleExtensions.GetModuleFromRealName(ModItem.Text))) {
51:                             if (ex.UsedModules.Contains(CompatMod)) ExampleUsesMod = true;
52:                         }
53:                         if (!ExampleUsesMod) show = false;
54:                     }
55:                 }
56:                 if (show) LstExamples.Items.Add(ex.Project.Name);
57:             }
58:             if (LstExamples.Items.Count > 0) {
59:                 LstExamples.Enabled = true;
60:                 LstExamples.SelectedIndex = 0;
61:             } else {
62:                 // no hay ejemplos disponibles
63:                 LstExamples.Enabled = false;
64:                 ChosenExample = null;
65:                 TxtDescription.Enabled = false;
66:                 TxtDescription.Text = i18n.str("NoExample");
67:                 LstExamples.Items.Add(i18n.str("NoExamples"));
68:             }
69:             UpdateBtnStatus();
70:         }
71:
72:         /// <summary>
73:         /// Actualiza el estado del botón "Guardar"
74:         /// </summary>
```

```
75:         private void UpdateBtnStatus() {
76:             if(ChosenExample != null && !string.IsNullOrEmpty(TxtPrjFile.Text)) BtnSave.Enabled = true
;
77:             else BtnSave.Enabled = false;
78:         }
79:
80:         private void UpdateLang() {
81:             this.Text = i18n.str("NewPrjFromEx");
82:             LblAvailEx.Text = i18n.str("ListAvailEx");
83:             LblSaveAs.Text = i18n.str("SavePrjAs:");
84:             BtnSave.Text = i18n.str("SaveExample");
85:             LblFilter.Text = i18n.str("Filter:");
86:             LblMainBoard.Text = i18n.str("MainBoards:");
87:             LblModules.Text = i18n.str("Modules:");
88:         }
89:
90:         private void BtnSaveAs_Click(object sender, EventArgs e) {
91:             SaveFileDialog SaveProjectDialog = new SaveFileDialog();
92:             SaveProjectDialog.AddExtension = true;
93:             SaveProjectDialog.FileName = i18n.str("NewPrjName");
94:             SaveProjectDialog.Filter = i18n.str("CrxPrjFilter");
95:             SaveProjectDialog.OverwritePrompt = true;
96:             SaveProjectDialog.RestoreDirectory = false;
97:             SaveProjectDialog.ShowHelp = false;
98:             SaveProjectDialog.Title = i18n.str("SavePrjAs");
99:             SaveProjectDialog.ValidateNames = true;
100:            if(SaveProjectDialog.ShowDialog(this) == DialogResult.OK) {
101:                TxtPrjFile.Text = SaveProjectDialog.FileName;
102:            }
103:            UpdateBtnStatus();
104:        }
105:
106:        private void LstExamples_SelectedValueChanged(object sender, EventArgs e) {
107:            SelectExample(LstExamples.SelectedItem.ToString());
108:        }
109:
110:        /// <summary>
111:        /// Actualiza el entorno para mostrar que cierto ejemplo se ha seleccionado
112:        /// </summary>
```

```
113:         /// <param name="Name"></param>
114:         private void SelectExample(string Name) {
115:             for(int i = 0; i < Example.AvailableExamples.Length; i++) {
116:                 if(Example.AvailableExamples[i].Project.Name == Name) {
117:                     ChosenExample = Example.AvailableExamples[i];
118:                     break;
119:                 }
120:             }
121:             Project p = ChosenExample.Project;
122:             TxtDescription.Enabled = true;
123:             TxtDescription.Clear();
124:             string modules = ChosenExample.UsedModules.CommaSeparatedList();
125:             if(string.IsNullOrEmpty(modules)) modules = i18n.str("none");
126:             TxtDescription.AppendText(i18n.str("ExampleDescription", p.Name, p.Description, p.Language
, p.MainBoard, modules).Replace("\\n", Environment.NewLine));
127:         }
128:
129:         private void LstFilterMB_ItemChecked(object sender, ItemCheckedEventArgs e) {
130:             UpdateExamplesList();
131:         }
132:
133:         private void LstFilterModules_ItemChecked(object sender, ItemCheckedEventArgs e) {
134:             UpdateExamplesList();
135:         }
136:
137:         private void BtnSave_Click(object sender, EventArgs e) {
138:             DialogResult = DialogResult.OK;
139:             Close();
140:         }
141:     }
142: }
```

```
1: // Archivo generado automáticamente por Visual Studio
2:
3: namespace CuruxaIDE {
4:     partial class FrmProjectSettings {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
>
14:        protected override void Dispose(bool disposing) {
15:            if(disposing && (components != null)) {
16:                components.Dispose();
17:            }
18:            base.Dispose(disposing);
19:        }
20:
21:        #region Windows Form Designer generated code
22:
23:        /// <summary>
24:        /// Required method for Designer support - do not modify
25:        /// the contents of this method with the code editor.
26:        /// </summary>
27:        private void InitializeComponent() {
28:            this.BtnOK = new System.Windows.Forms.Button();
29:            this.splitContainer1 = new System.Windows.Forms.SplitContainer();
30:            this.tableLayoutPanel5 = new System.Windows.Forms.TableLayoutPanel();
31:            this.CmbLanguage = new System.Windows.Forms.ComboBox();
32:            this.LblLanguage = new System.Windows.Forms.Label();
33:            this.Voltage = new System.Windows.Forms.NumericUpDown();
34:            this.LblVoltage = new System.Windows.Forms.Label();
35:            this.CmbMainBoard = new System.Windows.Forms.ComboBox();
36:            this.LblMainBoard = new System.Windows.Forms.Label();
37:            this.TxtName = new System.Windows.Forms.TextBox();
38:            this.LblName = new System.Windows.Forms.Label();
```



```
39:         this.LblDescription = new System.Windows.Forms.Label();
40:         this.TxtDescription = new System.Windows.Forms.TextBox();
41:         this.splitContainer1.Panel1.SuspendLayout();
42:         this.splitContainer1.Panel2.SuspendLayout();
43:         this.splitContainer1.SuspendLayout();
44:         this.tableLayoutPanel5.SuspendLayout();
45:         ((System.ComponentModel.ISupportInitialize)(this.Voltage)).BeginInit();
46:         this.SuspendLayout();
47:         //
48:         // BtnOK
49:         //
50:         this.BtnOK.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles
s.Bottom | System.Windows.Forms.AnchorStyles.Right)));
51:         this.BtnOK.Location = new System.Drawing.Point(155, 154);
52:         this.BtnOK.Name = "BtnOK";
53:         this.BtnOK.Size = new System.Drawing.Size(97, 28);
54:         this.BtnOK.TabIndex = 1;
55:         this.BtnOK.Text = "NS (OK)";
56:         this.BtnOK.UseVisualStyleBackColor = true;
57:         this.BtnOK.Click += new System.EventHandler(this.LblOK_Click);
58:         //
59:         // splitContainer1
60:         //
61:         this.splitContainer1.Dock = System.Windows.Forms.DockStyle.Fill;
62:         this.splitContainer1.Location = new System.Drawing.Point(0, 0);
63:         this.splitContainer1.MinimumSize = new System.Drawing.Size(370, 190);
64:         this.splitContainer1.Name = "splitContainer1";
65:         //
66:         // splitContainer1.Panel1
67:         //
68:         this.splitContainer1.Panel1.Controls.Add(this.tableLayoutPanel5);
69:         this.splitContainer1.Panel1MinSize = 168;
70:         //
71:         // splitContainer1.Panel2
72:         //
73:         this.splitContainer1.Panel2.Controls.Add(this.LblDescription);
74:         this.splitContainer1.Panel2.Controls.Add(this.TxtDescription);
75:         this.splitContainer1.Panel2.Controls.Add(this.BtnOK);
76:         this.splitContainer1.Size = new System.Drawing.Size(466, 190);
```

```
77:         this.splitContainer1.SplitterDistance = 198;
78:         this.splitContainer1.TabIndex = 2;
79:         //
80:         // tableLayoutPanel5
81:         //
82:         this.tableLayoutPanel5.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Left)
83:                                     | System.Windows.Forms.AnchorStyles.Right));
84:         this.tableLayoutPanel5.ColumnCount = 2;
85:         this.tableLayoutPanel5.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle());
86:         this.tableLayoutPanel5.ColumnStyles.Add(new System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 100F));
87:         this.tableLayoutPanel5.Controls.Add(this.CmbLanguage, 1, 3);
88:         this.tableLayoutPanel5.Controls.Add(this.LblLanguage, 0, 3);
89:         this.tableLayoutPanel5.Controls.Add(this.Voltage, 1, 2);
90:         this.tableLayoutPanel5.Controls.Add(this.LblVoltage, 0, 2);
91:         this.tableLayoutPanel5.Controls.Add(this.CmbMainBoard, 1, 1);
92:         this.tableLayoutPanel5.Controls.Add(this.LblMainBoard, 0, 1);
93:         this.tableLayoutPanel5.Controls.Add(this.TxtName, 1, 0);
94:         this.tableLayoutPanel5.Controls.Add(this.LblName, 0, 0);
95:         this.tableLayoutPanel5.GrowStyle = System.Windows.Forms.TableLayoutPanelGrowStyle.FixedSize;
96:         this.tableLayoutPanel5.Location = new System.Drawing.Point(3, 31);
97:         this.tableLayoutPanel5.Name = "tableLayoutPanel5";
98:         this.tableLayoutPanel5.RowCount = 5;
99:         this.tableLayoutPanel5.RowStyles.Add(new System.Windows.Forms.RowStyle());
100:        this.tableLayoutPanel5.RowStyles.Add(new System.Windows.Forms.RowStyle());
101:        this.tableLayoutPanel5.RowStyles.Add(new System.Windows.Forms.RowStyle());
102:        this.tableLayoutPanel5.RowStyles.Add(new System.Windows.Forms.RowStyle());
103:        this.tableLayoutPanel5.RowStyles.Add(new System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 20F));
104:        this.tableLayoutPanel5.Size = new System.Drawing.Size(192, 115);
105:        this.tableLayoutPanel5.TabIndex = 5;
106:        //
107:        // CmbLanguage
108:        //
109:        this.CmbLanguage.Dock = System.Windows.Forms.DockStyle.Fill;
110:        this.CmbLanguage.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
111:        this.CmbLanguage.FormattingEnabled = true;
```

```
112:         this.CmbLanguage.Location = new System.Drawing.Point(94, 82);
113:         this.CmbLanguage.Name = "CmbLanguage";
114:         this.CmbLanguage.Size = new System.Drawing.Size(95, 21);
115:         this.CmbLanguage.TabIndex = 9;
116:         //
117:         // LblLanguage
118:         //
119:         this.LblLanguage.AutoSize = true;
120:         this.LblLanguage.Dock = System.Windows.Forms.DockStyle.Fill;
121:         this.LblLanguage.Location = new System.Drawing.Point(3, 79);
122:         this.LblLanguage.Name = "LblLanguage";
123:         this.LblLanguage.Size = new System.Drawing.Size(85, 27);
124:         this.LblLanguage.TabIndex = 8;
125:         this.LblLanguage.Text = "NS (Language)";
126:         this.LblLanguage.TextAlign = System.Drawing.ContentAlignment.MiddleRight;
127:         //
128:         // Voltage
129:         //
130:         this.Voltage.DecimalPlaces = 1;
131:         this.Voltage.Dock = System.Windows.Forms.DockStyle.Fill;
132:         this.Voltage.Location = new System.Drawing.Point(94, 56);
133:         this.Voltage.Maximum = new decimal(new int[] {
134:             5,
135:             0,
136:             0,
137:             0});
138:         this.Voltage.Minimum = new decimal(new int[] {
139:             1,
140:             0,
141:             0,
142:             0});
143:         this.Voltage.Name = "Voltage";
144:         this.Voltage.Size = new System.Drawing.Size(95, 20);
145:         this.Voltage.TabIndex = 7;
146:         this.Voltage.TextAlign = System.Windows.Forms.HorizontalAlignment.Center;
147:         this.Voltage.Value = new decimal(new int[] {
148:             2,
149:             0,
150:             0,
```

```
151:         0});
152:         //
153:         // LblVoltage
154:         //
155:         this.LblVoltage.AutoSize = true;
156:         this.LblVoltage.Dock = System.Windows.Forms.DockStyle.Fill;
157:         this.LblVoltage.Location = new System.Drawing.Point(3, 53);
158:         this.LblVoltage.Name = "LblVoltage";
159:         this.LblVoltage.Size = new System.Drawing.Size(85, 26);
160:         this.LblVoltage.TabIndex = 6;
161:         this.LblVoltage.Text = "NS (Voltage)";
162:         this.LblVoltage.TextAlign = System.Drawing.ContentAlignment.MiddleRight;
163:         //
164:         // CmbMainBoard
165:         //
166:         this.CmbMainBoard.Dock = System.Windows.Forms.DockStyle.Fill;
167:         this.CmbMainBoard.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
168:         this.CmbMainBoard.FormattingEnabled = true;
169:         this.CmbMainBoard.Location = new System.Drawing.Point(94, 29);
170:         this.CmbMainBoard.Name = "CmbMainBoard";
171:         this.CmbMainBoard.Size = new System.Drawing.Size(95, 21);
172:         this.CmbMainBoard.TabIndex = 5;
173:         //
174:         // LblMainBoard
175:         //
176:         this.LblMainBoard.AutoSize = true;
177:         this.LblMainBoard.Dock = System.Windows.Forms.DockStyle.Fill;
178:         this.LblMainBoard.Location = new System.Drawing.Point(3, 26);
179:         this.LblMainBoard.Name = "LblMainBoard";
180:         this.LblMainBoard.Size = new System.Drawing.Size(85, 27);
181:         this.LblMainBoard.TabIndex = 4;
182:         this.LblMainBoard.Text = "NS (Main Board)";
183:         this.LblMainBoard.TextAlign = System.Drawing.ContentAlignment.MiddleRight;
184:         //
185:         // TxtName
186:         //
187:         this.TxtName.Dock = System.Windows.Forms.DockStyle.Fill;
188:         this.TxtName.Location = new System.Drawing.Point(94, 3);
189:         this.TxtName.Name = "TxtName";
```

```
190:         this.TxtName.ReadOnly = true;
191:         this.TxtName.Size = new System.Drawing.Size(95, 20);
192:         this.TxtName.TabIndex = 3;
193:         //
194:         // LblName
195:         //
196:         this.LblName.AutoSize = true;
197:         this.LblName.Dock = System.Windows.Forms.DockStyle.Fill;
198:         this.LblName.Location = new System.Drawing.Point(3, 0);
199:         this.LblName.Name = "LblName";
200:         this.LblName.Size = new System.Drawing.Size(85, 26);
201:         this.LblName.TabIndex = 2;
202:         this.LblName.Text = "NS (name)";
203:         this.LblName.TextAlign = System.Drawing.ContentAlignment.MiddleRight;
204:         //
205:         // LblDescription
206:         //
207:         this.LblDescription.AutoSize = true;
208:         this.LblDescription.Location = new System.Drawing.Point(13, 14);
209:         this.LblDescription.Name = "LblDescription";
210:         this.LblDescription.Size = new System.Drawing.Size(84, 13);
211:         this.LblDescription.TabIndex = 3;
212:         this.LblDescription.Text = "NS (Description)";
213:         //
214:         // TxtDescription
215:         //
216:         this.TxtDescription.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.
AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)
217:         | System.Windows.Forms.AnchorStyles.Left)
218:         | System.Windows.Forms.AnchorStyles.Right)));
219:         this.TxtDescription.Location = new System.Drawing.Point(16, 33);
220:         this.TxtDescription.Multiline = true;
221:         this.TxtDescription.Name = "TxtDescription";
222:         this.TxtDescription.Size = new System.Drawing.Size(236, 115);
223:         this.TxtDescription.TabIndex = 2;
224:         //
225:         // FrmProjectSettings
226:         //
227:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
```

```
228:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
229:         this.ClientSize = new System.Drawing.Size(466, 186);
230:         this.Controls.Add(this.splitContainer1);
231:         this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.SizableToolWindow;
232:         this.MaximizeBox = false;
233:         this.MinimumSize = new System.Drawing.Size(390, 210);
234:         this.Name = "FrmProjectSettings";
235:         this.ShowIcon = false;
236:         this.Text = "NS (Project Settings)";
237:         this.Load += new System.EventHandler(this.ProjectSettings_Load);
238:         this.splitContainer1.Panel1.ResumeLayout(false);
239:         this.splitContainer1.Panel2.ResumeLayout(false);
240:         this.splitContainer1.Panel2.PerformLayout();
241:         this.splitContainer1.ResumeLayout(false);
242:         this.tableLayoutPanel5.ResumeLayout(false);
243:         this.tableLayoutPanel5.PerformLayout();
244:         ((System.ComponentModel.ISupportInitialize)(this.Voltage)).EndInit();
245:         this.ResumeLayout(false);
246:
247:     }
248:
249: #endregion
250:
251:     private System.Windows.Forms.Button BtnOK;
252:     private System.Windows.Forms.SplitContainer splitContainer1;
253:     private System.Windows.Forms.Label LblDescription;
254:     private System.Windows.Forms.TextBox TxtDescription;
255:     private System.Windows.Forms.TableLayoutPanel tableLayoutPanel5;
256:     private System.Windows.Forms.ComboBox CmbLanguage;
257:     private System.Windows.Forms.Label LblLanguage;
258:     private System.Windows.Forms.NumericUpDown Voltage;
259:     private System.Windows.Forms.Label LblVoltage;
260:     private System.Windows.Forms.ComboBox CmbMainBoard;
261:     private System.Windows.Forms.Label LblMainBoard;
262:     private System.Windows.Forms.TextBox TxtName;
263:     private System.Windows.Forms.Label LblName;
264:
265: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Data;
5: using System.Drawing;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9:
10: namespace CuruxaIDE {
11:     public partial class FrmProjectSettings:Form {
12:         Project Prj;
13:
14:         public FrmProjectSettings(Project Prj) {
15:             InitializeComponent();
16:             this.Prj = Prj;
17:             UpdateLang();
18:         }
19:
20:         private void UpdateLang() {
21:             this.Text = i18n.str("ProjectSettings");
22:             LblDescription.Text = i18n.str("Description:");
23:             LblName.Text = i18n.str("Name:");
24:             LblMainBoard.Text = i18n.str("MainBoard:");
25:             LblVoltage.Text = i18n.str("Voltage:");
26:             LblLanguage.Text = i18n.str("Language:");
27:             BtnOK.Text = i18n.str("OK");
28:         }
29:
30:         private void ProjectSettings_Load(object sender, EventArgs e) {
31:             CmbMainBoard.Items.Clear();
32:             foreach(string MB in Enum.GetNames(typeof(MainBoard))) {
33:                 CmbMainBoard.Items.Add(MB);
34:             }
35:             TxtName.Text = Prj.Name;
36:             CmbMainBoard.SelectedItem = Prj.MainBoard.ToString();
37:             Voltage.Increment = 0.1M;
38:             Voltage.Value = (decimal)Prj.Voltage;
39:             CmbLanguage.Items.Clear();
```

```
40:         foreach(string lang in Enum.GetNames(typeof(Language))) {
41:             if(lang != "Text") CmbLanguage.Items.Add(lang);
42:         }
43:         CmbLanguage.SelectedItem = Prj.Language.ToString();
44:         TxtDescription.Text = Prj.Description;
45:     }
46:
47:     private void LblOK_Click(object sender, EventArgs e) {
48:         SaveSettings();
49:         this.DialogResult = DialogResult.OK;
50:         this.Close();
51:     }
52:
53:     private void SaveSettings() {
54:         Prj.Name = TxtName.Text;
55:         Prj.MainBoard = (MainBoard)Enum.Parse(typeof(MainBoard), (string)CmbMainBoard.SelectedItem
);
56:         Prj.Voltage = (float)Voltage.Value;
57:         Prj.Language = (Language)Enum.Parse(typeof(Language), (string)CmbLanguage.SelectedItem);
58:         Prj.Description = TxtDescription.Text;
59:     }
60:
61:     private void TxtName_TextChanged(object sender, EventArgs e) {
62:         CheckParams();
63:     }
64:
65:     /// <summary>
66:     /// Comprueba si todas las opciones son correctas y activa o desactiva el botón de aceptar según s
ea necesario
67:     /// </summary>
68:     private void CheckParams() {
69:         if(TxtName.Text.Length > 0) BtnOK.Enabled = true;
70:         else BtnOK.Enabled = false;
71:     }
72: }
73: }
```



```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Windows.Forms;
5: using System.Drawing;
6: using System.IO;
7: using System.Drawing.Imaging;
8: using System.Net;
9:
10: namespace CuruxaIDE {
11:     static class Globals {
12:         public static Project ActiveProject {
13:             get {
14:                 return _ActiveProject;
15:             }
16:             set {
17:                 _ActiveProject = value;
18:                 if(MainWindow != null) MainWindow.UpdateActivePrj(value);
19:             }
20:         }
21:         private static Project _ActiveProject;
22:
23:         public static SrcFile ActiveSrcFile;
24:
25:         public static FrmMainWindow MainWindow;
26:
27:         /// <summary>
28:         /// Punto de entrada de la aplicación
29:         /// </summary>
30:         [STAThread]
31:         static void Main() {
32:             //si quieres generar un nuevo ejemplo, descomenta la siguiente línea
33:             //GenerateExample();
34:
35:             #if !DEBUG
36:                 Application.SetUnhandledExceptionMode(System.Windows.Forms.UnhandledExceptionMode.CatchExc
37:                 Application.ThreadException += delegate(object sender, System.Threading.ThreadExceptionEve
ntArgs ar) {
```

```

38:                Crash(ar.Exception);
39:            };
40:            AppDomain.CurrentDomain.UnhandledException += delegate(object sender, UnhandledExceptionEv
entArgs ar) {
41:                Crash(ar.ExceptionObject);
42:            };
43: #endif
44:
45:         try {
46:             Settings.Load();
47:             Application.EnableVisualStyles();
48:             Application.SetCompatibleTextRenderingDefault(true);
49:             Application.Run(MainWindow = new FrmMainWindow());
50:         } catch (TypeInitializationException e) {
51:             if(e.TargetSite.ReflectedType.FullName == "System.Windows.Forms.Application" && e.
TargetSite.Name == "EnableVisualStyles") {
52:                 Console.WriteLine("Windows Forms not supported!!!");
53:             } else throw e;
54:         }
55: #if !DEBUG
56:         catch (Exception e) {
57:             Crash(e);
58:         }
59: #endif
60:     }
61:
62:     private static void GenerateExample() {
63:         Example NewExample = new Example();
64:         NewExample.Project.Name = "MBP18_test01";
65:         NewExample.Project.Description = "Example program for testing MBP18";
66:         NewExample.Project.Language = Language.C;
67:         NewExample.Project.MainBoard = MainBoard.MBP18;
68:         NewExample.Project.SrcFiles.Add(new SrcFile(NewExample.Project, NewExample.Project.Name +
"." + NewExample.Project.Language.GetExtension()));
69:         NewExample.Save(NewExample.Project.Name + "." + Example.ExampleExtension);
70:     }
71:
72:     private static void Crash(object Error) {
73:         new FrmCrash(Error).ShowDialog();

```

```
74:         }
75:
76:         /// <summary>
77:         /// Muestra un texto en el log de Curuxa IDE
78:         /// </summary>
79:         public static void LogIDE(string Text) {
80:             Debug("[IDE] " + Text);
81:             if(MainWindow != null) MainWindow.LogIDE(Text, false);
82:         }
83:
84:         /// <summary>
85:         /// Muestra un texto en el log de compilación
86:         /// </summary>
87:         public static void LogBuild(string Text) {
88:             if(MainWindow != null) MainWindow.LogBuild(Text);
89:         }
90:
91:         /// <summary>
92:         /// Muestra un texto en el log del programador
93:         /// </summary>
94:         public static void LogProgrammer(string Text) {
95:             if(MainWindow != null) MainWindow.LogProgrammer(Text);
96:         }
97:
98:         public static void Debug(string Text) {
99:             Console.WriteLine(string.Format("[{0:00}:{1:00}:{2:00}.{3:000}] {4}", DateTime.Now.Hour, DateTime.Now.Minute, DateTime.Now.Second, DateTime.Now.Millisecond, Text));
100:        }
101:
102:        public static void Debug(string Text, params object[] p) {
103:            Debug(string.Format(Text, p));
104:        }
105:
106:        /// <summary>
107:        /// Configura un nuevo log de compilación. Le da foco a la pestaña del log y elimina texto antiguo
108:        /// </summary>
109:        public static void SetupNewBuildLog() {
110:            if(MainWindow != null) {
111:                MainWindow.TxtLogBuild.Clear();
```

```
112:                 MainWindow.TabsMsg.SelectedTab = MainWindow.TabBuildLog;
113:             }
114:         }
115:
116:         /// <summary>
117:         /// Configura un nuevo log de programación. Le da foco a la pestaña del log y elimina texto antiguo
118:         /// </summary>
119:         public static void SetupNewProgrammerLog() {
120:             if(MainWindow != null) {
121:                 MainWindow.TxtLogProgrammer.Clear();
122:                 MainWindow.TabsMsg.SelectedTab = MainWindow.TabProgLog;
123:             }
124:         }
125:
126:         public static Image LoadImage(string FileName) {
127:             string path = Settings.ImagesDir + "/" + FileName;
128:             if(File.Exists(path)) return Image.FromFile(path);
129:             else {
130:                 LogIDE("Image not found: " + FileName);
131:                 Bitmap blank = new Bitmap(1, 1);
132:                 blank.SetPixel(0, 0, Color.White);
133:                 return blank;
134:             }
135:         }
136:
137:         public static void CheckVersion() {
138:             WebClient wc = new WebClient();
139:             wc.DownloadStringCompleted += new DownloadStringCompletedEventHandler(wc_DownloadStringCom
140:             try {
141:                 wc.DownloadStringAsync(new Uri("http://curuxa.org/curuxa_latest"));
142:             } catch(WebException we) {
143:                 LogIDE(i18n.str("UnableGetLatestVersion", we.Message));
144:             }
145:         }
146:
147:         static void wc_DownloadStringCompleted(object sender, DownloadStringCompletedEventArgs e) {
148:             if(e.Cancelled == true) LogIDE(i18n.str("UnableGetLatestVersion", "Cancelled"));
```

```
149:         else if(e.Error != null) LogIDE(i18n.str("UnableGetLatestVersion", e.Error.Message));
150:         else {
151:             string[] LatestVersion = e.Result.RemoveFrom('-').Split('.');
152:             string[] CurrentVersion = Settings.Version.RemoveFrom('-').Split('.');
153:             for(int i = 0; i < LatestVersion.Length; i++) {
154:                 UInt32 LatestVersionX = 0;
155:                 UInt32.TryParse(LatestVersion[i], out LatestVersionX);
156:                 UInt32 CurrentVersionX = 0;
157:                 if(i < CurrentVersion.Length) UInt32.TryParse(CurrentVersion[i], out CurrentVersionX);
158:                 if(LatestVersionX > CurrentVersionX) {
159:                     ThereIsNewVersion(e.Result);
160:                     break;
161:                 }
162:                 if(LatestVersionX < CurrentVersionX) {
163:                     Debug("You are using a more recent version than the latest official
164: release. Wow! You must be God or something.");
165:                 }
166:             }
167:         }
168:
169:         private static void ThereIsNewVersion(string NewVersion) {
170:             LogIDE(i18n.str("ThereIsNewVersion", NewVersion, Settings.Version));
171:         }
172:     }
173: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace CuruxaIDE {
7:     public static class IEnumerableOfStringExtensions {
8:         /// <summary>
9:         /// Obtiene la lista de textos separados por una coma y un espacio
10:        /// </summary>
11:        public static string CommaSeparatedList(this IEnumerable<string> strs) {
12:            return strs.CommaSeparatedList(true);
13:        }
14:
15:        /// <summary>
16:        /// Obtiene la lista de textos separados por una coma
17:        /// </summary>
18:        /// <param name="spaced">True is la lista debe estar separada por coma+espacio. False si el separa
19:        dor debe ser sólo una coma</param>
20:        public static string CommaSeparatedList(this IEnumerable<string> strs, bool spaced) {
21:            string separator = ",";
22:            if(spaced) separator = ", ";
23:            string list = "";
24:            foreach(string str in strs) list += str + separator;
25:            if(list.Length > 2) list = list.Substring(0, list.Length - separator.Length);
26:            return list;
27:        }
28:
29:        /// <summary>
30:        /// Obtiene la lista de Módulos separados por coma+espacio
31:        /// </summary>
32:        public static string CommaSeparatedList(this List<Module> M) {
33:            string[] ms = new string[M.Count];
34:            for(int i=0;i<ms.Length;i++) ms[i]=M[i].GetRealName();
35:            return ms.CommaSeparatedList();
36:        }
37:
38:        /// <summary>
39:        /// Comprueba si una colección contiene cierto valor
```

```
39:          /// </summary>
40:          public static bool Contains(this string[] strs, string value) {
41:              foreach(string v in strs) {
42:                  if(v == value) return true;
43:              }
44:              return false;
45:          }
46:      }
47: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace CuruxaIDE {
7:     /// <summary>
8:     /// Tipo de #include
9:     /// </summary>
10:    public enum IncludeType {
11:        /// <summary>
12:        /// #include "SomeLibrary.h"
13:        /// </summary>
14:        Local,
15:
16:        /// <summary>
17:        /// #include <SomeLibrary.h>
18:        /// </summary>
19:        Global
20:    }
21: }
```



```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace CuruxaIDE {
7:     /// <summary>
8:     /// Lenguajes de Programación
9:     /// </summary>
10:    public enum Language {
11:        /// <summary>
12:        /// Texto plano
13:        /// </summary>
14:        Text,
15:
16:        /// <summary>
17:        /// Lenguaje Ensamblador para PICs de Microchip
18:        /// </summary>
19:        PicAsm,
20:
21:        /// <summary>
22:        /// Lenguaje C
23:        /// </summary>
24:        C
25:    }
26:
27:    public static class LanguageExtensions {
28:        static Dictionary<Language, string[]> LangExtensions = new Dictionary<Language, string[]>();
29:
30:        static LanguageExtensions() {
31:            //NO AÑADAS ARRAYS VACIOS
32:            LangExtensions.Add(Language.Text, new string[]{"txt"});
33:            LangExtensions.Add(Language.C, new string[]{"c", "h"});
34:            LangExtensions.Add(Language.PicAsm, new string[]{"asm"});
35:        }
36:
37:        /// <summary>
38:        /// Obtiene el lenguaje dada una extensión de archivo
39:        /// </summary>
```

```
40:         public static Language GetFromExtension(string extension) {
41:             extension = extension.ToLowerInvariant();
42:             foreach(KeyValuePair<Language, string[]> pair in LangExtensions) {
43:                 if(pair.Value.Contains(extension)) return pair.Key;
44:             }
45:             return Language.Text;
46:         }
47:
48:         /// <summary>
49:         /// Obtiene la extensión por defecto de cierto lenguaje de programación (sólo la extensión, sin pu
nto)
50:         /// </summary>
51:         public static string GetExtension(this Language lang) {
52:             return LangExtensions[lang][0];
53:         }
54:
55:         /// <summary>
56:         /// Obtiene la herramienta usada para compilar el lenguaje de programación dado)
57:         /// </summary>
58:         public static Toolsuite GetToolsuite(this Language lang) {
59:             switch(lang) {
60:                 case Language.C:
61:                     return Toolsuite.SDCC;
62:                 default:
63:                     throw new NotImplementedException();
64:             }
65:         }
66:     }
67: }
```

```
1: using System;
2: using System.Collections.Generic;
3:
4: namespace CuruxaIDE {
5:     public static class ListSrcFileExtensions {
6:         /// <summary>
7:         /// Comprueba si el nombre del archivo existe en la colección o no
8:         /// </summary>
9:         public static bool ContainsFileName(this List<SrcFile> sources, string FileName) {
10:             foreach(SrcFile src in sources) {
11:                 if(src.FileName == FileName) return true;
12:             }
13:             return false;
14:         }
15:
16:         /// <summary>
17:         /// Comprueba si un archivo existe en la colección, dada la ruta completa al archivo
18:         /// </summary>
19:         public static bool ContainsFullPath(this List<SrcFile> sources, string FullPath) {
20:             foreach(SrcFile src in sources) {
21:                 if(src.FullPath == FullPath) return true;
22:             }
23:             return false;
24:         }
25:
26:         /// <summary>
27:         /// Obtiene el SrcFile de la colección, dado su FileName
28:         /// </summary>
29:         public static SrcFile GetByFileName(this List<SrcFile> sources, string FileName) {
30:             foreach(SrcFile src in sources) {
31:                 if(src.FileName == FileName) return src;
32:             }
33:             throw new ArgumentException("File name \" + FileName + "\" not found in List<SrcFile>");
34:         }
35:
36:         /// <summary>
37:         /// Obtiene el SrcFile de la colección, dada la ruta completa al archivo
38:         /// </summary>
39:         public static SrcFile GetByFullPath(this List<SrcFile> sources, string FullPath) {
```

```
40:         foreach(SrcFile src in sources) {
41:             if(src.FullPath == FullPath) return src;
42:         }
43:         throw new ArgumentException("Full path \" + FullPath + "\" not found in List<SrcFile>");
44:     }
45: }
46: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Windows.Forms;
4:
5: namespace CuruxaIDE {
6:     public static class ListViewGroupCollectionExtensions {
7:         /// <summary>
8:         /// Comprueba si hay algún grupo con cierto nombre en la colección
9:         /// </summary>
10:        public static bool ContainsName(this ListViewGroupCollection lvgc, string Name) {
11:            foreach(ListViewGroup group in lvgc) {
12:                if(group.Name == Name) return true;
13:            }
14:            return false;
15:        }
16:    }
17: }
```

```
1: using System;
2: using System.ComponentModel;
3: using System.Reflection;
4:
5: namespace CuruxaIDE {
6:     /// <summary>
7:     /// Lista de Placas Principales disponibles
8:     /// </summary>
9:     public enum MainBoard {
10:         [Description("PIC12F683")]
11:         MBP8,
12:
13:         [Description("PIC16F688")]
14:         MBP14,
15:
16:         [Description("PIC16F88")]
17:         MBP18,
18:
19:         [Description("PIC16F887")]
20:         MBP40
21:     }
22:
23:     public static class MainBoardsExtensions {
24:         /// <summary>
25:         /// Obtiene el Microcontrolador usado en esta Placa Principal
26:         /// </summary>
27:         public static Microcontroller GetMCU(this MainBoard MainBoard) {
28:             FieldInfo fi = MainBoard.GetType().GetField(MainBoard.ToString());
29:
30:             DescriptionAttribute[] attributes = (DescriptionAttribute[])fi.GetCustomAttributes(typeof(
DescriptionAttribute), false);
31:
32:             if(attributes != null && attributes.Length > 0)
33:                 return (Microcontroller)Enum.Parse(typeof(Microcontroller), attributes[0].Descript
ion);
34:             else
35:                 throw new NotImplementedException("No Microcontroller assigned to this Main board"
);
36:         }
```

```
37:      }  
38: }
```

```
1: using System;
2: using System.Reflection;
3:
4: namespace CuruxaIDE {
5:     /// <summary>
6:     /// Lista de microcontroladores disponibles
7:     /// </summary>
8:     public enum Microcontroller {
9:         PIC12F683,
10:        PIC16F688,
11:        PIC16F88,
12:        PIC16F887
13:    }
14:
15:     public static class MicrocontrollerExtensions {
16:         /// <summary>
17:         /// Obtiene la Placa Principal que utiliza este microcontrolador. Devuelve null si el microcontrolador no se usa en Curuxa
18:         /// </summary>
19:         public static MainBoard? GetMainBoard(this Microcontroller MCU) {
20:             foreach(MainBoard MB in Enum.GetValues(typeof(MainBoard))) {
21:                 if(MB.GetMCU() == MCU) return MB;
22:             }
23:             return null;
24:         }
25:     }
26: }
```



```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.ComponentModel;
6: using System.Reflection;
7:
8: namespace CuruxaIDE {
9:     /// <summary>
10:    /// Lista de Módulos disponibles
11:    /// </summary>
12:    public enum Module {
13:        [Description("SISW-SPST")]
14:        SISW_SPST,
15:
16:        [Description("SIBW-1Y")]
17:        SIBW_1Y,
18:
19:        [Description("LTIND-A")]
20:        LTIND_A,
21:
22:        [Description("LTIL-A")]
23:        LTIL_A,
24:
25:        [Description("AO-SPK")]
26:        AO_SPK,
27:
28:        [Description("CMIR-RC")]
29:        CMIR_RC,
30:
31:        [Description("SIDST-GP2")]
32:        SIDST_GP2,
33:
34:        MC2A,
35:
36:        MC2B
37:    }
38:
39:    public static class ModuleExtensions {
```

```
40:         /// <summary>
41:         /// Obtiene el nombre real de este Módulo
42:         /// </summary>
43:         public static string GetRealName(this Module M) {
44:             FieldInfo fi = M.GetType().GetField(M.ToString());
45:
46:             DescriptionAttribute[] attributes = (DescriptionAttribute[])fi.GetCustomAttributes(typeof(
DescriptionAttribute), false);
47:
48:             if(attributes != null && attributes.Length > 0) return attributes[0].Description;
49:             else return M.ToString();
50:         }
51:
52:         /// <summary>
53:         /// Obtiene un Mmódulo dado su nombre real
54:         /// </summary>
55:         public static Module GetModuleFromRealName(string M) {
56:             foreach(string Mod in Enum.GetNames(typeof(Module))) {
57:                 Module m = (Module)Enum.Parse(typeof(Module), Mod);
58:                 if(M == m.GetRealName()) return m;
59:             }
60:             throw new ArgumentException(string.Format("\'" + M + "\" is not the real name of any known
Module"));
61:         }
62:
63:         /// <summary>
64:         /// Obtiene la lista de Módulos compatibles con cierto Módulo, incluido él mismo
65:         /// </summary>
66:         public static List<Module> GetCompatibleModules(Module M) {
67:             List<Module> CM = new List<Module>();
68:             CM.Add(M);
69:             foreach(List<Module> L in CompatibleModules) {
70:                 if(L.Contains(M)) CM.AddRange(L);
71:             }
72:             return CM.ToArray();
73:         }
74:
75:         static List<List<Module>> CompatibleModules;
76:
```

```
77:         static ModuleExtensions() {
78:             CompatibleModules = new List<List<Module>>();
79:
80:             Module[] a = { Module.LTIND_A, Module.LTIL_A };
81:             Module[] b = { Module.MC2A, Module.MC2B };
82:             CompatibleModules.Add(new List<Module>(a));
83:             CompatibleModules.Add(new List<Module>(b));
84:         }
85:     }
86: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace CuruxaIDE {
7:     /// <summary>
8:     /// Lista de programadores de microcontroladores disponibles
9:     /// </summary>
10:    public enum Programmer {
11:        /// <summary>
12:        /// Microchip PICkit 2 programmer
13:        /// </summary>
14:        PICkit2
15:    }
16:
17:    public static class ProgrammerExtensions {
18:        /// <summary>
19:        /// Obtiene el programa usado para comunicarse con el programador
20:        /// </summary>
21:        public static ProgrammerApp GetProgrammerApp(this Programmer prog) {
22:            switch(prog) {
23:                case Programmer.PICkit2:
24:                    return ProgrammerApp.pk2cmd;
25:                default:
26:                    throw new NotImplementedException();
27:            }
28:        }
29:    }
30: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace CuruxaIDE {
7:     /// <summary>
8:     /// Programador de microcontroladores
9:     /// </summary>
10:    public abstract class ProgrammerApp:SoftApp {
11:        protected ProgrammerApp(string RealName, string Command)
12:            : base(RealName, Command) {
13:        }
14:
15:        /// <summary>
16:        /// Graba el programa dado en el microcontrolador
17:        /// </summary>
18:        public abstract int Write(Project prj);
19:
20:        /// <summary>
21:        /// Alimenta el circuito a la tensión configurada para que el programa comience a ejecutarse
22:        /// </summary>
23:        public abstract int Run(Project prj);
24:
25:        /// <summary>
26:        /// Apaga el circuito
27:        /// </summary>
28:        public abstract int Stop(Project prj);
29:
30:        /// <summary>
31:        /// Detecta el microcontrolador conectado al programador
32:        /// </summary>
33:        public abstract Microcontroller? Detect();
34:
35:        public static pk2cmd pk2cmd = new pk2cmd();
36:    }
37: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.IO;
6: using System.Xml.Serialization;
7: using System.Xml;
8:
9: namespace CuruxaIDE {
10:     [Serializable]
11:     public class Project {
12:         /// <summary>
13:         /// Nombre del proyecto
14:         /// </summary>
15:         public string Name = "UNDEFINED";
16:
17:         /// <summary>
18:         /// Ruta el directorio del proyecto
19:         /// </summary>
20:         [XmlIgnore]
21:         public string Path = "UNDEFINED";
22:
23:         /// <summary>
24:         /// Descripción del proyecto
25:         /// </summary>
26:         /// <remarks>
27:         /// Un texto que el usuario puede modificar y que explica qué hace este proyecto
28:         /// </remarks>
29:         public string Description;
30:
31:         /// <summary>
32:         /// Indica si este proyecto ha sido compilado correctamente y no ha sido modificado desde entonces
33:         /// </summary>
34:         [XmlIgnore]
35:         public bool IsCompiled { get; protected set; }
36:
37:         /// <summary>
38:         /// Indica si este proyecto ha sido correctamente grabado en el microcontrolador y no ha sido modificado desde entonces
```

```
39:         /// </summary>
40:         [XmlIgnore]
41:         public bool IsBurnt { get; protected set; }
42:
43:         /// <summary>
44:         /// Indica si este proyecto se está ejecutando
45:         /// </summary>
46:         [XmlIgnore]
47:         public bool IsRunning { get; protected set; }
48:
49:         /// <summary>
50:         /// Ruta completa al archivo de proyecto
51:         /// </summary>
52:         [XmlIgnore]
53:         public string PrjFilePath {
54:             get {
55:                 return Path + "/" + Name + ".crx";
56:             }
57:             set {
58:                 FileInfo FI = new FileInfo(value);
59:                 Path = FI.DirectoryName;
60:                 Name = FI.Name.Replace(".crx", "");
61:             }
62:         }
63:
64:         /// <summary>
65:         /// Lista de archivos de código. Rutas relativas al directorio del proyecto
66:         /// </summary>
67:         [XmlIgnore]
68:         public List<SrcFile> SrcFiles = new List<SrcFile>();
69:
70:         /// <summary>
71:         /// Lista de librerías usadas en este proyecto
72:         /// </summary>
73:         [XmlIgnore]
74:         public List<SrcFile> Libs {
75:             get {
76:                 _Libs.Clear();
77:                 foreach(SrcFile src in SrcFiles) {
```

```
78:                foreach(SrcFile lib in src.RefLibs) {
79:                    ParseRefLib(lib);
80:                }
81:            }
82:            return _Libs;
83:        }
84:    }
85:    protected List<SrcFile> _Libs = new List<SrcFile>();
86:
87:    /// <summary>
88:    /// Añade una librería y todas sus referencias a la lista de librerías referenciadas, sólo si aún
no existen
89:    /// </summary>
90:    /// <param name="lib">Librería referenciada desde este proyecto</param>
91:    protected void ParseRefLib(SrcFile lib) {
92:        if(!_Libs.Contains(lib)) {
93:            _Libs.Add(lib);
94:            foreach(SrcFile reflib in lib.RefLibs) {
95:                ParseRefLib(reflib);
96:            }
97:        }
98:    }
99:
100:    public void AddSrcFile(SrcFile src) {
101:        SrcFiles.Add(src);
102:    }
103:
104:    public void AddSrcFile(string FilePath) {
105:        SrcFiles.Add(new SrcFile(this, FilePath));
106:    }
107:
108:    public string[] SrcFileNames {
109:        get {
110:            string[] fn = new string[SrcFiles.Count];
111:            for(int i = 0; i < fn.Length; i++) {
112:                fn[i] = SrcFiles[i].FileName;
113:            }
114:            return fn;
115:        }
    }
```



```
116:         set {
117:             SrcFiles.Clear();
118:             foreach(string file in value) {
119:                 SrcFile s = new SrcFile(this, file);
120:                 if(!SrcFiles.Contains(s)) SrcFiles.Add(s);
121:             }
122:         }
123:     }
124:
125:     /// <summary>
126:     /// Obtiene el archivo principal del proyecto
127:     /// </summary>
128:     public SrcFile MainFile {
129:         get {
130:             switch(Language) {
131:                 case Language.C:
132:                     foreach(SrcFile src in SrcFiles) {
133:                         if(src.Extension == ".c") return src;
134:                     }
135:                     break;
136:                 default:
137:                     throw new NotImplementedException();
138:             }
139:             return null;
140:         }
141:     }
142:
143:     /// <summary>
144:     /// Directorio donde se almacenan los archivos de salida generados tras la compilación. Ruta relativa al directorio del proyecto
145:     /// </summary>
146:     public string OutputPath = "/bin";
147:
148:     /// <summary>
149:     /// Ruta completa al directorio donde se almacenan los archivos de salida generados tras la compilación
150:     /// </summary>
151:     public string OutputFullPath {
152:         get {
```

```
153:         return Path + OutputPath;
154:     }
155: }
156:
157: /// <summary>
158: /// Placa Principal usada en este proyecto
159: /// </summary>
160: public MainBoard MainBoard = MainBoard.MBP18;
161:
162: /// <summary>
163: /// Voltaje al que se debe alimentar este proyecto
164: /// </summary>
165: public float Voltage = 5f;
166:
167: /// <summary>
168: /// Lenguaje de Programación
169: /// </summary>
170: public Language Language = Language.C;
171:
172: public Programmer Programmer = Programmer.PICKit2;
173:
174:
175: /// <summary>
176: /// Indica si este proyecto ha sido modificado
177: /// </summary>
178: [XmlIgnore]
179: public bool Modified {
180:     get {
181:         return _Modified;
182:     }
183:     set {
184:         if(value == true) {
185:             IsCompiled = false;
186:             IsBurnt = false;
187:         }
188:         _Modified = value;
189:     }
190: }
191: private bool _Modified = false;
```

```
192:
193:     /// <summary>
194:     /// Guarda el proyecto y sus archivos de código fuente
195:     /// </summary>
196:     public void SaveAll() {
197:         Save();
198:         foreach(SrcFile src in SrcFiles) src.SaveFile();
199:     }
200:
201:     /// <summary>
202:     /// Guarda el proyecto, pero no los archivos de código fuente
203:     /// </summary>
204:     public void Save() {
205:         XmlSerializer xs = new XmlSerializer(this.GetType());
206:         StreamWriter sw = null;
207:         try {
208:             sw = new StreamWriter(PrjFilePath);
209:             xs.Serialize(sw, this);
210:             Modified = false;
211:             Globals.LogIDE(i18n.str("SavingFile", PrjFilePath));
212:         } catch(IOException e) {
213:             Globals.LogIDE(i18n.str("UnableSaveFile", PrjFilePath));
214:             Globals.Debug(e.Message);
215:         } finally {
216:             if(sw != null) sw.Close();
217:         }
218:     }
219:
220:     /// <summary>
221:     /// Cierra este proyecto
222:     /// </summary>
223:     public void Close() {
224:         Globals.Debug("Removing project {0} from the list of open projects", this.Name);
225:         _OpenProjects.Remove(this);
226:         Settings.RemoveOpenProject(this.PrjFilePath);
227:         if(Globals.ActiveProject == this) {
228:             Globals.ActiveProject = null;
229:             Globals.ActiveSrcFile = null;
230:         }
```

```
231:         if(Globals.MainWindow != null) Globals.MainWindow.UpdatePrjList();
232:     }
233:
234:     /// <summary>
235:     /// Compilar proyecto
236:     /// </summary>
237:     public int Build() {
238:         Globals.LogIDE(i18n.str("BuildingPrj", Name));
239:         int r = Language.GetToolsuite().Build(this);
240:         if(r == 0) IsCompiled = true;
241:         return r;
242:     }
243:
244:     /// <summary>
245:     /// Grabar proyecto en el microcontrolador
246:     /// </summary>
247:     public int Burn() {
248:         Globals.LogIDE(i18n.str("BurningPrj", Name));
249:         Globals.SetupNewProgrammerLog();
250:         int r = Programmer.GetProgrammerApp().Write(this);
251:         if(r == 0) IsBurnt = true;
252:         return r;
253:     }
254:
255:     /// <summary>
256:     /// Alimenta el microcontrolador para que el proyecto comience a ejecutarse
257:     /// </summary>
258:     public int Run() {
259:         Globals.LogIDE(i18n.str("RunningPrj", Name));
260:         Globals.SetupNewProgrammerLog();
261:         int r = Programmer.GetProgrammerApp().Run(this);
262:         if(r == 0) IsRunning = true;
263:         return r;
264:     }
265:
266:     /// <summary>
267:     /// Detiene el microcontrolador para que el programa deje de ejecutarse
268:     /// </summary>
269:     public int Stop() {
```

```
270:         Globals.LogIDE(i18n.str("StoppingPrj", Name));
271:         Globals.SetupNewProgrammerLog();
272:         int r = Programmer.GetProgrammerApp().Stop(this);
273:         if(r == 0) IsRunning = false;
274:         return r;
275:     }
276:
277:     ~Project() {
278:         if(IsRunning) Stop();
279:     }
280:
281:     #region static members
282:     static List<Project> _OpenProjects = new List<Project>();
283:
284:     /// <summary>
285:     /// Obtiene la lista de proyectos abiertos actualmente
286:     /// </summary>
287:     public static Project[] OpenProjects {
288:         get {
289:             return _OpenProjects.ToArray();
290:         }
291:     }
292:
293:     /// <summary>
294:     /// Abre un proyecto desde un archivo y lo añade a la lista de proyectos abiertos
295:     /// </summary>
296:     /// <param name="PrjFile">Archivo de proyecto</param>
297:     public static void Open(string PrjFile) {
298:         if(File.Exists(PrjFile)) {
299:             Project np = LoadFromFile(PrjFile);
300:             if(np != null) Add(np);
301:         } else {
302:             Globals.LogIDE(i18n.str("FileNotFound", PrjFile));
303:         }
304:     }
305:
306:     /// <summary>
307:     /// Añade un proyecto existente a la lista de proyectos abiertos
308:     /// </summary>
```

```
309:      /// <param name="Prj">Project being added to the list</param>
310:      public static void Add(Project Prj) {
311:          Globals.Debug("Adding an existing project to the list of open projects");
312:          _OpenProjects.Add(Prj);
313:          Settings.AddOpenProject(Prj.PrjFilePath);
314:          if(Globals.MainWindow != null) Globals.MainWindow.UpdatePrjList();
315:      }
316:
317:      /// <summary>
318:      /// Carga un proyecto desde un archivo
319:      /// </summary>
320:      /// <param name="PrjFile">Path to the project file</param>
321:      public static Project LoadFromFile(string PrjFile) {
322:          Project NewPrj = null;
323:          FileInfo fi = new FileInfo(PrjFile);
324:          if(fi.Exists) {
325:              Globals.Debug("Loading project file: " + PrjFile);
326:              try {
327:                  XmlSerializer xs = new XmlSerializer(typeof(Project));
328:                  NewPrj = (Project)xs.Deserialize(new StreamReader(PrjFile));
329:                  NewPrj.PrjFilePath = PrjFile;
330:                  foreach(SrcFile src in NewPrj.SrcFiles) {
331:                      src.Modified = false;
332:                  }
333:              } catch(XmlException e) {
334:                  Globals.LogIDE(i18n.str("UnableLoadPrjFile", PrjFile, e.Message));
335:              }
336:          } else Globals.LogIDE(i18n.str("PrjFileNotExist", PrjFile));
337:          return NewPrj;
338:      }
339:      #endregion
340:  }
341: }
```

```
1: using System;
2:
3: namespace CuruxaIDE {
4:     public static class ProjectArrayExtensions {
5:         /// <summary>
6:         /// Devuelve un proyecto desde un array dado su nombre
7:         /// </summary>
8:         public static Project GetByName(this Project[] prjs, string Name) {
9:             foreach(Project prj in prjs) {
10:                 if(prj.Name == Name) return prj;
11:             }
12:             throw new ArgumentException("Unknown project");
13:         }
14:
15:         /// <summary>
16:         /// Comprueba si un proyecto está ya abierto, dada la ruta completa al archivo de proyecto
17:         /// </summary>
18:         public static bool ContainsPrjPath(this Project[] prjs, string PrjPath) {
19:             foreach(Project prj in prjs) {
20:                 if(prj.PrjFilePath == PrjPath) return true;
21:             }
22:             return false;
23:         }
24:     }
25: }
```

```

1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.Diagnostics;
6:
7: namespace CuruxaIDE {
8:     /// <summary>
9:     /// Small Device C Compiler
10:    /// </summary>
11:    public sealed class SDCC:Toolsuite {
12:        public SDCC()
13:            : base("SDCC", "sdcc") {
14:            GetIncludePaths();
15:
16:            SetupEnvironment();
17:        }
18:
19:        private void SetupEnvironment() {
20:            string SdccInstallPath = Environment.GetEnvironmentVariable("ProgramFiles") + @"\SDCC";
21:            string GputilsInstallPath = Environment.GetEnvironmentVariable("ProgramFiles") + @"\gputils";
22:
23:            //on Windows, SDCCand GPUTILS binary directories are only set in the $PATH of the user who
24:            //installed them. We try to fix it
25:            if(Environment.OSVersion.Platform != PlatformID.Unix) {
26:                //add "%ProgramFiles%\SDCC\bin" to the PATH
27:                Environment.SetEnvironmentVariable("PATH", Environment.GetEnvironmentVariable("PATH") + ";" + Environment.GetEnvironmentVariable("ProgramFiles") + @"\SDCC\bin", EnvironmentVariableTarget.Process);
28:
29:                //add "%ProgramFiles%\gputils\bin" to the PATH
30:                Environment.SetEnvironmentVariable("PATH", Environment.GetEnvironmentVariable("PATH") + ";" + Environment.GetEnvironmentVariable("ProgramFiles") + @"\gputils\bin", EnvironmentVariableTarget.Process);
31:
32:                //SDCC can't find its own libraries by itself on Windows!! :-(
33:                //args += " -I \"" + Environment.GetEnvironmentVariable("ProgramFiles") + @"\SDCC\
34:                //include\"";
35:
36:                //gpasm can't find its own header files

```



```

35:                                     //args += " -Wa_I\" + Environment.GetEnvironmentVariable("ProgramFiles") + "\\gpu
tils\\header\"";
36:
37:                                     Environment.SetEnvironmentVariable("SDCC_HOME", SdccInstallPath);
38:                                     //Environment.SetEnvironmentVariable("SDCC_INCLUDE", SdccInstallPath+"\\include");
39:                                     //Environment.SetEnvironmentVariable("SDCC_LIB", ???);
40:                                     Environment.SetEnvironmentVariable("GPUTILS_HEADER_PATH", GputilsInstallPath + @"\"
header");
41:                                     Environment.SetEnvironmentVariable("GPUTILS_LKR_PATH", GputilsInstallPath + @"\\lkr
");
42:                                     //Environment.SetEnvironmentVariable("GPUTILS_LIB_PATH", SdccInstallPath);
43:                                     }
44:                                     }
45:
46:                                     /// <summary>
47:                                     /// Obtiene la lista de directorio de inclusión globales
48:                                     /// </summary>
49:                                     /// <remarks>
50:                                     /// Esta lista se obtiene con el comando: "sdcc --print-search-dirs"
51:                                     /// </remarks>
52:                                     private void GetIncludePaths() {
53:                                         IncludePaths.Clear();
54:                                         ProcessStartInfo ProcInfo;
55:                                         Process proc;
56:                                         try {
57:                                             string args = "--print-search-dirs";
58:                                             ProcInfo = new ProcessStartInfo(Command, args);
59:                                             ProcInfo.CreateNoWindow = true;
60:                                             ProcInfo.ErrorDialog = false;
61:                                             ProcInfo.RedirectStandardOutput = true;
62:                                             ProcInfo.UseShellExecute = false;
63:                                             proc = new Process();
64:                                             proc.StartInfo = ProcInfo;
65:                                             //proc.OutputDataReceived += new DataReceivedEventHandler(proc_GetIncPathStdOut);
66:                                             Globals.Debug("Executing: {0} {1}", Command, args);
67:                                             proc.Start();
68:                                             proc.WaitForExit(8000); //wait 8 seconds
69:                                             Globals.Debug("Reading outputs");
70:

```

```

71:         bool ReadingIncludeDir = false;
72:         foreach(string OutLine in proc.StandardOutput.ReadToEnd().Split('\n', '\r')) {
73:             //detectar inicio de directorios de inclusión
74:             if(OutLine.Contains("includedir:")) {
75:                 ReadingIncludeDir = true;
76:                 continue;
77:             }
78:
79:             //detectar fin de directorios de inclusión
80:             if(ReadingIncludeDir && OutLine.Contains(":") && (!OutLine.Contains("/") |
| !OutLine.Contains(@"\"))) {
81:                 return;
82:             }
83:
84:             //parsea un directorio de inclusión
85:             if(ReadingIncludeDir) {
86:                 Globals.Debug("New SDCC global include path: " + OutLine);
87:                 IncludePaths.Enqueue(OutLine);
88:
89:                 string AnotherIncludeDir=(OutLine + System.IO.Path.DirectorySepara
torChar + "/pic").Replace("//", "/").Replace(@"\" , @"\");
90:                 Globals.Debug("New SDCC global include path: " + AnotherIncludeDir
);
91:                 IncludePaths.Enqueue(AnotherIncludeDir);
92:             }
93:         }
94:
95:         Globals.LogBuild(LogBuild);
96:     } catch(System.ComponentModel.Win32Exception) {
97:         string msg = i18n.str("NoApp", RealName);
98:         Globals.LogIDE(msg);
99:         Globals.LogBuild(msg);
100:     }
101: }
102:
103: private string LogBuild;
104:
105: public override int Build(Project prj) {
106:     ProcessStartInfo ProcInfo;

```

```
107:         Process proc;
108:
109:         if(prj.MainFile == null) {
110:             Globals.LogIDE(i18n.str("NoMainSrc"));
111:             return 1;
112:         }
113:
114:         string OldDir = Environment.CurrentDirectory;
115:         Environment.CurrentDirectory = prj.Path;
116:
117:         string args = "-I \"" + Settings.IncludeDir + "\" --Werror";
118:
119:         args += " -mpic14 -" + prj.MainBoard.GetMCU().ToString().ToLowerInvariant().Replace("pic", ↵
"p") + " \"" + prj.MainFile.FullPath + "\" -o temp";
120:         //SDCC segmentation fault cuando el nombre del archivo contiene espacios
121:
122:         try {
123:             LogBuild = "";
124:             Globals.SetupNewBuildLog();
125:
126:             ProcInfo = new ProcessStartInfo(Command, args);
127:             ProcInfo.CreateNoWindow = true;
128:             ProcInfo.ErrorDialog = false;
129:             ProcInfo.RedirectStandardError = true;
130:             ProcInfo.RedirectStandardOutput = true;
131:             ProcInfo.UseShellExecute = false;
132:             proc = new Process();
133:             proc.StartInfo = ProcInfo;
134:             proc.OutputDataReceived += new DataReceivedEventHandler(proc_OutputDataReceived);
135:             proc.ErrorDataReceived += new DataReceivedEventHandler(proc_ErrorDataReceived);
136:             Globals.Debug("Executing: {0} {1}", Command, args);
137:             proc.Start();
138:             Globals.Debug("Reading outputs");
139:             proc.BeginErrorReadLine();
140:             proc.BeginOutputReadLine();
141:             Globals.Debug("Waiting for exit...");
142:             proc.WaitForExit();
143:
144:             Globals.LogBuild(LogBuild);
```

```
145:
146:         Environment.CurrentDirectory = OldDir;
147:         return proc.ExitCode;
148:     } catch(System.ComponentModel.Win32Exception) {
149:         string msg = i18n.str("NoApp", RealName);
150:         Globals.LogIDE(msg);
151:         Globals.LogBuild(msg);
152:         Environment.CurrentDirectory = OldDir;
153:         return 1;
154:     }
155: }
156:
157: void proc_ErrorDataReceived(object sender, DataReceivedEventArgs e) {
158:     lock(LogBuild) {
159:         if(string.IsNullOrEmpty(e.Data)) return;
160:         for(int i = 0; i <= e.Data.Length; i++) {
161:             if(i == e.Data.Length) return;
162:             if(e.Data[i] != ' ') break;
163:         }
164:         //Globals.LogBuild "[" + RealName + " Error] " + e.Data);
165:         LogBuild += "[" + RealName + " Error] " + e.Data + Environment.NewLine;
166:     }
167: }
168:
169: void proc_OutputDataReceived(object sender, DataReceivedEventArgs e) {
170:     lock(LogBuild) {
171:         //elimina mensajes vacíos
172:         if(string.IsNullOrEmpty(e.Data)) return;
173:         for(int i = 0; i <= e.Data.Length; i++) {
174:             if(i == e.Data.Length) return;
175:             if(e.Data[i] != ' ') break;
176:         }
177:
178:         //oculta mensajes indeseados
179:         if(e.Data.Contains("message: using default linker script")) return;
180:
181:         //Globals.LogBuild "[" + RealName + "]" + e.Data);
182:         LogBuild += "[" + RealName + "]" + e.Data + Environment.NewLine;
183:     }
```

```
184:      }  
185:    }  
186: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.Xml.Serialization;
6: using System.IO;
7: using System.Drawing;
8:
9: namespace CuruxaIDE {
10:     /// <summary>
11:     /// Opciones del programa
12:     /// </summary>
13:     public class Settings {
14:         //miembros de instancias de un objeto son los que se almacenan en el archivo de configuración
15:         #region instance members
16:         public List<string> _OpenProjects = new List<string>();
17:         public System.Windows.Forms.FormWindowState _MainWinState = System.Windows.Forms.FormWindowState.Normal;
18:         public bool _EnableSyntaxHighlight = (Environment.OSVersion.Platform == PlatformID.Unix) ? false : true;
19:         #endregion
20:
21:         #region static members
22:         static Settings s = new Settings();
23:
24:         /// <summary>
25:         /// Codificación de caracteres usada por Curuxa IDE
26:         /// </summary>
27:         public static readonly Encoding CharSet = Encoding.UTF8;
28:
29:         public const string Version = "0.3-svn";
30:
31:         public static Color PrjListBackColor = Color.White;
32:         public static Color PrjListSettingsColor = Color.Blue;
33:         public static Color PrjListSrcColor = Color.Black;
34:         public static Color PrjListLibsColor = Color.Gray;
35:
36:         /// <summary>
37:         /// Añade la ruta del proyecto a la lista de proyectos abiertos
```

```
38:         /// </summary>
39:         /// <param name="path"></param>
40:         public static void AddOpenProject(string path) {
41:             if(!s._OpenProjects.Contains(path)) {
42:                 Globals.Debug("Adding to the list of open projects: " + path);
43:                 s._OpenProjects.Add(path);
44:             }
45:         }
46:
47:         /// <summary>
48:         /// Elimina la ruta a un proyecto de la lista de proyectos abiertos
49:         /// </summary>
50:         /// <param name="path"></param>
51:         public static void RemoveOpenProject(string path) {
52:             Globals.Debug("Removing path from the list of opened projects (at the settings file): " + path);
53:             s._OpenProjects.Remove(path);
54:         }
55:
56:         /// <summary>
57:         /// Ruta completa al ejecutable de la aplicación
58:         /// </summary>
59:         public static string ExePath {
60:             get {
61:                 if(_ExePath == null) _ExePath = System.Reflection.Assembly.GetEntryAssembly().Location;
62:                 return _ExePath;
63:             }
64:         }
65:         private static string _ExePath;
66:
67:         /// <summary>
68:         /// Ruta completa al directorio que contiene el ejecutable de la aplicación. El último carácter NO
69:         /// es una barra (\'')
70:         /// </summary>
71:         public static string ExeLocation {
72:             get {
73:                 if(_ExeLocation == null) _ExeLocation = System.IO.Path.GetDirectoryName(ExePath);
74:                 return _ExeLocation;
75:             }
76:         }
77:         private static string _ExeLocation;
```

```
74:         }
75:     }
76:     private static string _ExeLocation;
77:
78:     /// <summary>
79:     /// Ruta completa al directorio que contiene librerías de terceros y programas instalados junto co
n Curuxa IDE
80:     /// </summary>
81:     public static string ThirdPartyDir {
82:         get {
83:             return ExeLocation + "/ThirdParty";
84:         }
85:     }
86:
87:     /// <summary>
88:     /// Ruta a los binarios de pk2cmd-windows
89:     /// </summary>
90:     public static string Pk2cmdWinDir {
91:         get {
92:             return ThirdPartyDir + "/pk2cmd-win";
93:         }
94:     }
95:
96:     /// <summary>
97:     /// Ruta completa al directorio que contiene los archivos de traducción. El último caracter NO es
una barra ('/')
98:     /// </summary>
99:     public static string LanguageFilesDir {
100:         get {
101:             return ExeLocation + "/i18n";
102:         }
103:     }
104:
105:     /// <summary>
106:     /// Ruta completa al directorio que contiene las plantillas de código. El último caracter NO es un
a barra ('/')
107:     /// </summary>
108:     public static string TemplatesDir {
109:         get {
```



```
110:                 return ExeLocation + "/Templates";
111:             }
112:         }
113:
114:         /// <summary>
115:         /// Ruta completa al directorio que contiene los ejemplos. El último caracter NO es una barra ('/' ↗
116:         /// </summary>
117:         public static string ExamplesDir {
118:             get {
119:                 return ExeLocation + "/Examples";
120:             }
121:         }
122:
123:         /// <summary>
124:         /// Ruta completa al directorio que contiene las imágenes usadas en este programa. El último caract ↗
125:         /// </summary>
126:         public static string ImagesDir {
127:             get {
128:                 return ExeLocation + "/Images";
129:             }
130:         }
131:
132:         public static System.Windows.Forms.FormWindowState MainWindowState {
133:             get {
134:                 return s._MainWinState;
135:             }
136:             set {
137:                 s._MainWinState = value;
138:             }
139:         }
140:
141:         public static bool EnableSyntaxHighlight {
142:             get {
143:                 return s._EnableSyntaxHighlight;
144:             }
145:             set {
146:                 s._EnableSyntaxHighlight = value;
```

```

147:         }
148:     }
149:
150:     /// <summary>
151:     /// Ruta completa al directorio que contiene librerías usadas por los usuarios finales. El último
caracter NO es una barra ('/')
152:     /// </summary>
153:     public static string IncludesDir {
154:         get {
155:             if(string.IsNullOrEmpty(_IncludesDir)) {
156:                 string[] PossibleLocations = {
157:                     "/usr/share/curuxa/lib", //default install path
158:                     ExeLocation + "../../../lib", //relative path when debugging (run
ning Curuxa IDE without installation)
159:                     ExeLocation + "../lib", //non-standard install path
160:                     ExeLocation + "/lib", //non-standard install path
161:                     ExeLocation //aaaarrrrrrggggg!!!!
162:                 };
163:                 foreach(string L in PossibleLocations) {
164:                     Globals.Debug("Looking for libraries in " + L);
165:                     if(File.Exists(L + "/MBP.h")) {
166:                         _IncludesDir = L;
167:                     }
168:                 }
169:             }
170:             if(string.IsNullOrEmpty(_IncludesDir)) Globals.LogIDE(i18n.str("LibsNotFound"));
171:             return _IncludesDir;
172:         }
173:     }
174:     protected static string _IncludesDir;
175:
176:     /// <summary>
177:     /// Ruta al directorio donde todas las configuraciones del usuario y archivos temporales se almace
nan. El último caracter NO es una barra ('/')
178:     /// </summary>
179:     public static string AppDataPath {
180:         get {
181:             //elegir directorio de configuración
182:             if(_AppDataPath == null) {

```

```
183:             Globals.Debug("Looking for the user settings folder");
184:             if(Environment.OSVersion.Platform == PlatformID.Unix) {
185:                 _AppDataPath = Environment.GetEnvironmentVariable("HOME") + "/.curuxa";
186:             } else {
187:                 _AppDataPath = "C:" + Environment.GetEnvironmentVariable("HOMEPATH") + "\\curuxa";
188:             }
189:             Globals.Debug("User settings folder: {0}", _AppDataPath);
190:         }
191:         return _AppDataPath;
192:     }
193: }
194: private static string _AppDataPath;
195:
196: public static string SettingsFile {
197:     get {
198:         if(_SettingsFile == null) {
199:             _SettingsFile = AppDataPath + "/CuruxaIDE.conf";
200:         }
201:         return _SettingsFile;
202:     }
203: }
204: private static string _SettingsFile;
205:
206: public static void Save() {
207:     if(Globals.MainWindow != null) s._MainWinState = Globals.MainWindow.WindowState;
208:
209:     Globals.Debug("Saving settings file: " + SettingsFile);
210:     if(!Directory.Exists(AppDataPath)) {
211:         try {
212:             Directory.CreateDirectory(AppDataPath);
213:         } catch(UnauthorizedAccessException) {
214:             Globals.LogIDE("Can't create directory: " + AppDataPath);
215:         }
216:     }
217:     StreamWriter sw = null;
218:     try {
219:         sw = new StreamWriter(SettingsFile);
```

```
220:                 XmlSerializer xs = new XmlSerializer(s.GetType());
221:                 xs.Serialize(sw, s);
222:             } catch (IOException e) {
223:                 Globals.LogIDE(i18n.str("UnableSaveSettings"));
224:                 Globals.Debug(e.Message);
225:             } finally {
226:                 if (sw != null) sw.Close();
227:             }
228:         }
229:
230:         public static void Load() {
231:             FileInfo fi = new FileInfo(SettingsFile);
232:             if (fi.Exists) {
233:                 Globals.Debug("Loading settings file: " + SettingsFile);
234:                 try {
235:                     XmlSerializer xs = new XmlSerializer(s.GetType());
236:                     s = (Settings)xs.Deserialize(new StreamReader(SettingsFile));
237:                     foreach (string PrjPath in s._OpenProjects) {
238:                         Project.Open(PrjPath);
239:                     }
240:                 } catch (IOException e) {
241:                     Globals.LogIDE(i18n.str("UnableLoadSettings"));
242:                     Globals.Debug(e.Message);
243:                 }
244:             }
245:         }
246:         #endregion
247:     }
248: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace CuruxaIDE {
7:     /// <summary>
8:     /// Aplicación informática
9:     /// </summary>
10:    public abstract class SoftApp {
11:        public readonly string RealName;
12:        public readonly string Command;
13:
14:        protected SoftApp(string RealName, string Command) {
15:            this.RealName=RealName;
16:            this.Command=Command;
17:        }
18:    }
19: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.IO;
6:
7: namespace CuruxaIDE {
8:     /// <summary>
9:     /// Representa un archivo que contiene código fuente
10:    /// </summary>
11:    public class SrcFile {
12:        public static List<SrcFile> KnownSrcFiles = new List<SrcFile>();
13:
14:        public readonly Project ParentProject;
15:
16:        /// <summary>
17:        /// Nombre del archivo, incluyendo extensión, excluyendo ruta
18:        /// </summary>
19:        public string FileName { get; protected set; }
20:
21:        /// <summary>
22:        /// Indica si este archivo es de sólo lectura
23:        /// </summary>
24:        public bool IsReadOnly {
25:            get {
26:                if(!IsLocal || new FileInfo(FullPath).IsReadOnly) return true;
27:                else return false;
28:            }
29:        }
30:
31:        /// <summary>
32:        /// Indica si el contenido se ha modificado pero no se ha guardado
33:        /// </summary>
34:        public bool Modified {
35:            get {
36:                return _Modified;
37:            }
38:            set {
39:                if(value == true) {
```

```

40:         if(ParentProject != null) ParentProject.Modified = true;
41:     }
42:     if(_Modified == false && value == true) {
43:         //(código eliminado. Antes hacía algo aquí)
44:     }
45:     _Modified = value;
46: }
47: }
48: private bool _Modified = false;
49:
50: /// <summary>
51: /// Indica si este archivo es local o no (misma ruta que el proyecto)
52: /// </summary>
53: public bool IsLocal {
54:     get {
55:         if(ParentProject != null && ParentProject.Path == SrcPath) return true;
56:         else return false;
57:     }
58: }
59:
60: /// <summary>
61: /// Ruta completa al archivo (ruta al directorio + nombre del archivo + extensión)
62: /// </summary>
63: public string FullPath {
64:     get {
65:         return Path.GetFullPath(SrcPath + "/" + FileName);
66:     }
67: }
68:
69: /// <summary>
70: /// Directorio donde se encuentra este archivo
71: /// </summary>
72: /// NOTA PARA DESARROLLADORES: deja _SrcPath vacío para obtener la misma ruta que el proyecto padre
73: public string SrcPath {
74:     get {
75:         if(_SrcPath == null) {
76:             if(ParentProject == null) return "";
77:             else return ParentProject.Path;

```

```
78:                 } else {
79:                     return _SrcPath;
80:                 }
81:             }
82:             protected set {
83:                 _SrcPath = value;
84:             }
85:         }
86:         protected string _SrcPath;
87:
88:         /// <summary>
89:         /// Nombre del archivo, excluyendo ruta y extensión
90:         /// </summary>
91:         public string Name {
92:             get {
93:                 return new FileInfo(FileName).Name;
94:             }
95:         }
96:
97:         /// <summary>
98:         /// Extensión del archivo sin punto
99:         /// </summary>
100:        public string Extension {
101:            get {
102:                return new FileInfo(FileName).Extension.Remove(0, 1);
103:            }
104:        }
105:
106:        public Language Language {
107:            get {
108:                return LanguageExtensions.GetFromExtension(Extension);
109:            }
110:        }
111:
112:        /// <summary>
113:        /// Crea un nuevo archivo de código fuente vacío en la ruta seleccionada
114:        /// </summary>
115:        /// <param name="FileName">Nombre del archivo, incluyendo extensión, excluyendo ruta</param>
116:        public SrcFile(Project ParentProject, string FileName)
```



```
117:         : this(FileName) {
118:             this.ParentProject = ParentProject;
119:         }
120:
121:         /// <summary>
122:         /// Crea un nuevo archivo de código fuente
123:         /// </summary>
124:         /// <param name="FileName">Nombre del archivo, incluyendo extensión, excluyendo ruta</param>
125:         protected SrcFile(string FileName) {
126:             this.FileName = FileName;
127:             KnownSrcFiles.Add(this);
128:         }
129:
130:         /// <summary>
131:         /// Crea un nuevo archivo de código vacío en la ruta dada a partir de una plantilla
132:         /// </summary>
133:         /// <param name="FileName">Nombre del archivo, incluyendo extensión, excluyendo ruta</param>
134:         public static SrcFile GetFromTemplate(Project ParentProject, string FileName, MainBoard MainBoard,
Language Language) {
135:             SrcFile src = new SrcFile(ParentProject, FileName);
136:             string file = string.Format("{0}/{1}.{2}", Settings.TemplatesDir, MainBoard, Language.GetE
xtension());
137:             if(!File.Exists(file)) file = string.Format("{0}/{1}.{2}", Settings.TemplatesDir, "Default
", Language.GetExtension());
138:             src.Content = File.ReadAllText(file, Settings.Charset);
139:             return src;
140:         }
141:
142:         /// <summary>
143:         /// Obtiene la instancia de un SrcFile que representa cierta librería
144:         /// </summary>
145:         /// <param name="type">Tipo de librerías (local o global)</param>
146:         /// <param name="LibName">Nombre de la librería</param>
147:         /// <param name="RefFrom">Archivo de código que hace referencia a esta librería</param>
148:         /// <returns>
149:         /// Una instancia de un SrcFile que representa la librería.
150:         /// Null si la librería no se encuentra.
151:         /// </returns>
152:         public static SrcFile GetFromLibrary(IncludeType type, string LibName, SrcFile RefFrom) {
```

```
153:         SrcFile src = new SrcFile(RefFrom.ParentProject, LibName);
154:         //src.IsReadOnly = true;
155:
156:         //encontrar ruta a la librería
157:         Queue<string> PossiblePaths = new Queue<string>();
158:         if(type == IncludeType.Local) PossiblePaths.Enqueue(RefFrom.SrcPath);
159:         PossiblePaths.Enqueue(Settings.IncludeDir);
160:         foreach(string path in RefFrom.ParentProject.Language.GetToolsuite().IncludePaths) {
161:             PossiblePaths.Enqueue(path);
162:         }
163:         while(PossiblePaths.Count > 0) {
164:             string SomePath = PossiblePaths.Dequeue();
165:             string SomeFullPath = Path.GetFullPath(SomePath + "/" + LibName);
166:             if(File.Exists(SomeFullPath)) {
167:                 //library found
168:                 src.SrcPath = Path.GetDirectoryName(SomeFullPath);
169:                 break;
170:             }
171:         }
172:
173:         if(!string.IsNullOrEmpty(src._SrcPath)) {
174:             if(KnownSrcFiles.ContainsFullPath(src.FullPath)) {
175:                 //Return a previously known object instead of the newly created one
176:                 return KnownSrcFiles.GetByFullPath(src.FullPath);
177:             }
178:
179:             src.Content = File.ReadAllText(src.FullPath, Settings.Charset);
180:             return src;
181:         } else {
182:             //devuelve null si no se encuentra la librería
183:             return null;
184:         }
185:     }
186:
187:     /// <summary>
188:     /// Librerías referenciadas desde este archivo de código fuente
189:     /// </summary>
190:     public SrcFile[] RefLibs {
191:         get {
```

```
192:         List<SrcFile> libs = new List<SrcFile>();
193:         //libs.Add(SrcFile.GetFromLibrary(IncludeType.Global, "MBP18.h", this));
194:
195:         foreach(string line in Content.Split('\n', '\r')) {
196:             if(line.Length < 10) continue;
197:             if(line.StartsWith("#include")) {
198:                 IncludeType type = IncludeType.Global;
199:                 string lib = "";
200:                 if(line.Contains('<') && line.Contains('>')) {
201:                     //this is a #include <lib.h> line
202:                     lib = line.Replace("#include", "");
203:                     lib = lib.TrimStart(' ', '\t');
204:                     lib = lib.TrimEnd(' ', '\t');
205:                     lib = lib.TrimStart('<');
206:                     lib = lib.TrimEnd('>');
207:                 } else if(line.Contains('"')) {
208:                     //this is a #include "lib.h" line
209:                     lib = line.Replace("#include", "");
210:                     lib = lib.TrimStart(' ', '\t');
211:                     lib = lib.TrimEnd(' ', '\t');
212:                     lib = lib.TrimStart('"');
213:                     lib = lib.TrimEnd('"');
214:                     type = IncludeType.Local;
215:                 }
216:                 SrcFile NewLib = SrcFile.GetFromLibrary(type, lib, this);
217:                 if(NewLib != null) libs.Add(NewLib);
218:                 else Globals.LogIDE(i18n.str("LibNotFound", lib));
219:             }
220:         }
221:
222:         return libs.ToArray();
223:     }
224: }
225:
226: /// <summary>
227: /// Contenido del archivo
228: /// </summary>
229: public string Content {
230:     get {
```

```
231:             if(string.IsNullOrEmpty(_Content)) ReadFile();
232:             return _Content;
233:         }
234:         set {
235:             _Content = value;
236:             Modified = true;
237:         }
238:     }
239:     protected string _Content;
240:
241:     /// <summary>
242:     /// Lee el contenido del archivo
243:     /// </summary>
244:     public void ReadFile() {
245:         try {
246:             Content = File.ReadAllText(FullPath, Settings.Charset);
247:             Modified = false;
248:         } catch(FileNotFoundException) {
249:             Globals.LogIDE(i18n.str("FileNotFound", FullPath));
250:         }
251:     }
252:
253:     /// <summary>
254:     /// Guarda el contenido del archivo a disco
255:     /// </summary>
256:     public void SaveFile() {
257:         if(IsReadOnly) {
258:             Globals.LogIDE(i18n.str("CantSaveIsReadOnly", FullPath));
259:         } else {
260:             Globals.LogIDE(i18n.str("SavingFile", FullPath));
261:             File.WriteAllText(FullPath, Content, Settings.Charset);
262:             Modified = false;
263:             if(Globals.MainWindow != null) Globals.MainWindow.UpdatePrjList();
264:         }
265:     }
266:
267:     public static bool operator ==(SrcFile a, SrcFile b) {
268:         if(object.ReferenceEquals(a, b)) return true;
269:         else if((object)a == null || (object)b == null) return false;
```

```
270:                else return a.FullPath == b.FullPath;
271:            }
272:
273:            public static bool operator !=(SrcFile a, SrcFile b) {
274:                return !(a == b);
275:            }
276:        }
277: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Drawing;
5: using System.Data;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9:
10: namespace CuruxaIDE {
11:     /// <summary>
12:     /// Gestiona un conjunto de pestañas que contienen editores de código
13:     /// </summary>
14:     public partial class SrcTabControl:TabControl {
15:         public Dictionary<SrcFile, SrcTabPage> Tabs = new Dictionary<SrcFile, SrcTabPage>();
16:
17:         public SrcTabControl()
18:             : base() {
19:         }
20:
21:         /// <summary>
22:         /// Abre un archivo de código. No importa si el archivo ya está abierto
23:         /// </summary>
24:         public void OpenSrc(SrcFile src) {
25:             if(!Tabs.ContainsKey(src)) {
26:                 SrcTabPage NewTab = new SrcTabPage(src);
27:                 Tabs.Add(src, NewTab);
28:                 TabPages.Add(NewTab);
29:             }
30:             SelectedTab = Tabs[src];
31:         }
32:
33:         internal void RemoveAllFromPrj(Project p) {
34:             bool mod;
35:             do {
36:                 mod = false;
37:                 foreach(KeyValuePair<SrcFile, SrcTabPage> pair in Tabs) {
38:                     if(pair.Key.ParentProject == p) {
39:                         i18n.str("ClosingTab", pair.Key.ParentProject.Name, pair.Value.Name);
```

```
e);
40:                                     CloseSrc(pair.Key);
41:                                     mod = true;
42:                                     break;
43:                                 }
44:                             }
45:                         } while(mod);
46:                     }
47:
48:     public void CloseSrc(SrcFile src) {
49:         if(src != null && Tabs.ContainsKey(src)) {
50:             //there is a weird bug here I can't fix. It happens on Mono+Linux
51:             if(TabPages.Contains(Tabs[src])) TabPages.Remove(Tabs[src]);
52:             Tabs.Remove(src);
53:         }
54:     }
55: }
56: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Diagnostics;
5: using System.Linq;
6: using System.Text;
7: using System.Windows.Forms;
8: using System.Drawing;
9:
10: namespace CuruxaIDE {
11:     public partial class SrcTabPage:TabPage {
12:         public readonly SrcFile src;
13:         public SyntaxRichTextBox TxtCode;
14:
15:         public SrcTabPage(SrcFile src)
16:             : base() {
17:             this.src = src;
18:             UpdateTitle();
19:             Enter += new EventHandler(SrcTabPage_Enter);
20:             TxtCode = new SyntaxRichTextBox(src.Language);
21:             TxtCode.Parent = this;
22:             TxtCode.Dock = DockStyle.Fill;
23:             TxtCode.AcceptsTab = true;
24:             TxtCode.Font = new Font("Courier New", 10f, FontStyle.Regular, GraphicsUnit.Point, ((byte)0));
25:             if(src.IsReadOnly) {
26:                 TxtCode.ReadOnly = true;
27:                 TxtCode.BackColor = SystemColors.Window;
28:             }
29:             TxtCode.Text = src.Content;
30:             TxtCode.ProcessAllLines();
31:             TxtCode.TextChanged += new EventHandler(TxtCode_TextChanged);
32:             TxtCode.SelectionChanged += new EventHandler(TxtCode_SelectionChanged);
33:         }
34:
35:         void SrcTabPage_Enter(object sender, EventArgs e) {
36:             Globals.ActiveSrcFile = src;
37:         }
38:     }
```



```
39:         void TxtCode_SelectionChanged(object sender, EventArgs e) {
40:             Point pt = new Point();
41:             CursorLocation c;
42:             int index = TxtCode.SelectionStart;
43:             c.Line = TxtCode.GetLineFromCharIndex(index) + 1;
44:             pt = TxtCode.GetPositionFromCharIndex(index);
45:             pt.X = 0;
46:             c.Column = index - TxtCode.GetCharIndexFromPosition(pt) + 1;
47:             Globals.MainWindow.CursorLocationChanged(c);
48:         }
49:
50:         void TxtCode_TextChanged(object sender, EventArgs e) {
51:             bool m = src.Modified;
52:             src.Content = TxtCode.Text;
53:             if(m == false) UpdateTitle();
54:             TxtCode.ProcessChangedText();
55:         }
56:
57:         public void UpdateTitle() {
58:             string title = src.Name;
59:             if(src.IsReadOnly) title += i18n.str("TitleReadOnly");
60:             if(src.Modified) title += " *";
61:             Text = title;
62:         }
63:     }
64: }
```

```
1: using System;
2:
3: namespace CuruxaIDE {
4:     public static class StringExtensions {
5:         public static bool IsNumber(this string str) {
6:             foreach(char c in str) {
7:                 if(!char.IsNumber(c)) return false;
8:             }
9:             return true;
10:        }
11:
12:        /// <summary>
13:        /// Elimina todo el texto desde cierto valor hasta el final
14:        /// </summary>
15:        public static string RemoveFrom(this string str, string value) {
16:            if(str.Contains(value)) return str.Remove(str.IndexOf(value));
17:            else return str;
18:        }
19:
20:        /// <summary>
21:        /// Elimina todo el texto desde cierto valor hasta el final
22:        /// </summary>
23:        public static string RemoveFrom(this string str, char value) {
24:            return RemoveFrom(str, value.ToString());
25:        }
26:    }
27: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Text;
4: using System.IO;
5: using System.Windows.Forms;
6: using System.ComponentModel;
7: using System.Text.RegularExpressions;
8: using System.Drawing;
9:
10: namespace CuruxaIDE {
11:     public class SyntaxRichTextBox: System.Windows.Forms.RichTextBox {
12:         public Language Lang;
13:         private SyntaxSettings m_settings = new SyntaxSettings();
14:         private static bool m_bPaint = true;
15:         private string m_strLine = "";
16:         private int m_nContentLength = 0;
17:         private int m_nLineLength = 0;
18:         private int m_nLineStart = 0;
19:         private int m_nLineEnd = 0;
20:         private string m_strKeywords = "";
21:         private int m_nCurSelection = 0;
22:
23:         public SyntaxRichTextBox(Language SrcLang) {
24:             SetLang(SrcLang);
25:         }
26:
27:         protected void SetLang(Language lang) {
28:             this.Lang = lang;
29:             switch(lang) {
30:                 case Language.C:
31:                     SetupLangC();
32:                     break;
33:                 case Language.PicAsm:
34:                     SetupLangPicasm();
35:                     break;
36:                 default:
37:                     throw new NotImplementedException("language not yet supported");
38:             }
39:         }
```

```
40:
41:         protected void SetupLangC() {
42:             Settings.Comment = "//";
43:             Settings.Keywords.Clear();
44:             Settings.Keywords.Add("auto");
45:             Settings.Keywords.Add("_Bool");
46:             Settings.Keywords.Add("break");
47:             Settings.Keywords.Add("case");
48:             Settings.Keywords.Add("char");
49:             Settings.Keywords.Add("_Complex");
50:             Settings.Keywords.Add("const");
51:             Settings.Keywords.Add("continue");
52:             Settings.Keywords.Add("default");
53:             Settings.Keywords.Add("do");
54:             Settings.Keywords.Add("double");
55:             Settings.Keywords.Add("else");
56:             Settings.Keywords.Add("enum");
57:             Settings.Keywords.Add("extern");
58:             Settings.Keywords.Add("float");
59:             Settings.Keywords.Add("for");
60:             Settings.Keywords.Add("goto");
61:             Settings.Keywords.Add("if");
62:             Settings.Keywords.Add("_Imaginary");
63:             Settings.Keywords.Add("inline");
64:             Settings.Keywords.Add("int");
65:             Settings.Keywords.Add("long");
66:             Settings.Keywords.Add("register");
67:             Settings.Keywords.Add("restrict");
68:             Settings.Keywords.Add("return");
69:             Settings.Keywords.Add("short");
70:             Settings.Keywords.Add("signed");
71:             Settings.Keywords.Add("sizeof");
72:             Settings.Keywords.Add("static");
73:             Settings.Keywords.Add("struct");
74:             Settings.Keywords.Add("switch");
75:             Settings.Keywords.Add("typedef");
76:             Settings.Keywords.Add("union");
77:             Settings.Keywords.Add("unsigned");
78:             Settings.Keywords.Add("void");
```

```
79:         Settings.Keywords.Add("volatile");
80:         Settings.Keywords.Add("while");
81:         CompileKeywords();
82:     }
83:
84:     protected void SetupLangPicasa() {
85:         Settings.Comment = ";";
86:         Settings.Keywords.Clear();
87:         Settings.Keywords.Add("__badram");
88:         Settings.Keywords.Add("__bardrom");
89:         Settings.Keywords.Add("cblock");
90:         Settings.Keywords.Add("config");
91:         Settings.Keywords.Add("__config");
92:         Settings.Keywords.Add("constant");
93:         Settings.Keywords.Add("da");
94:         Settings.Keywords.Add("data");
95:         Settings.Keywords.Add("db");
96:         Settings.Keywords.Add("de");
97:         Settings.Keywords.Add("#define");
98:         Settings.Keywords.Add("dt");
99:         Settings.Keywords.Add("dw");
100:        Settings.Keywords.Add("else");
101:        Settings.Keywords.Add("end");
102:        Settings.Keywords.Add("endc");
103:        Settings.Keywords.Add("endif");
104:        Settings.Keywords.Add("endw");
105:        Settings.Keywords.Add("equ");
106:        Settings.Keywords.Add("fill");
107:        Settings.Keywords.Add("goto");
108:        Settings.Keywords.Add("__idlocs");
109:        Settings.Keywords.Add("if");
110:        Settings.Keywords.Add("ifdef");
111:        Settings.Keywords.Add("ifndef");
112:        Settings.Keywords.Add("include");
113:        Settings.Keywords.Add("#include");
114:        Settings.Keywords.Add("list");
115:        Settings.Keywords.Add("__maxram");
116:        Settings.Keywords.Add("__maxrom");
117:        Settings.Keywords.Add("org");
```

```
118:         Settings.Keywords.Add("processor");
119:         Settings.Keywords.Add("radix");
120:         Settings.Keywords.Add("res");
121:         Settings.Keywords.Add("set");
122:         Settings.Keywords.Add("#undefine");
123:         Settings.Keywords.Add("variable");
124:         Settings.Keywords.Add("while");
125:         Settings.Keywords.Add("error");
126:         Settings.Keywords.Add("errorlevel");
127:         Settings.Keywords.Add("list");
128:         Settings.Keywords.Add("messg");
129:         Settings.Keywords.Add("nolist");
130:         Settings.Keywords.Add("page");
131:         Settings.Keywords.Add("space");
132:         Settings.Keywords.Add("subtitle");
133:         Settings.Keywords.Add("title");
134:         Settings.Keywords.Add("endm");
135:         Settings.Keywords.Add("exitm");
136:         Settings.Keywords.Add("expand");
137:         Settings.Keywords.Add("local");
138:         Settings.Keywords.Add("macro");
139:         Settings.Keywords.Add("noexpand");
140:         Settings.Keywords.Add("bankisel");
141:         Settings.Keywords.Add("banksel");
142:         Settings.Keywords.Add("code");
143:         Settings.Keywords.Add("code_pack");
144:         Settings.Keywords.Add("extern");
145:         Settings.Keywords.Add("global");
146:         Settings.Keywords.Add("idata");
147:         Settings.Keywords.Add("idata_acs");
148:         Settings.Keywords.Add("pagesel");
149:         Settings.Keywords.Add("pageselw");
150:         Settings.Keywords.Add("udata");
151:         Settings.Keywords.Add("udata_acs");
152:         Settings.Keywords.Add("udata_ovr");
153:         Settings.Keywords.Add("udata_shr");
154:         CompileKeywords();
155:     }
156:
```

```
157:         /// <summary>
158:         /// Configuración del resaltador de sintaxis
159:         /// </summary>
160:         public SyntaxSettings Settings {
161:             get { return m_settings; }
162:         }
163:
164:         /// <summary>
165:         /// WndProc
166:         /// </summary>
167:         /// <param name="m"></param>
168:         protected override void WndProc(ref System.Windows.Forms.Message m) {
169:             if(m.Msg == 0x00f) {
170:                 if(m_bPaint) {
171:                     base.WndProc(ref m);
172:                 } else {
173:                     m.Result = IntPtr.Zero;
174:                 }
175:             } else {
176:                 base.WndProc(ref m);
177:             }
178:         }
179:
180:         public void ProcessChangedText() {
181:             if(!Settings.EnableSyntaxHighlight) return;
182:
183:             // Calculate shit here
184:             m_nContentLength = this.TextLength;
185:
186:             int nCurrentSelectionStart = SelectionStart;
187:             int nCurrentSelectionLength = SelectionLength;
188:
189:             m_bPaint = false;
190:
191:             m_nLineStart = nCurrentSelectionStart;
192:             while((m_nLineStart > 0) && (Text[m_nLineStart - 1] != '\n')) m_nLineStart--;
193:             m_nLineEnd = nCurrentSelectionStart;
194:             while((m_nLineEnd < Text.Length) && (Text[m_nLineEnd] != '\n')) m_nLineEnd++;
195:             m_nLineLength = m_nLineEnd - m_nLineStart;
```

```
196:
197:         // Get the current line.
198:         m_strLine = Text.Substring(m_nLineStart, m_nLineLength);
199:
200:         ProcessLine();
201:
202:         m_bPaint = true;
203:     }
204:
205:     /// <summary>
206:     /// Procesa una línea
207:     /// </summary>
208:     private void ProcessLine() {
209:         if(!Settings.EnableSyntaxHighlight) return;
210:
211:         // guarda la posición y hace que toda la línea sea negra
212:         int nPosition = SelectionStart;
213:         SelectionStart = m_nLineStart;
214:         SelectionLength = m_nLineLength;
215:         SelectionColor = Color.Black;
216:
217:         bool IsPPD = false;
218:
219:         // procesa los keywords
220:         ProcessRegex(m_strKeywords, Settings.KeywordColor);
221:
222:         //procesa directivas del preprocesador
223:         /*if(Settings.EnablePPD) {
224:             if(m_strLine[0]=='#') {
225:                 IsPPD = true;
226:                 SelectionStart = m_nLineStart;
227:                 SelectionLength = m_nLineLength-1;
228:                 SelectionColor = Settings.PPDColor;
229:             }
230:         */
231:
232:         // Procesa números
233:         if(Settings.EnableIntegers && !IsPPD) ProcessRegex(@"\\b(?:[0-9]*\\.)?[0-9]+\\b", Settings.
IntegerColor);
```



```

234:
235:         // Procesa textos
236:         if(Settings.EnableStrings && !IsPPD) ProcessRegex("\"[^\\"\\\\\\r\\n]*(?:\\\\\\\\.\\\"[^\\"\\\\\\r\\n]
n]*)*\\\"", Settings.StringColor);
237:
238:         // Procesa comentarios
239:         if(Settings.EnableComments && !string.IsNullOrEmpty(Settings.Comment) && !IsPPD) ProcessRe
gex(Settings.Comment + ".*$", Settings.CommentColor);
240:
241:         SelectionStart = nPosition;
242:         SelectionLength = 0;
243:         SelectionColor = Color.Black;
244:
245:         m_nCurSelection = nPosition;
246:     }
247:     /// <summary>
248:     /// Procesa una expresión regular
249:     /// </summary>
250:     /// <param name="strRegex">La expresión regular</param>
251:     /// <param name="color">El color</param>
252:     private void ProcessRegex(string strRegex, Color color) {
253:         Regex regKeywords = new Regex(strRegex, RegexOptions.IgnoreCase | RegexOptions.Compiled);
254:         Match regMatch;
255:
256:         for(regMatch = regKeywords.Match(m_strLine); regMatch.Success; regMatch = regMatch.NextMat
ch()) {
257:             // procesa las palabras
258:             int nStart = m_nLineStart + regMatch.Index;
259:             int nLength = regMatch.Length;
260:             SelectionStart = nStart;
261:             SelectionLength = nLength;
262:             SelectionColor = color;
263:         }
264:     }
265:     /// <summary>
266:     /// Compila los keywords como expresiones regulares
267:     /// </summary>
268:     public void CompileKeywords() {
269:         for(int i = 0; i < Settings.Keywords.Count; i++) {

```

```
270:         string strKeyword = Settings.Keywords[i];
271:
272:         if(i == Settings.Keywords.Count - 1)
273:             m_strKeywords += "\\b" + strKeyword + "\\b";
274:         else
275:             m_strKeywords += "\\b" + strKeyword + "\\b|";
276:     }
277: }
278:
279: public void ProcessAllLines() {
280:     if(!Settings.EnableSyntaxHighlight) return;
281:
282:     m_bPaint = false;
283:
284:     int nStartPos = 0;
285:     int i = 0;
286:     int nOriginalPos = SelectionStart;
287:     while(i < Lines.Length) {
288:         m_strLine = Lines[i];
289:         m_nLineStart = nStartPos;
290:         m_nLineEnd = m_nLineStart + m_strLine.Length;
291:
292:         ProcessLine();
293:         i++;
294:
295:         nStartPos += m_strLine.Length + 1;
296:     }
297:
298:     m_bPaint = true;
299: }
300: }
301:
302: /// <summary>
303: /// Clase que almacena los objetos de sintaxis
304: /// </summary>
305: public class SyntaxList {
306:     public List<string> m_rgList = new List<string>();
307:     public Color m_color = new Color();
308: }
```

```
309:
310:     /// <summary>
311:     /// Configuración de keywords y colores
312:     /// </summary>
313:     public class SyntaxSettings {
314:         SyntaxList m_rgKeywords = new SyntaxList();
315:
316:         #region Properties
317:         /// <summary>
318:         /// Lista que contiene todos los keywords
319:         /// </summary>
320:         public List<string> Keywords {
321:             get { return m_rgKeywords.m_rgList; }
322:         }
323:
324:         /// <summary>
325:         /// Color de los keywords
326:         /// </summary>
327:         public Color KeywordColor = Color.Blue;
328:
329:         /// <summary>
330:         /// String que contiene el identificador de comentarios
331:         /// </summary>
332:         public string Comment = "";
333:
334:         /// <summary>
335:         /// Color de comentarios
336:         /// </summary>
337:         public Color CommentColor = Color.Green;
338:
339:         /// <summary>
340:         /// Activa procesado de comentarios
341:         /// </summary>
342:         public bool EnableComments = true;
343:
344:         /// <summary>
345:         /// Activa procesado de números enteros
346:         /// </summary>
347:         public bool EnableIntegers = true;
```

```
348:
349:     /// <summary>
350:     /// Activa procesado de strings
351:     /// </summary>
352:     public bool EnableStrings = true;
353:
354:     /// <summary>
355:     /// Activa procesado de directivas de preprocesador
356:     /// </summary>
357:     public bool EnablePPD = true;
358:
359:     /// <summary>
360:     /// Color de strings
361:     /// </summary>
362:     public Color StringColor = Color.DarkRed;
363:
364:     /// <summary>
365:     /// Color de números enteros
366:     /// </summary>
367:     public Color IntegerColor = Color.DarkGreen;
368:
369:     /// <summary>
370:     /// Color de directivas de preprocesador
371:     /// </summary>
372:     public Color PPDColor = Color.Green;
373:
374:     public bool EnableSyntaxHighlight {
375:         get {
376:             return Settings.EnableSyntaxHighlight;
377:         }
378:     }
379: #endregion
380: }
381: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace CuruxaIDE {
7:     /// <summary>
8:     /// Programa para compilar o ensamblar código
9:     /// </summary>
10:    public abstract class Toolsuite:SoftApp {
11:        protected Toolsuite(string RealName, string Command)
12:            : base(RealName, Command) {
13:        }
14:
15:        /// <summary>
16:        /// Lista de directorios de inclusión
17:        /// </summary>
18:        /// <remarks>
19:        /// Los directorios arriba del todo son los que primero se comprueban
20:        /// </remarks>
21:        public Queue<string> IncludePaths = new Queue<string>();
22:
23:        /// <summary>
24:        /// Construye (compila o ensambla) un proyecto
25:        /// </summary>
26:        /// <param name="prj">Proyecto a construir</param>
27:        public abstract int Build(Project prj);
28:
29:        public static SDCC SDCC = new SDCC();
30:    }
31: }
```

```
1: File=File
2: Exit=Exit
3: NewProject=New Project
4: NewPrjFromEx=New Project From Example
5: Save=Save
6: Project=Project
7: ProjectX=Project "{0}"
8: NewFile=New File
9: AddExistingFile=Add Existing File
10: Build=Build
11: Program=Write Program to Microcontroller
12: Run=Run
13: Stop=Stop
14: Settings=Settings
15: Help=Help
16: Website=Curuxa Website
17: About=About Curuxa IDE
18: NewPrjName=My project
19: SavePrjAs=Save Project As...
20: AppInitialized=Application initialized
21: ProjectSettings=Project Settings
22: MainBoard:=Main Board:
23: Voltage:=Voltage:
24: Language:=Language:
25: OK=OK
26: NewMainFileName=Main
27: CrxPrjFilter=Curuxa Project (*.crx)|*.crx
28: CfileFilter=C source file (*.c)|*.c
29: PicAsmFilter=Assembly language for PICs (*.asm)|*.asm
30: SaveMainFileAs=Save Main File As...
31: PrjFileNotExist=Unable to open project. File does not exist: {0}
32: NewEmptyPrj=New Project created and saved in {0}
33: FileNotFound=File not found: {0}
34: Close=Close
35: CuruxaDescription=Curuxa is an open source electronics and robotics modular educative prototyping platform. It consists of a collection of electronic boards and modules that can be connected to each other to build complex applications.
\n\nCuruxa IDE is a simple development environment designed for using Curuxa Main Boards easily.
36: Version:=Version: {0}
37: BuildingPrj=Building project "{0}"...
```

```
38: SaveFile=Save File
39: ClosePrj=Close Project
40: PrjSettings=Project Settings
41: NoSrcLoaded=(no source code file loaded)
42: OpenPrj=Open Project...
43: OpeningPrj=Opening project file: {0}
44: WantSaveFilePrj=Do you want to save file "{0}" in project "{1}"?
45: FileNotSaved=File not saved
46: WantSavePrj=Do you want to save project "{0}"?
47: PrjNotSaved=Project not saved
48: NoMainSrc=Main source file not defined
49: BuildOk=Build ended successfully
50: BuildFail=Build failed
51: NoApp=Unable to run {0}. Are you sure it is installed?
52: FileName=File Name
53: ExFileNotExist=Example does not exist
54: ListAvailEx=Available Examples:
55: Description:=Description:
56: UsedModules:=Modules used:
57: SavePrjAs:=Save Project as:
58: ExampleDescription=Name: {0}\n\nDescription: {1}\n\nLanguage: {2}\n\nMain Board: {3}\n\nUsed Modules: {4}
59: none=none
60: UnableSaveSettings=Unable to save settings
61: NoExample=(no Examples selected)
62: NoExamples=(no Examples available)
63: GeneratingExample=Creating new project from Example "{0}"
64: Filter:=Filter:
65: MainBoards:=Main Boards:
66: Modules:=Modules:
67: ERROR=ERROR
68: NoPrjAvail=(no projects available)
69: Name:=Name:
70: UnableSaveFile=Unable to save file: {0}
71: SavingFile=Saving file: {0}
72: BurningPrj=Writing to the microcontroller the program from project "{0}"...
73: NoHex=No "hex" file found. Are you sure your project has been compiled properly?
74: RunningPrj=Running project "{0}"...
75: StoppingPrj=Stopping project "{0}"...
76: WrongMB=No programmer, no Main Board or incorrect one plugged to the programmer. Expected: {0}
```

77: UnableLoadSettings=Unable to load settings
78: LibsNotFound=Curuxa libraries not found. Are you sure Curuxa has been installed properly?
79: IdeLog=Curuxa IDE log
80: BuildLog=Build log
81: ProgLog=Programmer log
82: PrjTreeTooltipMB=Using Main Board: {0} ({1})
83: PrjTreeTooltipLang=Project programming language: {0}
84: LineX=Line {0}
85: ColX=Column {0}
86: ClosingTab=Closing file {1} from project {0}
87: NoActivePrj=No project selected
88: MenuEdit=&Edit
89: MenuUndo=&Undo
90: MenuRedo=&Redo
91: MenuSelectAll=Select &All
92: MenuCut=Cu&t
93: MenuCopy=&Copy
94: MenuPaste=&Paste
95: CloseFile=Close File
96: MenuPrint=Print
97: MenuSaveAll=Save All
98: MenuLanguage=Language
99: BuildBurnRun=Compile + Program + Run
100: PrjProgrammed=Microcontroller programmed
101: PrjRun=Running
102: PrjStopped=Stopped
103: LibNotFound=Library not found: {0}
104: SaveExample=Save
105: NoMcuDetected=No microcontroller detected
106: DetectedX=Detected: {0} (used in {1})
107: CheckMB=Check Main Board
108: UnableLoadPrjFile=Unable to load project file "{0}": {1}
109: UnableGetLatestVersion=Unable to retrieve latest version: {0}
110: ThereIsNewVersion=There is a new version available: Curuxa {0} (current: {1})
111: EnableSyntaxHighlight=Enable syntax highlight
112: HumanLanguage:=Language:
113: Preferences=Preferences
114: Ok=Ok
115: Cancel=Cancel


```
116: Restart=Restart
117: Error=Error
118: NoSrcSelected=No source file selected
119: PrintPreview=Print Preview
120: TitleReadOnly= (read-only)
121: CantSaveIsReadOnly=Unable to save read-only file: {0}
```

```
1: File=Archivo
2: Exit=Salir
3: NewProject=Nuevo Proyecto
4: NewPrjFromEx=Nuevo proyecto desde ejemplo
5: Save=Guardar
6: Project=Proyecto
7: ProjectX=Proyecto "{0}"
8: NewFile=Nuevo Archivo
9: AddExistingFile=Añadir archivo existente
10: Build=Compilar
11: Program=Grabar Programa en Microcontrolador
12: Run=Ejecutar
13: Stop=Detener
14: Settings=Preferencias
15: Help=Ayuda
16: Website=Sitio Web de Curuxa
17: About=Acerca de Curuxa IDE
18: NewPrjName=Mi proyecto
19: SavePrjAs=Guardar proyecto como...
20: AppInitialized=Aplicación iniciada
21: ProjectSettings=Preferencias del Proyecto
22: MainBoard:=Placa Principal:
23: Voltage:=Tensión:
24: Language:=Lenguaje:
25: OK=Aceptar
26: NewMainFileName=Principal
27: CrxPrjFilter=Proyecto de Curuxa (*.crx)|*.crx
28: CfileFilter=Archivo de código C (*.c)|*.c
29: PicAsmFilter=Lenguaje ensamblador para PICs (*.asm)|*.asm
30: SaveMainFileAs=Guardar archivo principal como...
31: PrjFileNotExist=No se puede abrir el proyecto. El archivo no existe: {0}
32: NewEmptyPrj=Nuevo proyecto creado y guardado en {0}
33: FileNotFound=Archivo no encontrado: {0}
34: Close=Cerrar
35: CuruxaDescription=Curuxa es una plataforma de hardware libre modular, educativa y de prototipado. Consiste en una
colección de circuitos electrónicos y módulos que pueden ser interconectados para construir aplicaciones más complejas.\n
\nCuruxa IDE es un entorno de desarrollo simple diseñado para trabajar fácilmente con las Placas Principales de Curuxa.
36: Version:=Versión: {0}
37: BuildingPrj=Compilando proyecto "{0}"...
```

```
38: SaveFile=Guardar archivo
39: ClosePrj=Cerrar proyecto
40: PrjSettings=Configurar Proyecto
41: NoSrcLoaded=(no hay archivo de código fuente abierto)
42: OpenPrj=Abrir Proyecto...
43: OpeningPrj=Abriendo archivo de proyecto: {0}
44: WantSaveFilePrj=¿Quieres guardar el archivo "{0}" en el proyecto "{1}"?
45: FileNotSaved=Archivo no guardado
46: WantSavePrj=¿Quieres guardar el proyecto "{0}"?
47: PrjNotSaved=Proyecto no guardado
48: NoMainSrc=Archivo de código principal no definido
49: BuildOk=Compilación finalizada correctamente
50: BuildFail=Compilación fallida
51: NoApp=Imposible ejecutar {0}. ¿Estás seguro de que está instalado?
52: FileName=Nombre del Archivo
53: ExFileNotExist=El ejemplo no existe
54: ListAvailEx=Ejemplos Disponibles:
55: Description:=Descripción:
56: UsedModules:=Módulos usados:
57: SavePrjAs:=Guardar Proyecto como:
58: ExampleDescription=Nombre: {0}\n\nDescripción: {1}\n\nLenguaje: {2}\n\nPlaca Principal: {3}\n\nMódulos usados: {4}
59: none=ninguno
60: UnableSaveSettings=Imposible guardar configuración
61: NoExample=(ningún ejemplo seleccionado)
62: NoExamples=(ningún ejemplo disponible)
63: GeneratingExample=Creando nuevo proyecto a partir del ejemplo "{0}"
64: Filter:=Filtrar:
65: MainBoards:=Placas Principales:
66: Modules:=Módulos:
67: ERROR=ERROR
68: NoPrjAvail=(no hay proyectos disponibles)
69: Name:=Nombre:
70: UnableSaveFile=Imposible guardar archivo: {0}
71: SavingFile=Guardando archivo: {0}
72: BurningPrj=Grabando en el microcontrolador el programa del proyecto "{0}"...
73: NoHex=Archivo "hex" no encontrado. ¿Estás seguro de que se ha compilado correctamente?
74: RunningPrj=Ejecutando proyecto "{0}"...
75: StoppingPrj=Deteniendo proyecto "{0}"...
76: WrongMB=No hay programador, no hay Placa Principal ó Placa Principal incorrecta. Se esperaba: {0}
```

```
77: UnableLoadSettings=Imposible cargar configuración
78: LibsNotFound=Librerías de Curuxa no encontradas. ¿Estás seguro de que Curuxa se ha instalado correctamente?
79: IdeLog=Log de Curuxa IDE
80: BuildLog=Log de compilación
81: ProgLog=Log del programador
82: PrjTreeTooltipMB=Usando Placa Principal: {0} ({1})
83: PrjTreeTooltipLang=Lenguaje de programación del proyecto: {0}
84: LineX=Línea {0}
85: ColX=Columna {0}
86: ClosingTab=Cerrando archivo {1} del proyecto {0}
87: NoActivePrj=Ningún proyecto seleccionado
88: MenuEdit=&Editar
89: MenuUndo=&Deshacer
90: MenuRedo=&Rehacer
91: MenuSelectAll=Seleccionar &Todo
92: MenuCut=Cor&tar
93: MenuCopy=&Copiar
94: MenuPaste=&Pegar
95: CloseFile=Cerrar Archivo
96: MenuPrint=Imprimir
97: MenuSaveAll=Guardar Todo
98: MenuLanguage=Idioma
99: BuildBurnRun=Compilar + Programar + Ejecutar
100: PrjProgrammed=Microcontrollador programado
101: PrjRun=Ejecutándose
102: PrjStopped=Detenido
103: LibNotFound=Librería no encontrada: {0}
104: SaveExample=Guardar
105: NoMcuDetected=No se detectó ningún microcontrolador
106: DetectedX=Detectado: {0} (usado en {1})
107: CheckMB=Comprobar Placa Principal
108: UnableLoadPrjFile=Imposible cargar archivo de proyecto "{0}": {1}
109: UnableGetLatestVersion=Imposible detectar la última versión: {0}
110: ThereIsNewVersion=Existe una nueva versión disponible: Curuxa {0} (actual: {1})
111: EnableSyntaxHighlight=Activar resalto de sintaxis
112: HumanLanguage:=Idioma:
113: Preferences=Preferencias
114: Ok=Aceptar
115: Cancel=Cancelar
```

```
116: Restart=Reiniciar
117: Error=Error
118: NoSrcSelected=No hay archivo seleccionado
119: PrintPreview=Vista Preliminar
120: TitleReadOnly= (sólo lectura)
121: CantSaveIsReadOnly=No se puede guardar el archivo de sólo lectura: {0}
122:
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.IO;
6: using System.Globalization;
7:
8: namespace CuruxaIDE {
9:     /// <summary>
10:    /// Contiene todas las variables y métodos necesarios para mostrar textos internacionalizados (traducidos) ✓
11:    . Antes de usar cualquier debe configurarse CurrentLanguage
12:    /// </summary>
13:    /// <remarks>
14:    /// Sólo un idioma estará cargado en cualquier momento. Todos los textos se cargan automáticamente por est ✓
15:    a clase. Todos los textos se almacenan en archivos de texto plano llamados "CurrentLanguage.lang".
16:    /// </remarks>
17:    public class i18n {
18:        /// <summary>
19:        /// Gets or sets the current language
20:        /// </summary>
21:        public static string CurrentLanguage {
22:            get {
23:                if(_CurrentLanguage == null) _CurrentLanguage = DefaultLang;
24:                return _CurrentLanguage;
25:            }
26:            set {
27:                Globals.Debug("Switching to language: {0}", value);
28:                if(AvailableLanguages.Contains(value)) {
29:                    _CurrentLanguage = value;
30:                    LangStrings.Clear();
31:                } else throw new Exception("Unknown language: " + value);
32:            }
33:        }
34:        private static string _CurrentLanguage;
35:
36:        /// <summary>
37:        /// Obtiene el idioma por defecto del sistema
38:        /// </summary>
39:        public static string DefaultLang {
```

```

38:         get {
39:             string MyLang = System.Globalization.CultureInfo.CurrentCulture.TwoLetterISOLanguageName;
40:             if(AvailableLanguages.Contains(MyLang)) return MyLang;
41:             else return "en";
42:         }
43:     }
44:
45:     public static List<string> AvailableLanguages {
46:         get {
47:             if(_AvailableLanguages == null) {
48:                 Globals.Debug("Retrieving the list of available languages");
49:                 _AvailableLanguages = new List<string>();
50:                 _AvailableLanguages.Clear();
51:                 DirectoryInfo dir = new DirectoryInfo(Settings.LanguageFilesDir);
52:                 FileInfo[] files = dir.GetFiles("*.lang");
53:                 if(files.Length == 0) throw new Exception(string.Format("No language files found in {0}", Settings.LanguageFilesDir));
54:                 foreach(FileInfo fi in files) {
55:                     _AvailableLanguages.Add(fi.Name.Replace(".lang", ""));
56:                 }
57:                 Globals.Debug("Available languages: {0}", _AvailableLanguages.CommaSeparatedList());
58:             }
59:             return _AvailableLanguages;
60:         }
61:     }
62:     protected static List<string> _AvailableLanguages;
63:
64:     /// <summary>
65:     /// Lista de todos los textos no traducidos
66:     /// </summary>
67:     public static List<string> LangStrNotTranslated = new List<string>();
68:
69:     /// <summary>
70:     /// Dado el nombre nativo de un idioma ("English", "Español"... ) obtiene el identificador de 2 letras ISO ("en", "es"...)
71:     /// </summary>
72:     /// <param name="NativeName">Nombre nativo del idioma ("English", "Español"...)</param>

```

```

73:         /// <returns>Identificador de 2 letras ISO ("en", "es"...)</returns>
74:         public static string GetLangFromNativeName(string NativeName) {
75:             string TwoLetterName = "";
76:             foreach(CultureInfo culture in CultureInfo.GetCultures(CultureTypes.AllCultures)) {
77:                 if(culture.NativeName == NativeName) TwoLetterName = culture.TwoLetterISOLanguageName;
78:             }
79:             return TwoLetterName;
80:         }
81:
82:         public static string GetNativeName(string LangID) {
83:             foreach(CultureInfo culture in CultureInfo.GetCultures(CultureTypes.AllCultures)) {
84:                 if(culture.TwoLetterISOLanguageName == LangID) return culture.NativeName;
85:             }
86:             throw new ArgumentException("Unknown LangID");
87:         }
88:
89:         protected static Dictionary<string, string> LangStrings = new Dictionary<string, string>();
90:
91:         /// <summary>
92:         /// Obtiene un texto escrito en el lenguaje configurado actualmente. Si aún no ha sido traducido se
93:         /// </summary>
94:         /// <param name="ID">Identificador</param>
95:         /// <param name="replacements">Textos que reemplazan {0}, {1}, {2}... en el texto original</param>
96:         public static string str(UInt32 ID, params object[] replacements) {
97:             return str(ID.ToString(), replacements);
98:         }
99:
100:        /// <summary>
101:        /// Obtiene un texto escrito en el lenguaje configurado actualmente. Si aún no ha sido traducido se
102:        /// </summary>
103:        /// <param name="ID">Identificador</param>
104:        /// <param name="replacements">Textos que reemplazan {0}, {1}, {2}... en el texto original</param>
105:        public static string str(string ID, params object[] replacements) {
106:            if(LangStrings.Keys.Count == 0) LoadLangStrings();
107:            string str = "";

```



```

108:         if(LangStrings.ContainsKey(ID)) {
109:             try {
110:                 str = string.Format(LangStrings[ID], replacements);
111:             } catch (FormatException fe) {
112:                 throw new FormatException("Passed " + replacements.Length + " replacements
. String [" + ID.ToString() + "]= " + LangStrings[ID], fe);
113:             }
114:         } else {
115:             string msg = "Unknown i18n ID: \"" + ID + "\". Known IDs: " + LangStrings.Keys.Comma
maSeparatedList();
116:             Console.Error.WriteLine(msg);
117:             throw new Exception(msg);
118:         }
119:         if(string.IsNullOrEmpty(str)) throw new Exception(string.Format("Language string \"{0}\" is
s empty", ID));
120:         return str;
121:     }
122:
123:     /// <summary>
124:     /// Obtiene el texto en bruto (sin formatear, sólo el texto como está escrito en el archivo de idioma)
125:     /// </summary>
126:     public static string StrBulk(string ID) {
127:         Globals.Debug("Retrieving bulk language string, ID={0}", ID);
128:         if(LangStrings.Keys.Count == 0) LoadLangStrings();
129:         string str = "";
130:         if(LangStrings.ContainsKey(ID)) str = LangStrings[ID];
131:         else throw new Exception("Unknown i18n ID: " + ID);
132:         return str;
133:     }
134:
135:     /// <summary>
136:     /// Carga los textos de cierto idioma desde un archivo, según CurrentLanguage
137:     /// </summary>
138:     /// <remarks>
139:     /// Normalmente no se necesita llamar explícitamente, los textos se cargan automáticamente cuando
el primer texto traducido es utilizado
140:     /// </remarks>
141:     public static void LoadLangStrings() {

```

```
142:         Globals.Debug("Loading language strings");
143:
144:         LangStrings.Clear();
145:         string ID = "", text = "", file = Settings.LanguageFilesDir + "/" + "en.lang";
146:
147:         //primero carga textos en inglés
148:         TextReader tr = new StreamReader(file);
149:         while(true) {
150:             string NewLine = tr.ReadLine();
151:             if(NewLine != null) {
152:                 if(NewLine != "") {
153:                     ParseLine(NewLine, out ID, out text);
154:                     if(LangStrings.ContainsKey(ID)) throw new Exception("Duplicated ID
: " + ID);
155:                     LangStrings.Add(ID, text);
156:                     LangStrNotTranslated.Add(ID);
157:                 } else Globals.Debug("WARNING: empty line found in {0}", file);
158:                 } else break;
159:             }
160:             tr.Close();
161:
162:             file = Settings.LanguageFilesDir + "/" + CurrentLanguage + ".lang";
163:
164:             //los reemplaza por el texto traducido (si existe traducción)
165:             tr = new StreamReader(file);
166:             while(true) {
167:                 string NewLine = tr.ReadLine();
168:                 if(NewLine != null) {
169:                     ParseLine(NewLine, out ID, out text);
170:                     if(!LangStrings.ContainsKey(ID)) throw new Exception(string.Format("The la
nguage {0} has an incorrect string ID: {1} (it doesn't exist in the English language file)", CurrentLanguage, ID));
171:                     LangStrings[ID] = text;
172:                     LangStrNotTranslated.Remove(ID);
173:                 } else break;
174:             }
175:             tr.Close();
176:
177:             Globals.Debug("Loaded {0} language strings", LangStrings.Count);
178:         }
```

```
179:
180:     /// <summary>
181:     /// Parsea una línea de un archivo de traducción
182:     /// </summary>
183:     /// <param name="LineText">El texto completo de la línea traducida</param>
184:     /// <param name="ID">Devuelve el identificador de la línea</param>
185:     /// <param name="text">Devuelve el texto asociado a ese identificador</param>
186:     protected static void ParseLine(string LineText, out string ID, out string text) {
187:         ID = LineText.Remove(LineText.IndexOf('='));
188:         text = LineText.Substring(ID.Length + 1);
189:     }
190: }
191: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.Diagnostics;
6: using System.IO;
7:
8: namespace CuruxaIDE {
9:     /// <summary>
10:    /// Representa el programador PICKit2
11:    /// </summary>
12:    public class pk2cmd:ProgrammerApp {
13:        public pk2cmd()
14:            : base("pk2cmd", "pk2cmd") {
15:
16:            SetupEnvironment();
17:        }
18:
19:        private void SetupEnvironment() {
20:            if(Environment.OSVersion.Platform == PlatformID.Win32NT || Environment.OSVersion.Platform
21:            == PlatformID.Win32Windows) {
22:                //añadir pk2cmd-win al PATH
23:                Environment.SetEnvironmentVariable("PATH", Environment.GetEnvironmentVariable("PATH"
24:                H") + ";" + Settings.Pk2cmdWinDir, EnvironmentVariableTarget.Process);
25:                string s = Environment.GetEnvironmentVariable("PATH");
26:            }
27:
28:            public override int Write(Project prj) {
29:                ProcessStartInfo ProcInfo;
30:                Process proc;
31:
32:                string HexFile = prj.Path + "/temp.hex"; //debe ser la ruta completa!!!
33:
34:                if(!File.Exists(HexFile)) {
35:                    Globals.LogIDE(i18n.str("NoHex"));
36:                    Globals.LogProgrammer(i18n.str("NoHex"));
37:                    return 1;
38:                }
39:            }
40:        }
41:    }
42: }
```

```
38:         }
39:
40:         //pk2cmd -PPIC16Fxxx -M -F/home/my_user/my_program.hex
41:         string args = "-P" + prj.MainBoard.GetMCU() + " -M -F" + HexFile;
42:
43:         try {
44:             if(!CheckMB(prj)) Globals.LogIDE(i18n.str("WrongMB", prj.MainBoard));
45:
46:             ProcInfo = new ProcessStartInfo(Command, args);
47:             ProcInfo.CreateNoWindow = true;
48:             ProcInfo.ErrorDialog = false;
49:             ProcInfo.RedirectStandardError = true;
50:             ProcInfo.RedirectStandardOutput = true;
51:             ProcInfo.UseShellExecute = false;
52:             proc = new Process();
53:             proc.StartInfo = ProcInfo;
54:             proc.OutputDataReceived += new DataReceivedEventHandler(proc_OutputDataReceived);
55:             proc.ErrorDataReceived += new DataReceivedEventHandler(proc_ErrorDataReceived);
56:             Globals.Debug("Executing: {0} {1}", Command, args);
57:             proc.Start();
58:             Globals.Debug("Reading outputs");
59:             proc.BeginErrorReadLine();
60:             proc.BeginOutputReadLine();
61:             Globals.Debug("Waiting for exit...");
62:             proc.WaitForExit();
63:
64:             return proc.ExitCode;
65:         } catch (System.ComponentModel.Win32Exception) {
66:             string msg = i18n.str("NoApp", RealName);
67:             Globals.LogIDE(msg);
68:             Globals.LogProgrammer(msg);
69:             return 1;
70:         }
71:     }
72:
73:     void proc_ErrorDataReceived(object sender, DataReceivedEventArgs e) {
74:         if(string.IsNullOrEmpty(e.Data)) return;
75:         for(int i = 0; i <= e.Data.Length; i++) {
76:             if(i == e.Data.Length) return;
```

```
77:                 if(e.Data[i] != ' ') break;
78:             }
79:             Globals.LogProgrammer "[" + RealName + " Error] " + e.Data);
80:         }
81:
82:         void proc_OutputDataReceived(object sender, DataReceivedEventArgs e) {
83:             //remove empty messages
84:             if(string.IsNullOrEmpty(e.Data)) return;
85:             for(int i = 0; i <= e.Data.Length; i++) {
86:                 if(i == e.Data.Length) return;
87:                 if(e.Data[i] != ' ') break;
88:             }
89:
90:             Globals.LogProgrammer "[" + RealName + "]" + e.Data);
91:         }
92:
93:         public override int Run(Project prj) {
94:             ProcessStartInfo ProcInfo;
95:             Process proc;
96:
97:             //pk2cmd -PPIC16Fxxx -Ax.x -T
98:             string args = "-P" + prj.MainBoard.GetMCU() + " -A" + prj.Voltage + " -T";
99:
100:            try {
101:                ProcInfo = new ProcessStartInfo(Command, args);
102:                ProcInfo.CreateNoWindow = true;
103:                ProcInfo.ErrorDialog = false;
104:                ProcInfo.RedirectStandardError = true;
105:                ProcInfo.RedirectStandardOutput = true;
106:                ProcInfo.UseShellExecute = false;
107:                proc = new Process();
108:                proc.StartInfo = ProcInfo;
109:                proc.OutputDataReceived += new DataReceivedEventHandler(proc_OutputDataReceived);
110:                proc.ErrorDataReceived += new DataReceivedEventHandler(proc_ErrorDataReceived);
111:                Globals.Debug("Executing: {0} {1}", Command, args);
112:                proc.Start();
113:                Globals.Debug("Reading outputs");
114:                proc.BeginErrorReadLine();
115:                proc.BeginOutputReadLine();
```

```
116:                Globals.Debug("Waiting for exit...");
117:                proc.WaitForExit();
118:
119:                return proc.ExitCode;
120:            } catch(System.ComponentModel.Win32Exception) {
121:                Globals.LogIDE(i18n.str("NoApp", RealName));
122:                return 1;
123:            }
124:        }
125:
126:        public override int Stop(Project prj) {
127:            ProcessStartInfo ProcInfo;
128:            Process proc;
129:
130:            //pk2cmd -PPIC16Fxxx -R
131:            string args = "-P" + prj.MainBoard.GetMCU() + " -R";
132:
133:            try {
134:                ProcInfo = new ProcessStartInfo(Command, args);
135:                ProcInfo.CreateNoWindow = true;
136:                ProcInfo.ErrorDialog = false;
137:                ProcInfo.RedirectStandardError = true;
138:                ProcInfo.RedirectStandardOutput = true;
139:                ProcInfo.UseShellExecute = false;
140:                proc = new Process();
141:                proc.StartInfo = ProcInfo;
142:                proc.OutputDataReceived += new DataReceivedEventHandler(proc_OutputDataReceived);
143:                proc.ErrorDataReceived += new DataReceivedEventHandler(proc_ErrorDataReceived);
144:                Globals.Debug("Executing: {0} {1}", Command, args);
145:                proc.Start();
146:                Globals.Debug("Reading outputs");
147:                proc.BeginErrorReadLine();
148:                proc.BeginOutputReadLine();
149:                Globals.Debug("Waiting for exit...");
150:                proc.WaitForExit();
151:
152:                return proc.ExitCode;
153:            } catch(System.ComponentModel.Win32Exception e) {
154:                Globals.LogIDE(i18n.str("NoApp", RealName));
```

```
155:                 return 1;
156:             }
157:         }
158:
159:         private Microcontroller? DetectedMCU;
160:
161:         public override Microcontroller? Detect() {
162:             ProcessStartInfo ProcInfo;
163:             Process proc;
164:             DetectedMCU = null;
165:
166:             //pk2cmd -P
167:             string args = "-P";
168:
169:             try {
170:                 ProcInfo = new ProcessStartInfo(Command, args);
171:                 ProcInfo.CreateNoWindow = true;
172:                 ProcInfo.ErrorDialog = false;
173:                 ProcInfo.RedirectStandardError = true;
174:                 ProcInfo.RedirectStandardOutput = true;
175:                 ProcInfo.UseShellExecute = false;
176:                 proc = new Process();
177:                 proc.StartInfo = ProcInfo;
178:                 proc.OutputDataReceived += new DataReceivedEventHandler(proc_DataReceivedCheckMCU) ✓
;
179:                 proc.ErrorDataReceived += new DataReceivedEventHandler(proc_ErrorDataReceived);
180:                 Globals.Debug("Executing: {0} {1}", Command, args);
181:                 proc.Start();
182:                 Globals.Debug("Reading outputs");
183:                 proc.BeginErrorReadLine();
184:                 proc.BeginOutputReadLine();
185:                 Globals.Debug("Waiting for exit...");
186:                 proc.WaitForExit();
187:             } catch (System.ComponentModel.Win32Exception) {
188:                 Globals.LogIDE(i18n.str("NoApp", RealName));
189:             }
190:             return DetectedMCU;
191:         }
192:
```



```
193:         void proc_DataReceivedCheckMCU(object sender, DataReceivedEventArgs e) {
194:             string StrToDetect = "Auto-Detect: Found part ";
195:             if(e != null && e.Data != null && e.Data.Contains(StrToDetect)) {
196:                 string MCU = e.Data.Remove(0, e.Data.IndexOf(StrToDetect) + StrToDetect.Length).Trim();
197:                 try {
198:                     DetectedMCU = (Microcontroller)Enum.Parse(typeof(Microcontroller), MCU);
199:                 } catch {
200:                     DetectedMCU = null;
201:                 }
202:             }
203:         }
204:
205:         /// <summary>
206:         /// Comprueba si hay alguna Placa Principal conectada al programador y si es la correcta
207:         /// </summary>
208:         public bool CheckMB(Project prj) {
209:             Microcontroller? MCU = Detect();
210:             if(MCU.HasValue && prj.MainBoard.GetMCU() == MCU.Value) return true;
211:             else return false;
212:         }
213:     }
214: }
```

Anexo XXI

Tipo: Código fuente (librerías para microcontroladores)

Librerías que los usuarios pueden utilizar para desarrollar sus programas para microcontroladores

Lenguaje: C

Archivos de código: 8

```

1:  /*=====
2:  * Funciones de retrasos para PICs
3:  *
4:  * Aviso 1:
5:  * Estos retrasos mantienen al PIC sin hacer nada, desperdiciando
6:  * tiempo y energía. Utiliza temporizadores e interrupciones si
7:  * quieres ejecutar algo mientras esperas.
8:  *
9:  * Aviso 2:
10:  * Estas funciones son muy inexactas. Para una mejor temporización
11:  * usa temporizadores internos. Para temporizaciones de precisión
12:  * utiliza Relojes en Tiempo Real externos.
13:  *
14:  * Antes de utilizar esta librería debes definir uno de los siguientes valores:
15:  * OSC_8MHz
16:  *
17:  * Por ejemplo:
18:  * #define OSC_8MHz
19:  *
20:  * Adrián Bulnes Parra
21:  *=====*/
22:
23:  // Retrasa el programa 10 microsegundos
24:  void Delay10us(){
25:      #ifdef OSC_8MHz
26:          _asm
27:              GOTO $+1
28:              GOTO $+1
29:              GOTO $+1
30:              GOTO $+1
31:              GOTO $+1
32:              GOTO $+1
33:              GOTO $+1
34:              GOTO $+1
35:              GOTO $+1
36:          _endasm;
37:      #else
38:          #error Frecuencia del oscilador no definida. Pon '#define OSC_8MHz' (o tu frec. de osc.) antes de '#include <Delays.h>'
39:      #endif
40:  }
41:
42:  // Retrasa el programa durante una cierta cantidad de milisegundos
43:  void Delay(unsigned long ms){
44:      #ifdef OSC_8MHz
45:          unsigned long i;
46:          while(ms--){
47:              for (i=0; i<118; i++) {
48:                  //no hacer nada
49:              }
50:          }
51:      #else
52:          #error Frecuencia del oscilador no definida. Pon '#define OSC_8MHz' (o tu frec. de osc.) antes de '#include <Delays.h>'
53:      #endif
54:  }
55:
56:  // Retrasa el programa una cierta cantidad de segundos
57:  void DelaySec(unsigned long seconds){
58:      #ifdef OSC_8MHz

```

```
59:         while(seconds--){
60:             Delay(950);
61:         }
62:         #else
63:             #error Frecuencia del oscilador no definida. Pon '#define
OSC_8MHz' (o tu frec. de osc.) antes de '#include <Delays.h>'
64:         #endif
65:     }
66:
67:     // Retrasa el programa una cierta cantidad de minutos
68:     void DelayMin(unsigned long minutes){
69:         #ifdef OSC_8MHz
70:             while(minutes--){
71:                 Delay(55);
72:             }
73:         #else
74:             #error Frecuencia del oscilador no definida. Pon '#define
OSC_8MHz' (o tu frec. de osc.) antes de '#include <Delays.h>'
75:         #endif
76:     }
77:
```

```
1: /*=====
2:  * Librería para gestionar GP2D-12 / GP2Y0A21YK
3:  * Adrián Bulnes Parra
4:  *=====*/
5:
6: // valores digitales para distancias cuando Vcc=5V
7: // Aviso: los valores y distancias son inversamente proporcionales, esto
significa que un valor mayor de, por ejemplo, GP2_10cm, indica en realidad que e
l objeto está a menos de 10 cm del sensor
8: #define GP2_6cm 635
9: #define GP2_10cm 471
10: #define GP2_15cm 338
11: #define GP2_25cm 225
12: #define GP2_40cm 154
13: #define GP2_80cm 82
14:
15: // Nota para desarrolladores: las funciones para calcular distancias auto
máticamente se habían implementado pero fueron eliminadas debido a un consumo ex
cesivo de memoria. Se recomienda utilizar los "defines" y suponer relación linea
l entre valor del ADC y distancia real
16:
```

```
1:  /*=====
2:  *  Librería para el control de motores L293 (B, D, NE...)
3:  *
4:  *  Adrián Bulnes Parra
5:  *=====*/
6:
7:
8:  /* Antes de usar esta librería, los siguientes defines deben establecerse
9:  para que coincidan con los pines digitales a los cuales el L293 está conectado:
10:
11:  M1Enable, M1In1, M1In2, M2Enable, M2In1, M2In2
12:
13:
14:  Por ejemplo (pon esto en MiPrograma.c):
15:
16:  #define M1Enable RA2
17:  #define M1In1 RA3
18:  #define M1In2 RA4
19:  #define M2Enable RA1
20:  #define M2In1 RA0
21:  #define M2In2 RA7
22:  */
23:
24:  // Mover motor 1 hacia delante
25:  void M1Fwd(){
26:      M1Enable=1;
27:      M1In1=1;
28:      M1In2=0;
29:  }
30:
31:  // Mover motor 1 hacia atrás
32:  void M1Bck(){
33:      M1Enable=1;
34:      M1In1=0;
35:      M1In2=1;
36:  }
37:
38:  // Frena eléctricamente el motor 1
39:  void M1Brake(){
40:      M1Enable=1;
41:      M1In1=0;
42:      M1In2=0;
43:  }
44:
45:  // El motor 1 gira libremente
46:  void M1Free(){
47:      M1Enable=0;
48:  }
49:
50:  // Mover motor 2 hacia delante
51:  void M2Fwd(){
52:      M2Enable=1;
53:      M2In1=1;
54:      M2In2=0;
55:  }
56:
57:  // Mover motor 2 hacia atrás
58:  void M2Bck(){
59:      M2Enable=1;
60:      M2In1=0;
```

```
61:         M2In2=1;
62:     }
63:
64:     // Frena eléctricamente el motor 2
65:     void M2Brake(){
66:         M2Enable=1;
67:         M2In1=0;
68:         M2In2=0;
69:     }
70:
71:     // El motor 2 gira libremente
72:     void M2Free(){
73:         M2Enable=0;
74:     }
```

```
1: /*=====
2:  * Librería con características comunes para todos los MBP (usando PICs)
3:  *
4:  * Adrián Bulnes Parra
5:  *=====*/
6:
7: #define int8 char
8: #define int16 int
9: #define int32 long
10:
11: #define bool unsigned int8
12: #define true 1
13: #define TRUE 1
14: #define false 0
15: #define FALSE 0
16:
17: #define DigitalInput 1
18: #define DigitalOutput 0
19: #define AllDigitalInputs 0xFF
20: #define AllDigitalOutputs 0x00
21:
22: #define ConfigBits(bits) typedef unsigned int config; config at 0x2007 __✓
CONFIG = bits
23: #define ConfigBits1(bits) typedef unsigned int config; config at _CONFIG1✓
__CONFIG = bits
24:
25: #define SetBit(v,bit) v |= (1 << bit);
26: #define ClearBit(v,bit) v &= ~(1 << bit);
```



```
1:  /*=====
2:  *  Librería para MBP14 usando un PIC16F688
3:  *
4:  *  Adrián Bulnes Parra
5:  *=====*/
6:
7:  #include <pic16f688.h>
8:  #include <MBP.h>
9:
10: #define ADC_MAX_VALUE 1024
11:
12: // Desactiva el conversor analógico-digital
13: void AdcDisable(){
14:     ADON=0;
15:     ANSEL=0;
16: }
17:
18: // Configura AN0 como entrada analógica, y la usa como pin de origen para
la conversión analógico-digital
19: void EnableAN0(){
20:     TRISA0=1;
21:     SetBit(ANSEL, 0);
22:     CHS2=0;
23:     CHS1=0;
24:     CHS0=0;
25: }
26:
27: // Obtiene de manera síncrona el valor digital de la entrada analógica ac
tiva, es decir, espera sin hacer nada hasta que la medición termina
28: unsigned int16 AdcMeasure(){
29:     ADFM=1;
30:     ADON=1;
31:     GO=1;
32:     while(NOT_DONE) /* no hacer nada */;
33:     return ADRESH << 8 | ADRESL;
34: }
```

```
1:  /*=====
2:  *  Librería para MBP18 usando un PIC16F88
3:  *
4:  *  Adrián Bulnes Parra
5:  *=====*/
6:
7:  #include <pic16f88.h>
8:  #include <MBP.h>
9:
10: #define ADC_MAX_VALUE 1024
11:
12: #define Interrupt static void isr(void) interrupt 0
13: #define ISR Interrupt
14:
15: // Establece el oscilador interno en 31.25kHz
16: void SetIntosc31kHz(){
17:     IRCF2=0;
18:     IRCF1=0;
19:     IRCF0=0;
20: }
21:
22: // Establece el oscilador interno en 8MHz
23: void SetIntosc8MHz(){
24:     IRCF2=1;
25:     IRCF1=1;
26:     IRCF0=1;
27: }
28:
29: // Desactiva el conversor analógico-digital
30: void AdcDisable(){
31:     ADON=0;
32:     ANSEL=0;
33: }
34:
35: // Configura AN0 como entrada analógica, y la usa como pin de origen para
la conversión analógico-digital
36: void EnableAN0(){
37:     TRISA0=1;
38:     ANSEL&=0b00000001;
39:     CHS2=0;
40:     CHS1=0;
41:     CHS0=0;
42: }
43:
44: // Obtiene de manera síncrona el valor digital de la entrada analógica ac
tiva, es decir, espera sin hacer nada hasta que la medición termina
45: unsigned int16 AdcMeasure(){
46:     ADFM=1;
47:     ADON=1;
48:     GO=1;
49:     while(NOT_DONE) /*do nothing*/;
50:     return ADRESH << 8 | ADRESL;
51: }
52:
53: // Activa la interrupción del ADC. Una interrupción se lanzará cada vez q
ue la conversión analógico-digital termine
54: void AdcEnableInt(){
55:     ADIF=0;
56:     ADIE=1;
57: }
58:
```

```
59: // Desactiva la interrupción del ADC
60: void AdcDisableInt(){
61:     ADIE=0;
62: }
```

```
1:  /*=====
2:  *  Librería para MBP40 usando un PIC16F887
3:  *
4:  *  Adrián Bulnes Parra
5:  *=====*/
6:
7:  #include <pic16f887.h>
8:  #include <MBP.h>
9:
10: #define Interrupt static void isr(void) interrupt 0
11: #define ISR Interrupt
12:
13: #define ADC_MAX_VALUE 1024
14:
15: // Desactiva el conversor analógico-digital
16: void AdcDisable(){
17:     ADON=0;
18:     ANSEL=0;
19:     ANSELH=0;
20: }
21:
22: // Establece el oscilador interno en 8MHz
23: void SetIntosc8MHz(){
24:     IRCF2=1;
25:     IRCF1=1;
26:     IRCF0=1;
27: }
```

```
1:  /*=====
2:  *  Librería para MBP8 usando un PIC12F683
3:  *
4:  *  Adrián Bulnes Parra
5:  *=====*/
6:
7:  #include <pic12f683.h>
8:  #include <MBP.h>
9:
10: #define ADC_MAX_VALUE 1024
11:
12: #define Interrupt void Intr(void) __interrupt 0
13: #define interrupt Interrupt
14: #define ISR Interrupt
15:
16: #define SetDigitalInput(pin) SetBit(TRISIO, pin)
17: #define SetDigitalOutput(pin) ClearBit(TRISIO, pin)
18:
19: // Desactiva el conversor analógico-digital
20: void AdcDisable(){
21:     ADON=0;
22:     ANSEL=0;
23: }
24:
25: // Configura AN2 como entrada analógica, y la usa como pin de origen para la conversión analógico-digital
26: void EnableAN2(){
27:     SetBit(TRISIO, 2);
28:     ANS2=1;
29:     CHS1=1;
30:     CHS0=0;
31: }
32:
33: // Configura AN3 como entrada analógica, y la usa como pin de origen para la conversión analógico-digital
34: void EnableAN3(){
35:     SetBit(TRISIO, 4);
36:     ANS3=1;
37:     CHS1=1;
38:     CHS0=1;
39: }
40:
41: // Obtiene de manera síncrona el valor digital de la entrada analógica activa, es decir, espera sin hacer nada hasta que la medición termina
42: unsigned int16 AdcMeasure(){
43:     ADFM=1;
44:     ADON=1;
45:     GO=1;
46:     while(NOT_DONE) /* no hacer nada */;
47:     return ADRESH << 8 | ADRESL;
48: }
49:
50: // Establece el oscilador interno en 8MHz
51: void SetIntosc8MHz(){
52:     IRCF2=1;
53:     IRCF1=1;
54:     IRCF0=1;
55: }
```

Anexo XXII

Tipo: Código fuente (scripts para ordenador)

***Scripts* para la línea de comandos**

Lenguaje: Unix Shell (Bash o equivalente)

Archivos de código: 9

```
1: #!/bin/bash
2: # Graba un programa dado en el microcontrolador conectado al PICKit2
3:
4: if [ "$1" = "" ]; then
5:     echo "El primer argumento debe ser un archivo .hex"
6:     exit 1
7: fi
8:
9: CurrentPIC="'detectpic'"
10: pk2cmd -P$CurrentPIC -M -F'pwd'/$@
```

```
1: #!/bin/sh
2: # Compila un programa escrito para la Placa Principal MBP14
3:
4: sdcc -I [RUTA_DE_INSTALACION_DE_CURUXA]/lib -mpic14 -p16f688 "$@"
```



```
1: #!/bin/sh  
2: # Compila un programa escrito para la Placa Principal MBP18  
3:  
4: sdcc -I [RUTA_DE_INSTALACION_DE_CURUXA]/lib -mpic14 -p16f88 "$@"
```

```
1: #!/bin/sh
2: # Compila un programa escrito para la Placa Principal MBP40
3:
4: sdcc -I [RUTA_DE_INSTALACION_DE_CURUXA]/lib -mpic14 -p16f887 "$@"
```

```
1: #!/bin/sh
2: # Compila un programa escrito para la Placa Principal MBP8
3:
4: sdcc -I [RUTA_DE_INSTALACION_DE_CURUXA]/lib -mpic14 -p12f683 "$@"
```

1: `#!/bin/sh`

2: `# Ejecuta el entorno de desarrollo`

3:

4: `MONO_THEME="clearlooks" mono [RUTA_DE_INSTALACION_DE_CURUXA]/curuxa.exe "` ↗

`$@"`

```
1: #!/bin/sh
2: # Devuelve el modelo de microcontrolador conectado actualmente al PICKit2
3:
4: pk2cmd -P | sed 's/Auto-Detect: Found part //' | sed 's/\\./' | head -n 1
```

```
1: #!/bin/bash
2: # Alimenta la Placa Principal a 5V desde el PICKit2.
3: # El circuito y el programa del microcontrolador comenzarán a funcionar.
4:
5:
6: # El modelo de PIC no importa
7: pk2cmd -PPIC16F887 -A5 -T
```

```
1: #!/bin/bash
2: # Este programa quita la tensión de alimentación de la placa principal
3: # en el caso de que esté siendo alimentada desde el PICKit2, haciendo
4: # que el programa del microcontrolador deje de ejecutarse y el resto
5: # del circuito se desactive
6:
7: #El modelo de PIC no importa
8: pk2cmd -PPIC16F887 -R
```

Anexo XXIII

Tipo: Código fuente (programa para microcontrolador)

Programa: Electrooculógrafo

Lenguaje: C

Archivos de código: 9


```

1:  /*=====
2:  *  Electrooculógrafo
3:  *
4:  *  Conexiones:
5:  *  Módulo EOG: RA0/AN0 (pin 17)
6:  *  SISW-SPST: RB3 (pin 9)
7:  *  AO-SPK: RA1 (pin 18)
8:  *  Primer LTIND-A: RA2 (pin 1)
9:  *  Segundo LTIND-A: RA3 (pin 2)
10: *
11: *  Cada LED se iluminará y un sonido diferente será generado
12: *  según en qué dirección esté mirando el sujeto.
13: *
14: *  Mientras el botón está pulsado, ambos LEDs se apagarán, el
15: *  altavoz no sonará, y el electrooculógrafo se calibrará.
16: *
17: *  Adrián Bulnes Parra
18: *
19: *===== */
20:
21: #include <MBP18.h>
22:
23: #define OSC_8MHz
24: #include <Delays.h>
25:
26: ConfigBits1(_CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF &
& _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO);
27:
28: #define Button RB3
29: #define Pressed 0
30: #define Released 1
31:
32: #define LED_R RA2
33: #define LED_L RA3
34: #define LedON 1
35: #define LedOFF 0
36:
37: #define SquareOut RA1
38:
39: bool calibrating;
40:
41: Interrupt {
42:     if(!calibrating) SquareOut=!SquareOut;
43:     TMR0IF=0;
44: }
45:
46: //Mirando hacia la derecha
47: void LookR(){
48:     LED_R=LedON;
49:     LED_L=LedOFF;
50:
51:     //prescaler=16, 488Hz
52:     PS2=0;
53:     PS1=1;
54:     PS0=1;
55: }
56:
57: //Mirando hacia la izquierda
58: void LookL(){
59:     LED_R=LedOFF;
60:     LED_L=LedON;

```

```
61:
62:         //prescaler=32, 244Hz
63:         PS2=1;
64:         PS1=0;
65:         PS0=0;
66:     }
67:
68: void main() {
69:     // configuracion
70:     unsigned int8 i;
71:     float center=0;
72:     float val=0;
73:     calibrating=true;
74:     SetIntosc8MHz();
75:     EnableAN0();
76:     TRISA1=DigitalOutput;
77:     TRISA2=DigitalOutput;
78:     TRISA3=DigitalOutput;
79:     TRISB3=DigitalInput;
80:     PSA=0;
81:     T0CS=0;
82:     TMR0IF=0;
83:     GIE=1;
84:     TMR0IE=1;
85:     SquareOut=0;
86:
87:     // espera a la calibración
88:     while(Button==Released) {
89:         LED_R=LedON;
90:         LED_L=LedON;
91:     }
92:
93:     while(1) {
94:         // calibra el voltaje de mirada central mientras se pulsa
95:         while(Button==Pressed) {
96:             calibrating=true;
97:             center=(center*0.99+AdcMeasure()*0.01);
98:         }
99:         calibrating=false;
100:
101:         //media aritmética de 200 medidas
102:         val=0;
103:         for(i=0; i<200; i++) {
104:             val+=AdcMeasure();
105:         }
106:         val/=200;
107:
108:         if(val>center) LookR();
109:         else LookL();
110:     }
111: }
```

Anexo XXIV

Tipo: Código fuente (programa para microcontrolador)

Programa: “KETJO” – Octópodo

Lenguaje: C

Archivos de código: 2

```
1:  /*=====
2:  *  Librería para movimiento de patas del octópodo.
3:  *
4:  *  Adrián Bulnes Parra
5:  *=====*/
6:
7:  // Caminar hacia delante
8:  void MvFwd(){
9:      M1Fwd();
10:     M2Fwd();
11: }
12:
13: // Detener robot
14: void Stop(){
15:     M1Free();
16:     M2Free();
17: }
18:
19: // Caminar hacia atrás
20: void MvBck(){
21:     M1Bck();
22:     M2Bck();
23: }
24:
25: // Girar hacia la izquierda
26: void TurnL(){
27:     M1Free();
28:     M2Fwd();
29: }
30:
31: // Girar hacia la derecha
32: void TurnR(){
33:     M1Fwd();
34:     M2Free();
35: }
36:
37: // Rotar hacia la izquierda
38: void Rotatel(){
39:     M1Bck();
40:     M2Fwd();
41: }
42:
43: // Rotar hacia la derecha
44: void RotateR(){
45:     M1Fwd();
46:     M2Bck();
47: }
```

```
1:  /*=====
2:  * Archivo de código principal para el octópodo
3:  *
4:  * Configuración:
5:  * Motor izquierdo: Motor 1
6:  * Motor derecho: Motor 2
7:  * Receptor infrarrojos: RB0 (pin 6)
8:  *
9:  * Adrián Bulnes Parra
10:  *===== */
11:
12: #include <MBP18.h>
13:
14: ConfigBits1(_CP_OFF & _DEBUG_OFF & _WRT_PROTECT_OFF & _CPD_OFF & _LVP_OFF &
& _BODEN_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTRC_IO);
15:
16: #define OSC_8MHz
17: #include <Delays.h>
18:
19: // Conexiones
20: #define M1Enable RA2
21: #define M1In1 RA3
22: #define M1In2 RA4
23: #define M2Enable RA1
24: #define M2In1 RA0
25: #define M2In2 RA7
26: #define RC RB0
27:
28: #include <L293.h>
29: #include "BaseMovement.h"
30:
31: void Setup(){
32:     AdcDisable();
33:
34:     TRISA0=DigitalOutput;
35:     TRISA1=DigitalOutput;
36:     TRISA2=DigitalOutput;
37:     TRISA3=DigitalOutput;
38:     TRISA4=DigitalOutput;
39:     TRISA7=DigitalOutput;
40:     TRISB0=DigitalInput;
41:
42:     SetIntosc8MHz();
43: }
44:
45: void main(){
46:     int i=0;
47:
48:     Setup();
49:
50:     Stop();
51:
52:     while(1){
53:         if(RC==0) i++;
54:         else i=0;
55:         if(i>5){
56:             MvFwd(); DelaySec(5);
57:             RotateL(); DelaySec(3);
58:             MvFwd(); DelaySec(6);
59:             RotateR(); DelaySec(6);
60:             MvFwd(); DelaySec(6);
```

```
61:                                     RotateL(); DelaySec(3);
62:                                     MvFwd(); DelaySec(7);
63:                                     Stop();
64:                                     }
65:     }
66: }
```

Anexo XXV

Tipo: Código fuente (programa para microcontrolador)

Programa: Claqueta digital

Lenguaje: C

Archivos de código: 1

```

1:  /*=====
2:  *  Claqueta digital
3:  *
4:  *  Adrián Bulnes Parra
5:  *=====*/
6:
7:  #include <MBP40.h>
8:
9:  #define OSC_8MHz
10: #include <Delays.h>
11:
12: ConfigBits1(_CP_OFF & _DEBUG_OFF & _FCMEN_OFF & _IESO_OFF & _CPD_OFF & _
LVP_OFF & _BOR_OFF & _MCLRE_OFF & _PWRTE_ON & _WDT_OFF & _INTOSCIO);
13:
14: //displays
15: #define D1 RE2
16: #define D2 RE1
17: #define D3 RE0
18: #define D4 RA5
19: #define D5 RA4
20: #define D6 RA3
21:
22: //segmentos
23: #define SA RD1
24: #define SB RD0
25: #define SC RC3
26: #define SD RC2
27: #define SE RC1
28: #define SF RC0
29: #define SG RA6
30:
31: #define LED RD3
32:
33: //control de segmentos y LED
34: #define ON 1
35: #define OFF 0
36:
37: //BOTONES
38: //...grupo 1, 2, 3. Arriba/Abajo (Up/down):
39: #define G1u RB0
40: #define G1d RB1
41: #define G2u RB2
42: #define G2d RB3
43: #define G3u RB4
44: #define G3d RB5
45: //...botón de puesta en espera:
46: #define ButtonSB RD2
47: //...control:
48: #define Pressed 0
49: #define Released 1
50:
51: //retraso entre displays (tiempo que cada display está encendido)
52: #define CommonDelay() Delay(1)
53:
54: //retraso tras cada pulsación de botón
55: #define ButtonDelay() Delay(20);
56:
57: unsigned int8 G1[2];
58: unsigned int8 G2[2];
59: unsigned int8 G3[2];
60:

```



```
61: void AllSegmOn(){
62:     SA=ON;
63:     SB=ON;
64:     SC=ON;
65:     SD=ON;
66:     SE=ON;
67:     SF=ON;
68:     SG=ON;
69: }
70:
71: void AllSegmOff(){
72:     SA=OFF;
73:     SB=OFF;
74:     SC=OFF;
75:     SD=OFF;
76:     SE=OFF;
77:     SF=OFF;
78:     SG=OFF;
79: }
80:
81: void AllDispOn(){
82:     D1=ON;
83:     D2=ON;
84:     D3=ON;
85:     D4=ON;
86:     D5=ON;
87:     D6=ON;
88: }
89:
90: void AllDispOff(){
91:     D1=OFF;
92:     D2=OFF;
93:     D3=OFF;
94:     D4=OFF;
95:     D5=OFF;
96:     D6=OFF;
97: }
98:
99: void SetSegm(unsigned int8 n){
100:     switch(n){
101:         case 0:
102:             SA=ON;
103:             SB=ON;
104:             SC=ON;
105:             SD=ON;
106:             SE=ON;
107:             SF=ON;
108:             SG=OFF;
109:             break;
110:         case 1:
111:             SA=OFF;
112:             SB=ON;
113:             SC=ON;
114:             SD=OFF;
115:             SE=OFF;
116:             SF=OFF;
117:             SG=OFF;
118:             break;
119:         case 2:
120:             SA=ON;
121:             SB=ON;
```

```
122:                SC=OFF;
123:                SD=ON;
124:                SE=ON;
125:                SF=OFF;
126:                SG=ON;
127:                break;
128:    case 3:
129:        SA=ON;
130:        SB=ON;
131:        SC=ON;
132:        SD=ON;
133:        SE=OFF;
134:        SF=OFF;
135:        SG=ON;
136:        break;
137:    case 4:
138:        SA=OFF;
139:        SB=ON;
140:        SC=ON;
141:        SD=OFF;
142:        SE=OFF;
143:        SF=ON;
144:        SG=ON;
145:        break;
146:    case 5:
147:        SA=ON;
148:        SB=OFF;
149:        SC=ON;
150:        SD=ON;
151:        SE=OFF;
152:        SF=ON;
153:        SG=ON;
154:        break;
155:    case 6:
156:        SA=ON;
157:        SB=OFF;
158:        SC=ON;
159:        SD=ON;
160:        SE=ON;
161:        SF=ON;
162:        SG=ON;
163:        break;
164:    case 7:
165:        SA=ON;
166:        SB=ON;
167:        SC=ON;
168:        SD=OFF;
169:        SE=OFF;
170:        SF=OFF;
171:        SG=OFF;
172:        break;
173:    case 8:
174:        SA=ON;
175:        SB=ON;
176:        SC=ON;
177:        SD=ON;
178:        SE=ON;
179:        SF=ON;
180:        SG=ON;
181:        break;
182:    case 9:
```

```
183:             SA=ON;
184:             SB=ON;
185:             SC=ON;
186:             SD=OFF;
187:             SE=OFF;
188:             SF=ON;
189:             SG=ON;
190:             break;
191:         default:
192:             SA=OFF;
193:             SB=OFF;
194:             SC=OFF;
195:             SD=OFF;
196:             SE=OFF;
197:             SF=OFF;
198:             SG=OFF;
199:             break;
200:     }
201: }
202:
203: void main() {
204:     SetIntosc8MHz();
205:     AdcDisable();
206:
207:     TRISE2=DigitalOutput;
208:     TRISE1=DigitalOutput;
209:     TRISE0=DigitalOutput;
210:     TRISA5=DigitalOutput;
211:     TRISA4=DigitalOutput;
212:     TRISA3=DigitalOutput;
213:     TRISD1=DigitalOutput;
214:     TRISD0=DigitalOutput;
215:     TRISC3=DigitalOutput;
216:     TRISC2=DigitalOutput;
217:     TRISC1=DigitalOutput;
218:     TRISC0=DigitalOutput;
219:     TRISA6=DigitalOutput;
220:
221:     TRISD3=DigitalOutput;
222:
223:     TRISB0=DigitalInput;
224:     TRISB1=DigitalInput;
225:     TRISB2=DigitalInput;
226:     TRISB3=DigitalInput;
227:     TRISB4=DigitalInput;
228:     TRISB5=DigitalInput;
229:
230:     //valores por defecto
231:     G1[0]=0;
232:     G1[1]=0;
233:     G2[0]=0;
234:     G2[1]=0;
235:     G3[0]=0;
236:     G3[1]=0;
237:
238:     //comprobando LEDs...
239:     while(ButtonSB==Released) {
240:         AllSegmOn();
241:         AllDispOn();
242:         LED=ON;
243:     }
```

```
244:         ButtonDelay();
245:         while(ButtonSB==Pressed) ;
246:         ButtonDelay();
247:
248:         //bucle principal
249:         while(1){
250:             LED=ON;
251:
252:             //actualizar valores
253:             if(G1u==Pressed) {
254:                 ButtonDelay();
255:                 G1[0]++;
256:                 if(G1[0]>9) {
257:                     G1[0]=0;
258:                     G1[1]++;
259:                 }
260:                 if(G1[1]>9) {
261:                     G1[1]=0;
262:                 }
263:                 while(G1u==Pressed) ;
264:             }
265:
266:             if(G1d==Pressed) {
267:                 ButtonDelay();
268:                 if(G1[0]!=0) G1[0]--;
269:                 else {
270:                     G1[0]=9;
271:                     if(G1[1]!=0) G1[1]--;
272:                     else G1[1]=9;
273:                 }
274:                 while(G1d==Pressed);
275:             }
276:
277:             if(G2u==Pressed) {
278:                 ButtonDelay();
279:                 G2[0]++;
280:                 if(G2[0]>9) {
281:                     G2[0]=0;
282:                     G2[1]++;
283:                 }
284:                 if(G2[1]>9) {
285:                     G2[1]=0;
286:                 }
287:                 while(G2u==Pressed) ;
288:             }
289:
290:             if(G2d==Pressed) {
291:                 ButtonDelay();
292:                 if(G2[0]!=0) G2[0]--;
293:                 else {
294:                     G2[0]=9;
295:                     if(G2[1]!=0) G2[1]--;
296:                     else G2[1]=9;
297:                 }
298:                 while(G2d==Pressed);
299:             }
300:
301:             if(G3u==Pressed) {
302:                 ButtonDelay();
303:                 G3[0]++;
304:                 if(G3[0]>9) {
```

```

305:                                G3[0]=0;
306:                                G3[1]++;
307:                                }
308:                                if(G3[1]>9) {
309:                                    G3[1]=0;
310:                                }
311:                                while(G3u==Pressed) ;
312:                            }
313:
314:                            if(G3d==Pressed) {
315:                                ButtonDelay();
316:                                if(G3[0]!=0) G3[0]--;
317:                                else {
318:                                    G3[0]=9;
319:                                    if(G3[1]!=0) G3[1]--;
320:                                    else G3[1]=9;
321:                                }
322:                                while(G3d==Pressed);
323:                            }
324:
325:                            //mostrar números
326:                            SetSegm(G1[0]);
327:                            D1=ON;
328:                            CommonDelay();
329:                            D1=OFF,
330:
331:                            SetSegm(G1[1]);
332:                            D2=ON;
333:                            CommonDelay();
334:                            D2=OFF,
335:
336:                            SetSegm(G2[0]);
337:                            D3=ON;
338:                            CommonDelay();
339:                            D3=OFF;
340:
341:                            SetSegm(G2[1]);
342:                            D4=ON;
343:                            CommonDelay();
344:                            D4=OFF;
345:
346:                            SetSegm(G3[0]);
347:                            D5=ON;
348:                            CommonDelay();
349:                            D5=OFF;
350:
351:                            SetSegm(G3[1]);
352:                            D6=ON;
353:                            CommonDelay();
354:                            D6=OFF;
355:
356:                            //modo de puesta en espera (bajo consumo)
357:                            if(ButtonSB==Pressed){
358:                                ButtonDelay();
359:                                while(ButtonSB==Pressed) ; //esperar a que se suelte el botón
360:                                ButtonDelay();
361:                                AllDispOff();
362:                                while(ButtonSB==Released) {
363:                                    LED=!LED;
364:                                    Delay(100);

```

```
365:                                     }  
366:                                     while(ButtonSB==Pressed) ;  
367:                                 }  
368:         }  
369:     }  
370:  
371:
```

Anexo XXVI

Tipo: Código fuente (programa para microcontrolador)

Programa: PCBot

Lenguaje: C

Archivos de código: 4

```
1:  /*=====
2:  * Robot de dos ruedas motrices controlado desde ordenador
3:  *
4:  * Librería para controlar el movimiento de la base del robot.
5:  * Es decir, controla los motores para que la estructura completa
6:  * del robot se mueva en cierta dirección.
7:  *
8:  * Adrián Bulnes Parra
9:  *=====*/
10:
11: unsigned int8 BaseMvStatus = CccBaseMvStop;
12:
13: // Envía por el puerto serie el estado actual de movimiento
14: void SendBaseMvStatus() {
15:     WriteSP(CcStatusBaseMovement);
16:     WriteSP(BaseMvStatus);
17: }
18:
19: // Mover base hacia delante
20: void BaseMvFwd(){
21:     M1Fwd();
22:     M2Fwd();
23:     BaseMvStatus=CccBaseMvFwd;
24:     SendBaseMvStatus();
25: }
26:
27: // Detener base del robot
28: void BaseStop(){
29:     M1Free();
30:     M2Free();
31:     BaseMvStatus=CccBaseMvStop;
32:     SendBaseMvStatus();
33: }
34:
35: // Mover hacia atrás
36: void BaseMvBck(){
37:     M1Bck();
38:     M2Bck();
39:     BaseMvStatus=CccBaseMvBck;
40:     SendBaseMvStatus();
41: }
42:
43: // Girar hacia la izquierda
44: void BaseTurnL(){
45:     M1Free();
46:     M2Fwd();
47:     BaseMvStatus=CccBaseTurnL;
48:     SendBaseMvStatus();
49: }
50:
51: // Girar hacia la derecha
52: void BaseTurnR(){
53:     M1Fwd();
54:     M2Free();
55:     BaseMvStatus=CccBaseTurnR;
56:     SendBaseMvStatus();
57: }
58:
59: // Rotar hacia la izquierda
60: void BaseRotatel(){
61:     M1Bck();
```



```
62:         M2Fwd();
63:         BaseMvStatus=CccBaseRotateL;
64:         SendBaseMvStatus();
65:     }
66:
67:     // Rotar hacia la derecha
68:     void BaseRotateR(){
69:         M1Fwd();
70:         M2Bck();
71:         BaseMvStatus=CccBaseRotateR;
72:         SendBaseMvStatus();
73:     }
```

```
1:  /*=====
2:  * Robot de dos ruedas motrices controlado desde ordenador
3:  *
4:  * Librería para procesar las comunicaciones
5:  *
6:  * Adrián Bulnes Parra
7:  *=====*/
8:
9:  // variable temporada para almacenar los datos recibidos
10: unsigned int8 Rcv;
11:
12: void Authenticate(){
13:     WriteSP(CcAuthID);
14:     WriteSP(CccAuthID);
15: }
16:
17: // Se recibió una instrucción para establecer el estado de una luz
18: void SetLight(){
19:     Rcv=ReadSPWait();
20:     switch(Rcv) {
21:         case CccLightFrontLeftOn:
22:             LightFrontLeft=ON;
23:             break;
24:         case CccLightFrontLeftOff:
25:             LightFrontLeft=OFF;
26:             break;
27:         case CccLightFrontRightOn:
28:             LightFrontRight=ON;
29:             break;
30:         case CccLightFrontRightOff:
31:             LightFrontRight=OFF;
32:             break;
33:     }
34: }
35:
36: // Se recibió una instrucción para establecer el estado del movimiento de
la base
37: void SetBaseMv(){
38:     Rcv=ReadSPWait();
39:     switch(Rcv) {
40:         case CccBaseMvFwd:
41:             BaseMvFwd();
42:             break;
43:         case CccBaseMvStop:
44:             BaseStop();
45:             break;
46:         case CccBaseMvBck:
47:             BaseMvBck();
48:             break;
49:         case CccBaseTurnL:
50:             BaseTurnL();
51:             break;
52:         case CccBaseTurnR:
53:             BaseTurnR();
54:             break;
55:         case CccBaseRotateL:
56:             BaseRotateL();
57:             break;
58:         case CccBaseRotateR:
59:             BaseRotateR();
60:             break;
```

```
61:         }
62:     }
63:
64:     // Procesar cualquier tipo de datos recibidos
65:     void ReceiveData() {
66:         Rcv=ReadSP();
67:         if(Rcv != 0xFF) {
68:             switch(Rcv) {
69:                 case CcAuthID:
70:                     Authenticate();
71:                     break;
72:                 case CcSetLight:
73:                     SetLight();
74:                     break;
75:                 case CcSetBaseMovement:
76:                     SetBaseMv();
77:                     break;
78:             }
79:         }
80:     }
```

```

1:  /*=====
2:  * Robot de dos ruedas motrices controlado desde ordenador
3:  *
4:  * Conexiones:
5:  * CMSP-MAX MB_TX a MBP40 TX/CK (pin 25)
6:  * CMSP-MAX MB_RX a MBP40 RX/DT (pin 26)
7:  * Ver otras conexiones más abajo
8:  *
9:  * Adrián Bulnes Parra
10:  *=====*/
11:
12: #include <MBP40.h>
13:
14: // Lista de códigos de control. Éste es el primer byte transmitido por el
    puerto serie e indica la operación que se desea ejecutar
15: // #define CcRESERVED 0
16: #define CcAuthID 1
17: #define CcSetLight 10
18: #define CcSetBaseMovement 11
19: #define CcStatusLight 100
20: #define CcStatusBaseMovement 101
21: #define CcStatusBumper 102
22: // #define CcRESERVED 255
23:
24: // Códigos usados por operaciones individuales:
25:
26: // CcAuthID - Identificación
27: #define CccAuthID 13
28:
29: // CcSetLight, CcStatusLight - Recibir/enviar estado de una luz
30: #define CccLightFrontLeftOn 50
31: #define CccLightFrontLeftOff 51
32: #define CccLightFrontRightOn 52
33: #define CccLightFrontRightOff 53
34:
35: // CcSetBaseMovement, CcStatusBaseMovement - Recibir/enviar estado del mo
    vimiento de la base
36: #define CccBaseMvFwd 20
37: #define CccBaseMvStop 21
38: #define CccBaseMvBck 22
39: #define CccBaseTurnL 23
40: #define CccBaseTurnR 24
41: #define CccBaseRotateL 25
42: #define CccBaseRotateR 26
43:
44: // CcStatusBumper - Enviar estado de un búmper
45: #define CccBumperFrontLeftPressed 70
46: #define CccBumperFrontLeftReleased 71
47: #define CccBumperFrontRightPressed 72
48: #define CccBumperFrontRightReleased 73
49:
50: //=====
51:
52: void ProcessSensors();
53: void WriteSP(unsigned int8);
54:
55: #define OSC_8MHz
56: #include <Delays.h>
57:
58: // Luces
59: #define LightFrontLeft RD0

```

```

60: #define LightFrontRight RD1
61: #define TestLed1 RD2
62: #define TestLed2 RD3
63: #define TestLed3 RD4
64: #define TestLed4 RA4
65: #define ON 1
66: #define OFF 0
67:
68: // Búmpers
69: #define BumperFrontLeft RB2
70: #define BumperFrontRight RB1
71: #define Pressed 0
72: #define Released 1
73:
74: // Movimiento de la base
75: #define M1Enable RE0
76: #define M1In1 RE1
77: #define M1In2 RE2
78: #define M2Enable RD7
79: #define M2In1 RD6
80: #define M2In2 RD5
81: #include <L293.h>
82: #include "BaseMovement.h"
83:
84: #include "SerialPort.h"
85: #include "Communication.h"
86:
87: ConfigBits1(_CP_OFF & _DEBUG_OFF & _FCMEN_OFF & _IESO_OFF & _CPD_OFF & _
LVP_OFF & _BOR_OFF & _MCLR_OFF & _PWRTE_ON & _WDT_OFF & _INTOSCIO);
88:
89: void Setup(){
90:     SetIntosc8MHz();
91:     AdcDisable();
92:
93:     //configurar UART
94:     //Baud rate timer: SPBRG=((Fosc/(16 * 9600) -1))
95:     //Baud rate = Fosc/(16*(SPBRG+1))
96:     //baud rate real para Fosc=8MHz: 9615.384615 bauds
97:     // SPRG=51,
98:     SPBRGH=0;
99:     SPBRG=51;
100:    BRGH=1; //alta velocidad
101:    CREN=1; //activar receptor
102:    TXEN=1; // (registro TXSTA) activar transmisor
103:    SYNC=0; // (registro TXSTA) operación asíncrona
104:    SPEN=1; // (registro RCSTA) configura TX/CK como salida, RX/DT en
trada
105:    //...por defecto: 8 bits, sin interrupciones
106:
107:    // Conexiones del control de motores
108:    TRISE0=DigitalOutput;
109:    TRISE1=DigitalOutput;
110:    TRISE2=DigitalOutput;
111:    TRISD7=DigitalOutput;
112:    TRISD6=DigitalOutput;
113:    TRISD5=DigitalOutput;
114:
115:    // Conexiones de luces
116:    TRISD0=DigitalOutput;
117:    TRISD1=DigitalOutput;
118:    TRISD2=DigitalOutput;

```

```
119:         TRISD3=DigitalOutput;
120:         TRISD4=DigitalOutput;
121:         TRISA4=DigitalOutput;
122:
123:         // Conexiones de bmpers
124:         TRISB2=DigitalInput;
125:         TRISB1=DigitalInput;
126:
127:         // estado por defecto
128:         BaseStop();
129:         LightFrontLeft=OFF;
130:         LightFrontRight=OFF;
131:         TestLed1=OFF;
132:         TestLed2=OFF;
133:         TestLed3=OFF;
134:         TestLed4=OFF;
135:     }
136:
137:     bool PrevBumperFrontLeft=Released;
138:     bool PrevBumperFrontRight=Released;
139:
140:     void ProcessSensors() {
141:         Delay(20);
142:         if(BumperFrontLeft==Pressed && PrevBumperFrontLeft==Released) {
143:             BaseStop();
144:             WriteSP(CcStatusBumper);
145:             WriteSP(CccBumperFrontLeftPressed);
146:         }
147:         if(BumperFrontLeft==Released && PrevBumperFrontLeft==Pressed) {
148:             WriteSP(CcStatusBumper);
149:             WriteSP(CccBumperFrontLeftReleased);
150:         }
151:         PrevBumperFrontLeft=BumperFrontLeft;
152:
153:         if(BumperFrontRight==Pressed && PrevBumperFrontRight==Released) {
154:             BaseStop();
155:             //TestLed1=ON;
156:             WriteSP(CcStatusBumper);
157:             WriteSP(CccBumperFrontRightPressed);
158:         }
159:         if(BumperFrontRight==Released && PrevBumperFrontRight==Pressed) {
160:             //TestLed1=OFF;
161:             WriteSP(CcStatusBumper);
162:             WriteSP(CccBumperFrontRightReleased);
163:         }
164:         PrevBumperFrontRight=BumperFrontRight;
165:     }
166:
167:     void main() {
168:         Setup();
169:
170:         while(1) {
171:             ReceiveData();
172:             ProcessSensors();
173:         }
174:     }
```

```
1:  /*=====
2:  * Robot de dos ruedas motrices controlado desde ordenador
3:  *
4:  * Librería para el Puerto Serie
5:  *
6:  * Adrián Bulnes Parra
7:  *=====*/
8:
9:  // Limpiar errores UART
10: void ClearErrors(){
11:     unsigned int8 dummy;
12:
13:     if (OERR) {
14:         TXEN=0;
15:         TXEN=1;
16:         CREN=0;
17:         CREN=1;
18:     }
19:     if (FERR) {
20:         dummy=RCREG;
21:         TXEN=0;
22:         TXEN=1;
23:     }
24: }
25:
26: // Lee un byte del puerto serie. Si no hay datos en el búfer de recepción
, espera indefinidamente al próximo byte recibido
27: unsigned int8 ReadSPWait(){
28:     while(!RCIF) {
29:         ClearErrors();
30:     }
31:     return RCREG;
32: }
33:
34: // Lee un byte del puerto serie. Si no hay datos en el búfer de recepción
, devuelve 0xFF
35: unsigned int8 ReadSP() {
36:     if(RCIF) return RCREG;
37:     else {
38:         ClearErrors();
39:         return 0xFF;
40:     }
41: }
42:
43: // Envía un byte a través del puerto serie
44: void WriteSP(unsigned int8 value) {
45:     // espera a que termine la anterior transmisión
46:     while(!TXIF) {
47:         //no hacer nada
48:     }
49:     TXREG=value;
50:     Delay10us();
51: }
```

Anexo XXVII

Tipo: Código fuente (programa para ordenador)

Programa: PCBot

Lenguaje: C#

Archivos de código: 10


```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5: using System.Threading;
6: using SWF = System.Windows.Forms;
7:
8: namespace PCBot_PCAApp {
9:     /// <summary>
10:    /// Estado del Puerto Serie
11:    /// </summary>
12:    public enum Status {
13:        Disconnected,
14:        Connected
15:    }
16:
17:    /// <summary>
18:    /// Bytes de control del puerto serie. Indican la operación a realizar a continuación
19:    /// </summary>
20:    public enum ControlByte {
21:        /// <summary>
22:        /// Pregunta ID de autenticación
23:        /// </summary>
24:        AuthID = 1,
25:
26:        /// <summary>
27:        /// Establece el estado de una luz
28:        /// </summary>
29:        SetLight = 10,
30:
31:        /// <summary>
32:        /// Establece el estado del movimiento de la base
33:        /// </summary>
34:        SetBaseMovement = 11,
35:
36:        /// <summary>
37:        /// Obtiene el estado de una luz
38:        /// </summary>
39:        StatusLight = 100,
```

```
40:
41:         /// <summary>
42:         /// Obtiene el estado del movimiento de la base
43:         /// </summary>
44:         StatusBaseMovement = 101,
45:
46:         /// <summary>
47:         /// Obtiene el estado de un búmer
48:         /// </summary>
49:         StatusBumpers = 102,
50:     }
51:
52:     public enum ControlByteLight {
53:         FrontLeftOn = 50,
54:         FrontLeftOff = 51,
55:         FrontRightOn = 52,
56:         FrontRightOff = 53
57:     }
58:
59:     public enum ControlByteBaseMv {
60:         Forward = 20,
61:         Stop = 21,
62:         Backwards = 22,
63:         TurnLeft = 23,
64:         TurnRight = 24,
65:         RotateLeft = 25,
66:         RotateRight = 26
67:     }
68:
69:     public enum ControlByteBumper {
70:         FrontLeftPressed = 70,
71:         FrontLeftReleased = 71,
72:         FrontRightPressed = 72,
73:         FrontRightReleased = 73
74:     }
75:
76:     class Dummy {
77:     }
78:
```

```
79:         public static class Communication {
80:             static SWF.Timer AuthTimer;
81:             static DateTime LastAuthentication = new DateTime(1900, 1, 1);
82:             static Dummy s = new Dummy();
83:             public static bool Authenticated { get; private set; }
84:
85:             public static Status Status {
86:                 get {
87:                     return _Status;
88:                 }
89:                 private set {
90:                     _Status = value;
91:                     if(Global.MainWindow != null) Global.MainWindow.SetConnectionStatus(value);
92:                     if(value == Status.Disconnected) Authenticated = false;
93:                 }
94:             }
95:             private static Status _Status = Status.Disconnected;
96:
97:             static SerialPort SP;
98:
99:             /// <summary>
100:            /// Hilo para hacer polling al Puerto Serie en busca de nuevos datos recibidos
101:            /// </summary>
102:            static Thread ThreadReceiver;
103:
104:            static Communication() {
105:                AuthTimer = new SWF.Timer();
106:                AuthTimer.Tick += new EventHandler(AuthTimer_Tick);
107:                AuthTimer.Interval = 1000;
108:                AuthTimer.Start();
109:            }
110:
111:            /// <summary>
112:            /// Conectar al microcontrolador
113:            /// </summary>
114:            public static void Connect() {
115:                Global.Log("Trying to connect");
116:                if(Communication.SP != null && Communication.SP.IsOpen) Communication.SP.Close();
117:                Communication.SP = new SerialPort(Settings.SpName, Settings.BaudRate, System.IO.Ports.Pari
```

```
ty.None, 8, System.IO.Ports.StopBits.One);
118:         try {
119:             Communication.SP.Open();
120:             Global.Log("Connected to " + Settings.SpName);
121:             if(ThreadReceiver == null) {
122:                 ThreadReceiver = new Thread(Communication.DataReceiver);
123:                 ThreadReceiver.Start();
124:             }
125:             //Check();
126:         } catch(Exception e) {
127:             Global.Log("Unable to connect: " + e.Message);
128:             Status = Status.Disconnected;
129:         }
130:     }
131:
132:     static void AuthTimer_Tick(object sender, EventArgs e) {
133:         // pedir autenticación
134:         Check();
135:
136:         if((DateTime.Now - LastAuthentication).TotalSeconds > 2.5) {
137:             Global.Log("Not authenticated for too long...");
138:             //Disconnect();
139:             Authenticated = false;
140:         }
141:
142:         if(Status == Status.Disconnected) Connect();
143:     }
144:
145:     public static void Disconnect() {
146:         if(ThreadReceiver != null) {
147:             ThreadReceiver.Abort();
148:             ThreadReceiver = null;
149:         }
150:         if(SP != null) {
151:             SP.Close();
152:             SP.Dispose();
153:             SP = null;
154:         }
155:         Authenticated = false;
    }
```

```
156:         Status = Status.Disconnected;
157:         Global.Log("Disconnected");
158:     }
159:
160:     // ¡¡¡ÉSTE NO DEBE CONTENER UN LOCK!!!
161:     public static void Send(byte B) {
162:         if(SP != null && SP.IsOpen) {
163:             try {
164:                 SP.WriteByte(B);
165:                 Global.Log("Byte sent: " + B.ToString());
166:                 Thread.Sleep(10);
167:             } catch(Exception e) {
168:                 Global.Log("Error while sending byte \"" + B.ToString() + "\": " + e.Message);
169:             }
170:         } else {
171:             Global.Log("Unable to send byte: " + B.ToString());
172:         }
173:     }
174:
175:     public static void Check() {
176:         lock(s) {
177:             Send(ControlByte.AuthID);
178:         }
179:     }
180:
181:     public static void Send(ControlByte CB) {
182:         lock(s) {
183:             Send((byte)CB);
184:         }
185:     }
186:
187:     public static void SetLight(ControlByteLight CBL) {
188:         lock(s) {
189:             Send(ControlByte.SetLight);
190:             Send((byte)CBL);
191:         }
192:     }
193:
```

```

194:         static DateTime LastBaseMvChange = DateTime.Now;
195:
196:         public static void SetBaseMv(ControlByteBaseMv CBB) {
197:             //no enviar cambios demasiado rápido
198:             if((DateTime.Now - LastBaseMvChange).TotalMilliseconds < 400) return;
199:
200:             lock(s) {
201:                 Send(ControlByte.SetBaseMovement);
202:                 Send((byte)CBB);
203:             }
204:             LastBaseMvChange = DateTime.Now;
205:         }
206:
207:         /// <summary>
208:         /// Función ejecutada en un segundo hilo que hace polling al puerto serie y procesa los datos recibidos
209:         /// </summary>
210:         public static void DataReceiver() {
211:             Global.Log("Starting receiver thread");
212:             do {
213:                 while(SP != null && SP.IsOpen && Global.MainWindow != null) {
214:                     try {
215:                         byte Rcv = (byte)SP.ReadByte();
216:                         ControlByte BC = (ControlByte)Rcv;
217:                         if(BC == ControlByte.AuthID) {
218:                             Rcv = (byte)SP.ReadByte();
219:                             if(Rcv == Settings.RobotID) {
220:                                 // auth success
221:                                 Status = Status.Connected;
222:                                 Authenticated = true;
223:                                 LastAuthentication = DateTime.Now;
224:                             } else {
225:                                 Global.Log("Authentication failure, disconnecting.");
226:                                 Disconnect();
227:                             }
228:                         }
229:                     }
230:                     if(Authenticated) {

```

```

231:                                     switch(BC) {
232:                                         case ControlByte.AuthID:
233:                                             break;
234:                                         case ControlByte.StatusBumpers:
235:                                             Global.Log("Received bumper status...");
236:                                             Rcv = (byte)SP.ReadByte();
237:                                             ControlByteBumper BumperStatus = ((ControlByteBumper)Enum.Parse(typeof(ControlByteBumper), Rcv.ToString()));
238:                                             Global.Log("..." + BumperStatus.ToString());
239:                                             if(Global.MainWindow != null) Global.MainWindow.SetBumperStatus(BumperStatus);
240:                                             break;
241:                                         case ControlByte.StatusBaseMovement:
242:                                             Global.Log("Received base movement status...");
243:                                             Rcv = (byte)SP.ReadByte();
244:                                             ControlByteBaseMv BaseMvStatus = ((ControlByteBaseMv)Enum.Parse(typeof(ControlByteBaseMv), Rcv.ToString()));
245:                                             Global.Log("..." + BaseMvStatus.ToString());
246:                                             if(Global.MainWindow != null) Global.MainWindow.SetBaseMvStatus(BaseMvStatus);
247:                                             break;
248:                                         default:
249:                                             Global.Log("Unknown control byte: " + BC.ToString());
250:                                             break;
251:                                     } else {
252:                                         Global.Log("Not authenticated, ignoring received byte: " + Rcv);
253:                                     }
254:                                     } catch (Exception e) {
255:                                         Global.Log("Error receiving data: " + e.Message);
256:                                     }
257:                                 }
258:                                 Thread.Sleep(100);
259:                             } while(true);
260:

```

```
261:                                Global.Log("Closing receiver thread");
262:                                Disconnect(); //evitar estar conectado sin un DataReceiver ejecutándose
263:                                }
264:                                }
265: }
```



```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace PCBot_PCAApp {
7:     public static class UInt16Extensions {
8:         public static byte[] GetBytes(this UInt16 n) {
9:             byte[] bytes = new byte[2];
10:            bytes[1] = (byte)(n >> 8);
11:            bytes[0] = (byte)n;
12:            return bytes;
13:        }
14:    }
15:
16:    public static class UInt32Extensions {
17:        public static byte[] GetBytes(this UInt32 n) {
18:            byte[] bytes = new byte[4];
19:            bytes[3] = (byte)(n >> 24);
20:            bytes[2] = (byte)(n >> 16);
21:            bytes[1] = (byte)(n >> 8);
22:            bytes[0] = (byte)n;
23:            return bytes;
24:        }
25:    }
26: }
```

```
1: // código generado automáticamente por Visual Studio
2:
3: namespace PCBot_PCAApp {
4:     partial class FrmAbout {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
>
14:        protected override void Dispose(bool disposing) {
15:            if(disposing && (components != null)) {
16:                components.Dispose();
17:            }
18:            base.Dispose(disposing);
19:        }
20:
21:        #region Windows Form Designer generated code
22:
23:        /// <summary>
24:        /// Required method for Designer support - do not modify
25:        /// the contents of this method with the code editor.
26:        /// </summary>
27:        private void InitializeComponent() {
28:            this.textBox1 = new System.Windows.Forms.TextBox();
29:            this.SuspendLayout();
30:            //
31:            // textBox1
32:            //
33:            this.textBox1.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles
Styles.Top | System.Windows.Forms.AnchorStyles.Bottom)
34:                | System.Windows.Forms.AnchorStyles.Left)
35:                | System.Windows.Forms.AnchorStyles.Right)));
36:            this.textBox1.BorderStyle = System.Windows.Forms.BorderStyle.None;
37:            this.textBox1.Location = new System.Drawing.Point(12, 12);
```

```
38:         this.textBox1.Multiline = true;
39:         this.textBox1.Name = "textBox1";
40:         this.textBox1.ReadOnly = true;
41:         this.textBox1.Size = new System.Drawing.Size(268, 99);
42:         this.textBox1.TabIndex = 0;
43:         this.textBox1.Text = "Graphical interface for controlling a two-wheeled robot from a compu
ter.\r\n\r\nAutho" +
44:         "r: Adrián Bulnes, aka Urriellu\r\n\r\nhttp://community.curuxa.org\r\nhttp://urrie
llu.n" +
45:         "et";
46:         this.textBox1.Enter += new System.EventHandler(this.textBox1_Enter);
47:         //
48:         // FrmAbout
49:         //
50:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
51:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
52:         this.ClientSize = new System.Drawing.Size(292, 123);
53:         this.Controls.Add(this.textBox1);
54:         this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;
55:         this.MaximizeBox = false;
56:         this.MinimizeBox = false;
57:         this.Name = "FrmAbout";
58:         this.Text = "About...";
59:         this.ResumeLayout(false);
60:         this.PerformLayout();
61:
62:     }
63:
64:     #endregion
65:
66:     private System.Windows.Forms.TextBox textBox1;
67: }
68: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.ComponentModel;
4: using System.Data;
5: using System.Drawing;
6: using System.Linq;
7: using System.Text;
8: using System.Windows.Forms;
9:
10: namespace PCBot_PCAApp {
11:     public partial class FrmAbout:Form {
12:         public FrmAbout() {
13:             InitializeComponent();
14:         }
15:
16:         private void textBox1_Enter(object sender, EventArgs e) {
17:             textBox1.Select(0, 0);
18:             this.Focus();
19:         }
20:     }
21: }
```

```
1: // código generado automáticamente por Visual Studio
2:
3: namespace PCBot_PCAApp {
4:     partial class FrmMainWindow {
5:         /// <summary>
6:         /// Required designer variable.
7:         /// </summary>
8:         private System.ComponentModel.IContainer components = null;
9:
10:        /// <summary>
11:        /// Clean up any resources being used.
12:        /// </summary>
13:        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
14:
15:        protected override void Dispose(bool disposing) {
16:            if(disposing && (components != null)) {
17:                components.Dispose();
18:            }
19:            base.Dispose(disposing);
20:        }
21:
22:        #region Windows Form Designer generated code
23:
24:        /// <summary>
25:        /// Required method for Designer support - do not modify
26:        /// the contents of this method with the code editor.
27:        /// </summary>
28:        private void InitializeComponent() {
29:            System.ComponentModel.ComponentResourceManager resources = new System.ComponentModel.Compo
30:            nentResourceManager(typeof(FrmMainWindow));
31:            this.LblConnStatus = new System.Windows.Forms.Label();
32:            this.BtnLightFrontLeftOn = new System.Windows.Forms.Button();
33:            this.BtnLightFrontLeftOff = new System.Windows.Forms.Button();
34:            this.ImgLightFrontLeftOff = new System.Windows.Forms.PictureBox();
35:            this.ImgLightFrontLeftOn = new System.Windows.Forms.PictureBox();
36:            this.ImgBody = new System.Windows.Forms.PictureBox();
37:            this.ImgBumperFrontLeft = new System.Windows.Forms.PictureBox();
38:            this.ImgBumperFrontRight = new System.Windows.Forms.PictureBox();
39:            this.ImgLightFrontRightOff = new System.Windows.Forms.PictureBox();
```

```
38:         this.BtnLightFrontRightOff = new System.Windows.Forms.Button();
39:         this.BtnLightFrontRightOn = new System.Windows.Forms.Button();
40:         this.ImgLightFrontRightOn = new System.Windows.Forms.PictureBox();
41:         this.BtnAbout = new System.Windows.Forms.Button();
42:         this.BtnLightFrontOn = new System.Windows.Forms.Button();
43:         this.BtnLightFrontOff = new System.Windows.Forms.Button();
44:         this.BtnBaseMvFwd = new System.Windows.Forms.Button();
45:         this.ImgBaseMvFwd = new System.Windows.Forms.PictureBox();
46:         this.BtnBaseMvTurnL = new System.Windows.Forms.Button();
47:         this.BtnBaseMvTurnR = new System.Windows.Forms.Button();
48:         this.BtnBaseMvRotateR = new System.Windows.Forms.Button();
49:         this.BtnBaseMvRotateL = new System.Windows.Forms.Button();
50:         this.BtnBaseMvBck = new System.Windows.Forms.Button();
51:         this.BtnBaseMvStop = new System.Windows.Forms.Button();
52:         this.ImgBaseStop = new System.Windows.Forms.PictureBox();
53:         this.ImgBaseRotateL = new System.Windows.Forms.PictureBox();
54:         this.ImgBaseRorateR = new System.Windows.Forms.PictureBox();
55:         this.ImgBaseTurnL = new System.Windows.Forms.PictureBox();
56:         this.ImgBaseTurnR = new System.Windows.Forms.PictureBox();
57:         this.ImgBaseMvBck = new System.Windows.Forms.PictureBox();
58:         this.BtnInteractive = new System.Windows.Forms.Button();
59:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontLeftOff)).BeginInit();
60:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontLeftOn)).BeginInit();
61:         ((System.ComponentModel.ISupportInitialize)(this.ImgBody)).BeginInit();
62:         ((System.ComponentModel.ISupportInitialize)(this.ImgBumperFrontLeft)).BeginInit();
63:         ((System.ComponentModel.ISupportInitialize)(this.ImgBumperFrontRight)).BeginInit();
64:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontRightOff)).BeginInit();
65:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontRightOn)).BeginInit();
66:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseMvFwd)).BeginInit();
67:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseStop)).BeginInit();
68:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseRotateL)).BeginInit();
69:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseRorateR)).BeginInit();
70:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseTurnL)).BeginInit();
71:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseTurnR)).BeginInit();
72:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseMvBck)).BeginInit();
73:         this.SuspendLayout();
74:         //
75:         // LblConnStatus
76:         //
```

```
77:         this.LblConnStatus.AutoSize = true;
78:         this.LblConnStatus.Location = new System.Drawing.Point(12, 9);
79:         this.LblConnStatus.Name = "LblConnStatus";
80:         this.LblConnStatus.Size = new System.Drawing.Size(148, 13);
81:         this.LblConnStatus.TabIndex = 0;
82:         this.LblConnStatus.Text = "Connection status (NOT SET)";
83:         //
84:         // BtnLightFrontLeftOn
85:         //
86:         this.BtnLightFrontLeftOn.Location = new System.Drawing.Point(128, 39);
87:         this.BtnLightFrontLeftOn.Name = "BtnLightFrontLeftOn";
88:         this.BtnLightFrontLeftOn.Size = new System.Drawing.Size(38, 23);
89:         this.BtnLightFrontLeftOn.TabIndex = 2;
90:         this.BtnLightFrontLeftOn.Text = "ON";
91:         this.BtnLightFrontLeftOn.UseVisualStyleBackColor = true;
92:         this.BtnLightFrontLeftOn.Click += new System.EventHandler(this.BtnLightFrontLeftOn_Click);
93:         //
94:         // BtnLightFrontLeftOff
95:         //
96:         this.BtnLightFrontLeftOff.Location = new System.Drawing.Point(128, 66);
97:         this.BtnLightFrontLeftOff.Name = "BtnLightFrontLeftOff";
98:         this.BtnLightFrontLeftOff.Size = new System.Drawing.Size(38, 23);
99:         this.BtnLightFrontLeftOff.TabIndex = 3;
100:        this.BtnLightFrontLeftOff.Text = "OFF";
101:        this.BtnLightFrontLeftOff.UseVisualStyleBackColor = true;
102:        this.BtnLightFrontLeftOff.Click += new System.EventHandler(this.BtnLightFrontLeftOff_Click);
103:    };
104:    //
105:    // ImgLightFrontLeftOff
106:    //
107:    this.ImgLightFrontLeftOff.Image = ((System.Drawing.Image)(resources.GetObject("ImgLightFrontLeftOff.Image")));
108:    this.ImgLightFrontLeftOff.Location = new System.Drawing.Point(85, 39);
109:    this.ImgLightFrontLeftOff.Name = "ImgLightFrontLeftOff";
110:    this.ImgLightFrontLeftOff.Size = new System.Drawing.Size(37, 50);
111:    this.ImgLightFrontLeftOff.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
112:    this.ImgLightFrontLeftOff.TabIndex = 4;
113:    this.ImgLightFrontLeftOff.TabStop = false;
114:    //
```

```
114:         // ImgLightFrontLeftOn
115:         //
116:         this.ImgLightFrontLeftOn.Image = ((System.Drawing.Image)(resources.GetObject("ImgLightFrontLeftOn.Image")));
117:         this.ImgLightFrontLeftOn.Location = new System.Drawing.Point(85, 39);
118:         this.ImgLightFrontLeftOn.Name = "ImgLightFrontLeftOn";
119:         this.ImgLightFrontLeftOn.Size = new System.Drawing.Size(37, 50);
120:         this.ImgLightFrontLeftOn.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
121:         this.ImgLightFrontLeftOn.TabIndex = 5;
122:         this.ImgLightFrontLeftOn.TabStop = false;
123:         //
124:         // ImgBody
125:         //
126:         this.ImgBody.Image = ((System.Drawing.Image)(resources.GetObject("ImgBody.Image")));
127:         this.ImgBody.Location = new System.Drawing.Point(15, 95);
128:         this.ImgBody.Name = "ImgBody";
129:         this.ImgBody.Size = new System.Drawing.Size(372, 441);
130:         this.ImgBody.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
131:         this.ImgBody.TabIndex = 6;
132:         this.ImgBody.TabStop = false;
133:         //
134:         // ImgBumperFrontLeft
135:         //
136:         this.ImgBumperFrontLeft.Image = ((System.Drawing.Image)(resources.GetObject("ImgBumperFrontLeft.Image")));
137:         this.ImgBumperFrontLeft.Location = new System.Drawing.Point(15, 65);
138:         this.ImgBumperFrontLeft.Name = "ImgBumperFrontLeft";
139:         this.ImgBumperFrontLeft.Size = new System.Drawing.Size(64, 30);
140:         this.ImgBumperFrontLeft.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
141:         this.ImgBumperFrontLeft.TabIndex = 7;
142:         this.ImgBumperFrontLeft.TabStop = false;
143:         //
144:         // ImgBumperFrontRight
145:         //
146:         this.ImgBumperFrontRight.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
147:         this.ImgBumperFrontRight.Image = ((System.Drawing.Image)(resources.GetObject("ImgBumperFrontRight.Image")));
148:         this.ImgBumperFrontRight.Location = new System.Drawing.Point(323, 65);
```



```
149:         this.ImgBumperFrontRight.Name = "ImgBumperFrontRight";
150:         this.ImgBumperFrontRight.Size = new System.Drawing.Size(64, 30);
151:         this.ImgBumperFrontRight.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
152:         this.ImgBumperFrontRight.TabIndex = 8;
153:         this.ImgBumperFrontRight.TabStop = false;
154:         //
155:         // ImgLightFrontRightOff
156:         //
157:         this.ImgLightFrontRightOff.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Fo
orms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
158:         this.ImgLightFrontRightOff.Image = (System.Drawing.Image)(resources.GetObject("ImgLightFr
ontRightOff.Image"));
159:         this.ImgLightFrontRightOff.Location = new System.Drawing.Point(280, 39);
160:         this.ImgLightFrontRightOff.Name = "ImgLightFrontRightOff";
161:         this.ImgLightFrontRightOff.Size = new System.Drawing.Size(37, 50);
162:         this.ImgLightFrontRightOff.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
163:         ;
164:         this.ImgLightFrontRightOff.TabIndex = 11;
165:         this.ImgLightFrontRightOff.TabStop = false;
166:         //
167:         // BtnLightFrontRightOff
168:         //
169:         this.BtnLightFrontRightOff.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Fo
orms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
170:         this.BtnLightFrontRightOff.Location = new System.Drawing.Point(236, 66);
171:         this.BtnLightFrontRightOff.Name = "BtnLightFrontRightOff";
172:         this.BtnLightFrontRightOff.Size = new System.Drawing.Size(38, 23);
173:         this.BtnLightFrontRightOff.TabIndex = 10;
174:         this.BtnLightFrontRightOff.Text = "OFF";
175:         this.BtnLightFrontRightOff.UseVisualStyleBackColor = true;
176:         this.BtnLightFrontRightOff.Click += new System.EventHandler(this.BtnLightFrontRightOff_Cli
ck);
177:         //
178:         // BtnLightFrontRightOn
179:         //
180:         this.BtnLightFrontRightOn.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Fo
orms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
181:         this.BtnLightFrontRightOn.Location = new System.Drawing.Point(236, 39);
182:         this.BtnLightFrontRightOn.Name = "BtnLightFrontRightOn";
```

```
182:         this.BtnLightFrontRightOn.Size = new System.Drawing.Size(38, 23);
183:         this.BtnLightFrontRightOn.TabIndex = 9;
184:         this.BtnLightFrontRightOn.Text = "ON";
185:         this.BtnLightFrontRightOn.UseVisualStyleBackColor = true;
186:         this.BtnLightFrontRightOn.Click += new System.EventHandler(this.BtnLightFrontRightOn_Click
);
187:         //
188:         // ImgLightFrontRightOn
189:         //
190:         this.ImgLightFrontRightOn.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Fo
rms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
191:         this.ImgLightFrontRightOn.Image = ((System.Drawing.Image)(resources.GetObject("ImgLightFro
ntRightOn.Image")));
192:         this.ImgLightFrontRightOn.Location = new System.Drawing.Point(280, 39);
193:         this.ImgLightFrontRightOn.Name = "ImgLightFrontRightOn";
194:         this.ImgLightFrontRightOn.Size = new System.Drawing.Size(37, 50);
195:         this.ImgLightFrontRightOn.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
196:         this.ImgLightFrontRightOn.TabIndex = 12;
197:         this.ImgLightFrontRightOn.TabStop = false;
198:         //
199:         // BtnAbout
200:         //
201:         this.BtnAbout.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorSt
yles.Top | System.Windows.Forms.AnchorStyles.Right)));
202:         this.BtnAbout.Location = new System.Drawing.Point(365, 4);
203:         this.BtnAbout.Name = "BtnAbout";
204:         this.BtnAbout.Size = new System.Drawing.Size(22, 23);
205:         this.BtnAbout.TabIndex = 13;
206:         this.BtnAbout.Text = "?";
207:         this.BtnAbout.UseVisualStyleBackColor = true;
208:         this.BtnAbout.Click += new System.EventHandler(this.BtnAbout_Click);
209:         //
210:         // BtnLightFrontOn
211:         //
212:         this.BtnLightFrontOn.Location = new System.Drawing.Point(172, 39);
213:         this.BtnLightFrontOn.Name = "BtnLightFrontOn";
214:         this.BtnLightFrontOn.Size = new System.Drawing.Size(58, 23);
215:         this.BtnLightFrontOn.TabIndex = 14;
216:         this.BtnLightFrontOn.Text = "ON";
```

```
217:         this.BtnLightFrontOn.UseVisualStyleBackColor = true;
218:         this.BtnLightFrontOn.Click += new System.EventHandler(this.BtnLightFrontOn_Click);
219:         //
220:         // BtnLightFrontOff
221:         //
222:         this.BtnLightFrontOff.Location = new System.Drawing.Point(172, 68);
223:         this.BtnLightFrontOff.Name = "BtnLightFrontOff";
224:         this.BtnLightFrontOff.Size = new System.Drawing.Size(58, 23);
225:         this.BtnLightFrontOff.TabIndex = 15;
226:         this.BtnLightFrontOff.Text = "OFF";
227:         this.BtnLightFrontOff.UseVisualStyleBackColor = true;
228:         this.BtnLightFrontOff.Click += new System.EventHandler(this.BtnLightFrontOff_Click);
229:         //
230:         // BtnBaseMvFwd
231:         //
232:         this.BtnBaseMvFwd.Image = ((System.Drawing.Image)(resources.GetObject("BtnBaseMvFwd.Image"
)))
233:         this.BtnBaseMvFwd.Location = new System.Drawing.Point(181, 193);
234:         this.BtnBaseMvFwd.Name = "BtnBaseMvFwd";
235:         this.BtnBaseMvFwd.Size = new System.Drawing.Size(38, 38);
236:         this.BtnBaseMvFwd.TabIndex = 16;
237:         this.BtnBaseMvFwd.UseVisualStyleBackColor = true;
238:         this.BtnBaseMvFwd.Click += new System.EventHandler(this.BtnBaseMvFwd_Click);
239:         //
240:         // ImgBaseMvFwd
241:         //
242:         this.ImgBaseMvFwd.BackColor = System.Drawing.Color.White;
243:         this.ImgBaseMvFwd.Image = ((System.Drawing.Image)(resources.GetObject("ImgBaseMvFwd.Image"
)))
244:         this.ImgBaseMvFwd.Location = new System.Drawing.Point(166, 237);
245:         this.ImgBaseMvFwd.Name = "ImgBaseMvFwd";
246:         this.ImgBaseMvFwd.Size = new System.Drawing.Size(68, 120);
247:         this.ImgBaseMvFwd.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
248:         this.ImgBaseMvFwd.TabIndex = 17;
249:         this.ImgBaseMvFwd.TabStop = false;
250:         //
251:         // BtnBaseMvTurnL
252:         //
253:         this.BtnBaseMvTurnL.Image = ((System.Drawing.Image)(resources.GetObject("BtnBaseMvTurnL.Im
```

```
age"))));
254:         this.BtnBaseMvTurnL.Location = new System.Drawing.Point(96, 193);
255:         this.BtnBaseMvTurnL.Name = "BtnBaseMvTurnL";
256:         this.BtnBaseMvTurnL.Size = new System.Drawing.Size(38, 38);
257:         this.BtnBaseMvTurnL.TabIndex = 18;
258:         this.BtnBaseMvTurnL.UseVisualStyleBackColor = true;
259:         this.BtnBaseMvTurnL.Click += new System.EventHandler(this.BtnBaseMvTurnL_Click);
260:         //
261:         // BtnBaseMvTurnR
262:         //
263:         this.BtnBaseMvTurnR.Image = ((System.Drawing.Image)(resources.GetObject("BtnBaseMvTurnR.Im
age"))));
264:         this.BtnBaseMvTurnR.Location = new System.Drawing.Point(266, 193);
265:         this.BtnBaseMvTurnR.Name = "BtnBaseMvTurnR";
266:         this.BtnBaseMvTurnR.Size = new System.Drawing.Size(38, 38);
267:         this.BtnBaseMvTurnR.TabIndex = 19;
268:         this.BtnBaseMvTurnR.UseVisualStyleBackColor = true;
269:         this.BtnBaseMvTurnR.Click += new System.EventHandler(this.BtnBaseMvTurnR_Click);
270:         //
271:         // BtnBaseMvRotateR
272:         //
273:         this.BtnBaseMvRotateR.Image = ((System.Drawing.Image)(resources.GetObject("BtnBaseMvRotate
R.Image"))));
274:         this.BtnBaseMvRotateR.Location = new System.Drawing.Point(266, 276);
275:         this.BtnBaseMvRotateR.Name = "BtnBaseMvRotateR";
276:         this.BtnBaseMvRotateR.Size = new System.Drawing.Size(38, 38);
277:         this.BtnBaseMvRotateR.TabIndex = 20;
278:         this.BtnBaseMvRotateR.UseVisualStyleBackColor = true;
279:         this.BtnBaseMvRotateR.Click += new System.EventHandler(this.BtnBaseMvRotateR_Click);
280:         //
281:         // BtnBaseMvRotateL
282:         //
283:         this.BtnBaseMvRotateL.Image = ((System.Drawing.Image)(resources.GetObject("BtnBaseMvRotate
L.Image"))));
284:         this.BtnBaseMvRotateL.ImageAlign = System.Drawing.ContentAlignment.MiddleRight;
285:         this.BtnBaseMvRotateL.Location = new System.Drawing.Point(96, 276);
286:         this.BtnBaseMvRotateL.Name = "BtnBaseMvRotateL";
287:         this.BtnBaseMvRotateL.Size = new System.Drawing.Size(38, 38);
288:         this.BtnBaseMvRotateL.TabIndex = 21;
```

```
289:         this.BtnBaseMvRotateL.UseVisualStyleBackColor = true;
290:         this.BtnBaseMvRotateL.Click += new System.EventHandler(this.BtnBaseMvRotateL_Click);
291:         //
292:         // BtnBaseMvBck
293:         //
294:         this.BtnBaseMvBck.Image = ((System.Drawing.Image)(resources.GetObject("BtnBaseMvBck.Image"
)))
295:         this.BtnBaseMvBck.Location = new System.Drawing.Point(181, 363);
296:         this.BtnBaseMvBck.Name = "BtnBaseMvBck";
297:         this.BtnBaseMvBck.Size = new System.Drawing.Size(38, 38);
298:         this.BtnBaseMvBck.TabIndex = 22;
299:         this.BtnBaseMvBck.UseVisualStyleBackColor = true;
300:         this.BtnBaseMvBck.Click += new System.EventHandler(this.BtnBaseMvBck_Click);
301:         //
302:         // BtnBaseMvStop
303:         //
304:         this.BtnBaseMvStop.Image = ((System.Drawing.Image)(resources.GetObject("BtnBaseMvStop.Imag
e")));
305:         this.BtnBaseMvStop.Location = new System.Drawing.Point(181, 407);
306:         this.BtnBaseMvStop.Name = "BtnBaseMvStop";
307:         this.BtnBaseMvStop.Size = new System.Drawing.Size(38, 38);
308:         this.BtnBaseMvStop.TabIndex = 23;
309:         this.BtnBaseMvStop.UseVisualStyleBackColor = true;
310:         this.BtnBaseMvStop.Click += new System.EventHandler(this.BtnBaseMvStop_Click);
311:         //
312:         // ImgBaseStop
313:         //
314:         this.ImgBaseStop.BackColor = System.Drawing.Color.White;
315:         this.ImgBaseStop.Image = ((System.Drawing.Image)(resources.GetObject("ImgBaseStop.Image"))
);
316:         this.ImgBaseStop.Location = new System.Drawing.Point(140, 237);
317:         this.ImgBaseStop.Name = "ImgBaseStop";
318:         this.ImgBaseStop.Size = new System.Drawing.Size(120, 120);
319:         this.ImgBaseStop.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
320:         this.ImgBaseStop.TabIndex = 24;
321:         this.ImgBaseStop.TabStop = false;
322:         //
323:         // ImgBaseRotateL
324:         //
```

```
325:         this.ImgBaseRotateL.BackColor = System.Drawing.Color.White;
326:         this.ImgBaseRotateL.Image = ((System.Drawing.Image)(resources.GetObject("ImgBaseRotateL.Im
age"))));
327:         this.ImgBaseRotateL.Location = new System.Drawing.Point(140, 237);
328:         this.ImgBaseRotateL.Name = "ImgBaseRotateL";
329:         this.ImgBaseRotateL.Size = new System.Drawing.Size(120, 120);
330:         this.ImgBaseRotateL.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
331:         this.ImgBaseRotateL.TabIndex = 25;
332:         this.ImgBaseRotateL.TabStop = false;
333:         //
334:         // ImgBaseRorateR
335:         //
336:         this.ImgBaseRorateR.BackColor = System.Drawing.Color.White;
337:         this.ImgBaseRorateR.Image = ((System.Drawing.Image)(resources.GetObject("ImgBaseRorateR.Im
age"))));
338:         this.ImgBaseRorateR.Location = new System.Drawing.Point(140, 237);
339:         this.ImgBaseRorateR.Name = "ImgBaseRorateR";
340:         this.ImgBaseRorateR.Size = new System.Drawing.Size(120, 120);
341:         this.ImgBaseRorateR.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
342:         this.ImgBaseRorateR.TabIndex = 26;
343:         this.ImgBaseRorateR.TabStop = false;
344:         //
345:         // ImgBaseTurnL
346:         //
347:         this.ImgBaseTurnL.BackColor = System.Drawing.Color.White;
348:         this.ImgBaseTurnL.Image = ((System.Drawing.Image)(resources.GetObject("ImgBaseTurnL. Image"
)));
349:         this.ImgBaseTurnL.Location = new System.Drawing.Point(166, 237);
350:         this.ImgBaseTurnL.Name = "ImgBaseTurnL";
351:         this.ImgBaseTurnL.Size = new System.Drawing.Size(68, 120);
352:         this.ImgBaseTurnL.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
353:         this.ImgBaseTurnL.TabIndex = 27;
354:         this.ImgBaseTurnL.TabStop = false;
355:         //
356:         // ImgBaseTurnR
357:         //
358:         this.ImgBaseTurnR.BackColor = System.Drawing.Color.White;
359:         this.ImgBaseTurnR.Image = ((System.Drawing.Image)(resources.GetObject("ImgBaseTurnR. Image"
)));
```

```
360:         this.ImgBaseTurnR.Location = new System.Drawing.Point(166, 237);
361:         this.ImgBaseTurnR.Name = "ImgBaseTurnR";
362:         this.ImgBaseTurnR.Size = new System.Drawing.Size(68, 120);
363:         this.ImgBaseTurnR.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
364:         this.ImgBaseTurnR.TabIndex = 28;
365:         this.ImgBaseTurnR.TabStop = false;
366:         //
367:         // ImgBaseMvBck
368:         //
369:         this.ImgBaseMvBck.BackColor = System.Drawing.Color.White;
370:         this.ImgBaseMvBck.Image = ((System.Drawing.Image)(resources.GetObject("ImgBaseMvBck.Image"
))) );
371:         this.ImgBaseMvBck.Location = new System.Drawing.Point(166, 237);
372:         this.ImgBaseMvBck.Name = "ImgBaseMvBck";
373:         this.ImgBaseMvBck.Size = new System.Drawing.Size(68, 120);
374:         this.ImgBaseMvBck.SizeMode = System.Windows.Forms.PictureBoxSizeMode.StretchImage;
375:         this.ImgBaseMvBck.TabIndex = 29;
376:         this.ImgBaseMvBck.TabStop = false;
377:         //
378:         // BtnInteractive
379:         //
380:         this.BtnInteractive.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.An
chorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
381:         this.BtnInteractive.Font = new System.Drawing.Font("Microsoft Sans Serif", 8.25F, System.D
rawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
382:         this.BtnInteractive.ForeColor = System.Drawing.Color.Red;
383:         this.BtnInteractive.Location = new System.Drawing.Point(337, 4);
384:         this.BtnInteractive.Name = "BtnInteractive";
385:         this.BtnInteractive.Size = new System.Drawing.Size(22, 23);
386:         this.BtnInteractive.TabIndex = 30;
387:         this.BtnInteractive.Text = "i";
388:         this.BtnInteractive.UseVisualStyleBackColor = true;
389:         this.BtnInteractive.Click += new System.EventHandler(this.BtnInteractive_Click);
390:         //
391:         // FrmMainWindow
392:         //
393:         this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
394:         this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
395:         this.ClientSize = new System.Drawing.Size(399, 550);
```

```
396:         this.Controls.Add(this.BtnInteractive);
397:         this.Controls.Add(this.ImgBaseMvBck);
398:         this.Controls.Add(this.ImgBaseTurnR);
399:         this.Controls.Add(this.ImgBaseTurnL);
400:         this.Controls.Add(this.ImgBaseMvFwd);
401:         this.Controls.Add(this.ImgBaseRorateR);
402:         this.Controls.Add(this.ImgBaseRotateL);
403:         this.Controls.Add(this.BtnBaseMvStop);
404:         this.Controls.Add(this.BtnBaseMvBck);
405:         this.Controls.Add(this.BtnBaseMvRotateL);
406:         this.Controls.Add(this.BtnBaseMvRotateR);
407:         this.Controls.Add(this.BtnBaseMvTurnR);
408:         this.Controls.Add(this.BtnBaseMvTurnL);
409:         this.Controls.Add(this.BtnBaseMvFwd);
410:         this.Controls.Add(this.BtnLightFrontOff);
411:         this.Controls.Add(this.BtnLightFrontOn);
412:         this.Controls.Add(this.BtnAbout);
413:         this.Controls.Add(this.BtnLightFrontRightOff);
414:         this.Controls.Add(this.BtnLightFrontRightOn);
415:         this.Controls.Add(this.ImgBumperFrontRight);
416:         this.Controls.Add(this.ImgBumperFrontLeft);
417:         this.Controls.Add(this.ImgLightFrontLeftOff);
418:         this.Controls.Add(this.BtnLightFrontLeftOff);
419:         this.Controls.Add(this.BtnLightFrontLeftOn);
420:         this.Controls.Add(this.LblConnStatus);
421:         this.Controls.Add(this.ImgLightFrontLeftOn);
422:         this.Controls.Add(this.ImgLightFrontRightOff);
423:         this.Controls.Add(this.ImgLightFrontRightOn);
424:         this.Controls.Add(this.ImgBaseStop);
425:         this.Controls.Add(this.ImgBody);
426:         this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;
427:         this.KeyPreview = true;
428:         this.MaximizeBox = false;
429:         this.Name = "FrmMainWindow";
430:         this.SizeGripStyle = System.Windows.Forms.SizeGripStyle.Hide;
431:         this.Text = "PCBot";
432:         this.Load += new System.EventHandler(this.MainWindow_Load);
433:         this.FormClosed += new System.Windows.Forms.FormClosedEventHandler(this.FrmMainWindow_Form
Closed);
```



```
434:         this.KeyDown += new System.Windows.Forms.KeyEventHandler(this.FrmMainWindow_KeyDown);
435:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontLeftOff)).EndInit();
436:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontLeftOn)).EndInit();
437:         ((System.ComponentModel.ISupportInitialize)(this.ImgBody)).EndInit();
438:         ((System.ComponentModel.ISupportInitialize)(this.ImgBumperFrontLeft)).EndInit();
439:         ((System.ComponentModel.ISupportInitialize)(this.ImgBumperFrontRight)).EndInit();
440:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontRightOff)).EndInit();
441:         ((System.ComponentModel.ISupportInitialize)(this.ImgLightFrontRightOn)).EndInit();
442:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseMvFwd)).EndInit();
443:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseStop)).EndInit();
444:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseRotateL)).EndInit();
445:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseRotateR)).EndInit();
446:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseTurnL)).EndInit();
447:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseTurnR)).EndInit();
448:         ((System.ComponentModel.ISupportInitialize)(this.ImgBaseMvBck)).EndInit();
449:         this.ResumeLayout(false);
450:         this.PerformLayout();
451:
452:     }
453:
454:     #endregion
455:
456:     private System.Windows.Forms.Label lblConnStatus;
457:     private System.Windows.Forms.Button btnLightFrontLeftOn;
458:     private System.Windows.Forms.Button btnLightFrontLeftOff;
459:     private System.Windows.Forms.PictureBox imgLightFrontLeftOff;
460:     private System.Windows.Forms.PictureBox imgLightFrontLeftOn;
461:     private System.Windows.Forms.PictureBox imgBody;
462:     private System.Windows.Forms.PictureBox imgBumperFrontLeft;
463:     private System.Windows.Forms.PictureBox imgBumperFrontRight;
464:     private System.Windows.Forms.PictureBox imgLightFrontRightOff;
465:     private System.Windows.Forms.Button btnLightFrontRightOff;
466:     private System.Windows.Forms.Button btnLightFrontRightOn;
467:     private System.Windows.Forms.PictureBox imgLightFrontRightOn;
468:     private System.Windows.Forms.Button btnAbout;
469:     private System.Windows.Forms.Button btnLightFrontOn;
470:     private System.Windows.Forms.Button btnLightFrontOff;
471:     private System.Windows.Forms.Button btnBaseMvFwd;
472:     private System.Windows.Forms.PictureBox imgBaseMvFwd;
```

```
473:         private System.Windows.Forms.Button BtnBaseMvTurnL;
474:         private System.Windows.Forms.Button BtnBaseMvTurnR;
475:         private System.Windows.Forms.Button BtnBaseMvRotateR;
476:         private System.Windows.Forms.Button BtnBaseMvRotateL;
477:         private System.Windows.Forms.Button BtnBaseMvBck;
478:         private System.Windows.Forms.Button BtnBaseMvStop;
479:         private System.Windows.Forms.PictureBox ImgBaseStop;
480:         private System.Windows.Forms.PictureBox ImgBaseRotateL;
481:         private System.Windows.Forms.PictureBox ImgBaseRorateR;
482:         private System.Windows.Forms.PictureBox ImgBaseTurnL;
483:         private System.Windows.Forms.PictureBox ImgBaseTurnR;
484:         private System.Windows.Forms.PictureBox ImgBaseMvBck;
485:         private System.Windows.Forms.Button BtnInteractive;
486:     }
487: }
488:
```

```
1: using System;
2: using System.Windows.Forms;
3: using SWF = System.Windows.Forms;
4: using System.Drawing;
5: using System.Threading;
6:
7: namespace PCBot_PCAp {
8:     public partial class FrmMainWindow:Form {
9:         SWF.Timer EventsTimer = new SWF.Timer();
10:        ControlByteBaseMv BaseMvCurrentStatus = ControlByteBaseMv.Stop;
11:        bool LightFrontLeftON = false;
12:        bool LightFrontRightON = false;
13:
14:        public FrmMainWindow() {
15:            InitializeComponent();
16:            Global.MainWindow = this;
17:        }
18:
19:        public void SetConnectionStatus(Status st) {
20:            if(st == Status.Connected) {
21:                LblConnStatus.Text = "Connected";
22:                LblConnStatus.ForeColor = Color.Green;
23:            } else {
24:                LblConnStatus.Text = "Disconnected";
25:                LblConnStatus.ForeColor = Color.Red;
26:                SetDefaults();
27:            }
28:        }
29:
30:        public void SetBumperStatus(ControlByteBumper BumperStatus) {
31:            switch(BumperStatus) {
32:                case ControlByteBumper.FrontLeftPressed:
33:                    ImgBumperFrontLeft.Show();
34:                    break;
35:                case ControlByteBumper.FrontLeftReleased:
36:                    ImgBumperFrontLeft.Hide();
37:                    break;
38:                case ControlByteBumper.FrontRightPressed:
39:                    ImgBumperFrontRight.Show();
```

```
40:                                     break;
41:             case ControlByteBumper.FrontRightReleased:
42:                 ImgBumperFrontRight.Hide();
43:                 break;
44:             default:
45:                 Global.Log("Unknown bumper status: " + BumperStatus.ToString());
46:                 break;
47:         }
48:     }
49:
50:     public void SetBaseMvStatus(ControlByteBaseMv BaseMvStatus) {
51:         BaseMvCurrentStatus = BaseMvStatus;
52:         ImgBaseMvFwd.Hide();
53:         ImgBaseStop.Hide();
54:         ImgBaseMvBck.Hide();
55:         ImgBaseTurnL.Hide();
56:         ImgBaseTurnR.Hide();
57:         ImgBaseRotateL.Hide();
58:         ImgBaseRorateR.Hide();
59:         switch(BaseMvStatus) {
60:             case ControlByteBaseMv.Forward:
61:                 ImgBaseMvFwd.Show();
62:                 break;
63:             case ControlByteBaseMv.Stop:
64:                 ImgBaseStop.Show();
65:                 break;
66:             case ControlByteBaseMv.Backwards:
67:                 ImgBaseMvBck.Show();
68:                 break;
69:             case ControlByteBaseMv.TurnLeft:
70:                 ImgBaseTurnL.Show();
71:                 break;
72:             case ControlByteBaseMv.TurnRight:
73:                 ImgBaseTurnR.Show();
74:                 break;
75:             case ControlByteBaseMv.RotateLeft:
76:                 ImgBaseRotateL.Show();
77:                 break;
78:             case ControlByteBaseMv.RotateRight:
```

```
79:                 ImgBaseRorateR.Show();
80:                 break;
81:             default:
82:                 Global.Log("Unknown base movement status: " + BaseMvStatus.ToString());
83:                 break;
84:         }
85:     }
86:
87:     private void MainWindow_Load(object sender, EventArgs e) {
88:         SetConnectionStatus(Status.Disconnected); //esto también establece los valores por defecto
89:         Communication.Connect();
90:         EventsTimer.Tick += new EventHandler(EventsTimer_Tick);
91:         EventsTimer.Interval = 100;
92:         EventsTimer.Start();
93:     }
94:
95:     private void SetDefaults() {
96:         ImgBumperFrontLeft.Hide();
97:         ImgBumperFrontRight.Hide();
98:         ImgLightFrontLeftOff.BringToFront();
99:         ImgLightFrontRightOff.BringToFront();
100:         SetBaseMvStatus(ControlByteBaseMv.Stop);
101:     }
102:
103:     private void BtnLightFrontLeftOn_Click(object sender, EventArgs e) {
104:         Global.Log("Turning ON front left light");
105:         ImgLightFrontLeftOn.BringToFront();
106:         Communication.SetLight(ControlByteLight.FrontLeftOn);
107:         LightFrontLeftON = true;
108:     }
109:
110:     private void BtnLightFrontLeftOff_Click(object sender, EventArgs e) {
111:         Global.Log("Turning OFF front left light");
112:         ImgLightFrontLeftOff.BringToFront();
113:         Communication.SetLight(ControlByteLight.FrontLeftOff);
114:         LightFrontLeftON = false;
115:     }
116:
117:     private void FrmMainWindow_FormClosed(object sender, FormClosedEventArgs e) {
```

```
118:         Global.Log("Closing main window");
119:         Communication.Disconnect();
120:     }
121:
122:     private void BtnAbout_Click(object sender, EventArgs e) {
123:         FrmAbout f = new FrmAbout();
124:         f.ShowDialog(this);
125:     }
126:
127:     private void BtnLightFrontRightOn_Click(object sender, EventArgs e) {
128:         Global.Log("Turning ON front right light");
129:         ImgLightFrontRightOn.BringToFront();
130:         Communication.SetLight(ControlByteLight.FrontRightOn);
131:         LightFrontRightON = true;
132:     }
133:
134:     private void BtnLightFrontRightOff_Click(object sender, EventArgs e) {
135:         Global.Log("Turning OFF front right light");
136:         ImgLightFrontRightOff.BringToFront();
137:         Communication.SetLight(ControlByteLight.FrontRightOff);
138:         LightFrontRightON = false;
139:     }
140:
141:     private void BtnLightFrontOn_Click(object sender, EventArgs e) {
142:         BtnLightFrontLeftOn.PerformClick();
143:         Thread.Sleep(50);
144:         BtnLightFrontRightOn.PerformClick();
145:     }
146:
147:     private void BtnLightFrontOff_Click(object sender, EventArgs e) {
148:         BtnLightFrontLeftOff.PerformClick();
149:         Thread.Sleep(50);
150:         BtnLightFrontRightOff.PerformClick();
151:     }
152:
153:     private void BtnBaseMvFwd_Click(object sender, EventArgs e) {
154:         Communication.SetBaseMv(ControlByteBaseMv.Forward);
155:     }
156:
```

```
157:         private void BtnBaseMvStop_Click(object sender, EventArgs e) {
158:             Communication.SetBaseMv(ControlByteBaseMv.Stop);
159:         }
160:
161:         private void BtnBaseMvBck_Click(object sender, EventArgs e) {
162:             Communication.SetBaseMv(ControlByteBaseMv.Backwards);
163:         }
164:
165:         private void BtnBaseMvTurnL_Click(object sender, EventArgs e) {
166:             Communication.SetBaseMv(ControlByteBaseMv.TurnLeft);
167:         }
168:
169:         private void BtnBaseMvTurnR_Click(object sender, EventArgs e) {
170:             Communication.SetBaseMv(ControlByteBaseMv.TurnRight);
171:         }
172:
173:         private void BtnBaseMvRotateL_Click(object sender, EventArgs e) {
174:             Communication.SetBaseMv(ControlByteBaseMv.RotateLeft);
175:         }
176:
177:         private void BtnBaseMvRotateR_Click(object sender, EventArgs e) {
178:             Communication.SetBaseMv(ControlByteBaseMv.RotateRight);
179:         }
180:
181:         /// <summary>
182:         /// Límite mínimo de movimiento del ratón para que se detecte movimiento
183:         /// </summary>
184:         const int thr = 10;
185:
186:         /// <summary>
187:         /// Posición anterior del cursor
188:         /// </summary>
189:         Point PrevMouse;
190:
191:         bool InteractiveMode {
192:             get {
193:                 return _InteractiveMode;
194:             }
195:             set {
```

```
196:                 if(value) BtnInteractive.ForeColor = Color.Green;
197:                 else BtnInteractive.ForeColor = Color.Red;
198:                 _InteractiveMode = value;
199:             }
200:         }
201:         bool _InteractiveMode = false;
202:
203:         void EventsTimer_Tick(object sender, EventArgs e) {
204:             if(PrevMouse == null) PrevMouse = MousePosition;
205:             if(InteractiveMode) {
206:                 if(MousePosition.Y < PrevMouse.Y - thr) BaseMvFwd();
207:                 if(MousePosition.Y > PrevMouse.Y + thr) BaseMvBck();
208:                 if(MousePosition.X < PrevMouse.X - thr) BaseMvLeft();
209:                 if(MousePosition.X > PrevMouse.X + thr) BaseMvRight();
210:             }
211:             PrevMouse = new Point(MousePosition.X, MousePosition.Y);
212:         }
213:
214:         void BaseMvFwd() {
215:             if(BaseMvCurrentStatus == ControlByteBaseMv.Backwards) BtnBaseMvStop.PerformClick();
216:             else if(BaseMvCurrentStatus == ControlByteBaseMv.RotateLeft) BtnBaseMvTurnL.PerformClick()
;
217:             else if(BaseMvCurrentStatus == ControlByteBaseMv.RotateRight) BtnBaseMvTurnR.PerformClick(
);
218:             else BtnBaseMvFwd.PerformClick();
219:         }
220:
221:         void BaseMvBck() {
222:             if(BaseMvCurrentStatus == ControlByteBaseMv.Forward) BtnBaseMvStop.PerformClick();
223:             else BtnBaseMvBck.PerformClick();
224:         }
225:
226:         void BaseMvLeft() {
227:             if(BaseMvCurrentStatus == ControlByteBaseMv.TurnRight) BtnBaseMvFwd.PerformClick();
228:             else if(BaseMvCurrentStatus == ControlByteBaseMv.Forward) BtnBaseMvTurnL.PerformClick();
229:             else if(BaseMvCurrentStatus == ControlByteBaseMv.RotateRight) BtnBaseMvStop.PerformClick(
;
230:             else BtnBaseMvRotateL.PerformClick();
231:         }
```



```
232:
233:     void BaseMvRight() {
234:         if(BaseMvCurrentStatus == ControlByteBaseMv.TurnLeft) BtnBaseMvFwd.PerformClick();
235:         else if(BaseMvCurrentStatus == ControlByteBaseMv.Forward) BtnBaseMvTurnR.PerformClick();
236:         else if(BaseMvCurrentStatus == ControlByteBaseMv.RotateLeft) BtnBaseMvStop.PerformClick();
237:         else BtnBaseMvRotateR.PerformClick();
238:     }
239:
240:     private void BtnInteractive_Click(object sender, EventArgs e) {
241:         // switch interactive/normal mode
242:         InteractiveMode = !InteractiveMode;
243:     }
244:
245:     private void FrmMainWindow_KeyDown(object sender, KeyEventArgs e) {
246:         switch(e.KeyCode) {
247:             case Keys.Escape:
248:                 BtnBaseMvStop.PerformClick();
249:                 break;
250:             case Keys.F5:
251:                 InteractiveMode = !InteractiveMode;
252:                 break;
253:             case Keys.L:
254:             case Keys.Space:
255:                 if(LightFrontLeftON && LightFrontRightON) BtnLightFrontOff.PerformClick();
256:                 else BtnLightFrontOn.PerformClick();
257:                 break;
258:             case Keys.W:
259:                 BaseMvFwd();
260:                 break;
261:             case Keys.S:
262:                 BaseMvBck();
263:                 break;
264:             case Keys.A:
265:                 BaseMvLeft();
266:                 break;
267:             case Keys.D:
268:                 BaseMvRight();
269:                 break;
270:         }
```

```
271:         }  
272:     }  
273: }
```

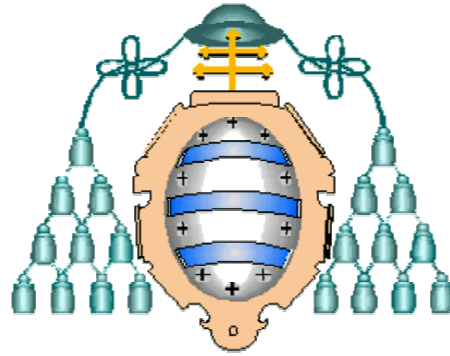
```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace PCBot_PCApp {
7:     public class Global {
8:         public static FrmMainWindow MainWindow;
9:
10:        public static void Log(string p) {
11:            string LogText = p;
12:            if(p[0] != '[') LogText = string.Format("[{0:00}:{1:00}:{2:00}.{3:000}] {4}", DateTime.Now.
.Hour, DateTime.Now.Minute, DateTime.Now.Second, DateTime.Now.Millisecond, p);
13:            Console.WriteLine(LogText);
14:            //if(MainWindow != null) MainWindow.TxtLog.AppendText(p);
15:        }
16:    }
17: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Windows.Forms;
5:
6: namespace PCBot_PCAApp {
7:     static class Program {
8:         /// <summary>
9:         /// El punto de entrada de la aplicación
10:        /// </summary>
11:        [STAThread]
12:        static void Main() {
13:            Application.EnableVisualStyles();
14:            Application.SetCompatibleTextRenderingDefault(false);
15:            Application.Run(new FrmMainWindow());
16:        }
17:    }
18: }
```

```
1: using System;
2: using System.IO.Ports;
3: using BclSP = System.IO.Ports.SerialPort;
4:
5: namespace PCBot_PCAApp {
6:     /// <summary>
7:     /// Representa un Puerto Serie
8:     /// </summary>
9:     public class SerialPort:BclSP {
10:         public SerialPort(string portName, int baudRate, Parity parity, int dataBits, StopBits stopBits)
11:             : base(portName, baudRate, parity, dataBits, stopBits) {
12:         }
13:
14:         public new byte ReadByte() {
15:             return (byte)base.ReadByte();
16:         }
17:
18:         public UInt16 ReadUInt16() {
19:             byte Byte1, Byte0;
20:             Byte1 = (byte)ReadByte();
21:             Byte0 = (byte)ReadByte();
22:             return (UInt16)((Byte1 << 8) + Byte0);
23:         }
24:
25:         public UInt32 ReadUInt32() {
26:             byte Byte3, Byte2, Byte1, Byte0;
27:             Byte3 = (byte)ReadByte();
28:             Byte2 = (byte)ReadByte();
29:             Byte1 = (byte)ReadByte();
30:             Byte0 = (byte)ReadByte();
31:             return (UInt32)((((UInt32)Byte3 << 24) + ((UInt32)Byte2 << 16) + ((UInt32)Byte1 << 8) + (UI
nt32)Byte0));
32:         }
33:
34:         public void WriteByte(byte Byte) {
35:             Write(new byte[] { Byte }, 0, 1);
36:         }
37:
38:         public void WriteUInt16(UInt16 Value) {
```

```
39:         Write(Value.GetBytes(), 0, 2);
40:     }
41:
42:     public void WriteUInt32(UInt32 Value) {
43:         Write(Value.GetBytes(), 0, 4);
44:     }
45: }
46: }
```

```
1: using System;
2: using System.Collections.Generic;
3: using System.Linq;
4: using System.Text;
5:
6: namespace PCBot_PCAp {
7:     public static class Settings {
8:         public static string SpName = "/dev/ttyUSB0";
9:         public static int BaudRate = 9600;
10:        public static byte RobotID = 13;
11:    }
12: }
```

Universidad de Oviedo

Escuela Universitaria de Ingeniería Técnica Industrial

Ingeniería Técnica Industrial, Especialidad Electrónica
Industrial

**Plataforma modular educativa para
Aprendizaje de microcontroladores**

Documento Planos

Departamento de Ingeniería Eléctrica,
Electrónica, de Computadores y Sistemas
Autor: Adrián Bulnes Parra
Tutor: Miguel Ángel José Prieto

Gijón, 30 de junio de 2010

Fdo.: Adrián Bulnes Parra

1. Notas

Debido a que el objetivo de los circuitos diseñados para ser construidos en placa perforada es que sea el propio usuario o cliente quien los construya, en lugar de incluir los despieces y mecanizados habituales con las vistas superior, inferior y distribución de componentes, se ha optado por incluir en su lugar las vistas del diseño de la placa completa a color, ya que son la representación más adecuada para que los usuarios puedan utilizarlas como referencia durante la construcción del circuito. Este tipo de representación es la utilizada en las *versiones imprimibles* disponibles en el CD (ver Anexo XXVIII) y se han utilizado sin ningún problema en los Talleres de Robótica (ver Documento Memoria, apartado N°7).

En los circuitos que, en lugar de haber sido diseñados para ser construidos solamente con cables o utilizando placas perforadas, se han construido en placas de circuito impreso, se incluye también el fotolito, es decir, la imagen que se necesita para fabricar el PCB utilizando una insoladora.

Por lo tanto, las imágenes incluidas en los planos pueden ser de los siguientes tipos:

- Esquema electrónico: representación gráfica de un circuito electrónico.
- Diagrama de cableado:
 - En circuitos construidos sobre el propio cable: representación gráfica del circuito real construido.
 - En circuitos construidos en placa perforada o PCB: representación del cableado que debe fabricarse y utilizarse para conectar Placas Principales y Módulos entre sí.

El diagrama de cableado se diferencia del esquema electrónico ya que este último es la representación formal del circuito y se utiliza para comprender y analizar su funcionamiento, mientras que el diagrama de cableado representa la estructura real del circuito a construir y se utiliza durante la construcción del mismo.

- Vista a color de placa y componentes: representación gráfica del resultado final de una placa perforada o una placa de circuito impreso que incluye todas las capas en una sola imagen. Habitualmente los colores son:
 - Azul: pistas de cobre trazadas en la cara inferior de la placa. En caso de que sea una placa perforada, las pistas se crearán utilizando puntos de soldadura uniendo las bahías de cobre situadas alrededor de cada agujero.
 - Rojo: pistas de cobre trazadas en la cara superior de la placa. Estas pistas se crearán utilizando trozos de cable con o sin recubrimiento se manera que se cortocircuiten ciertas pistas de la cara inferior a través del cable situado en la parte superior de la placa.
 - Amarillo (fondo negro) o verde (fondo blanco): límites o detalles interiores del cuerpo de los componentes situados en la parte superior de la placa.

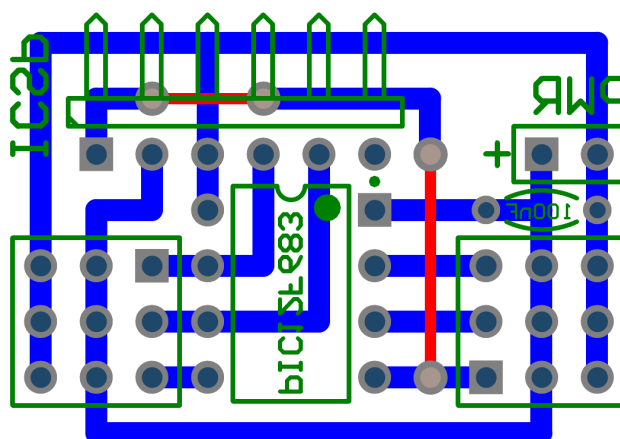
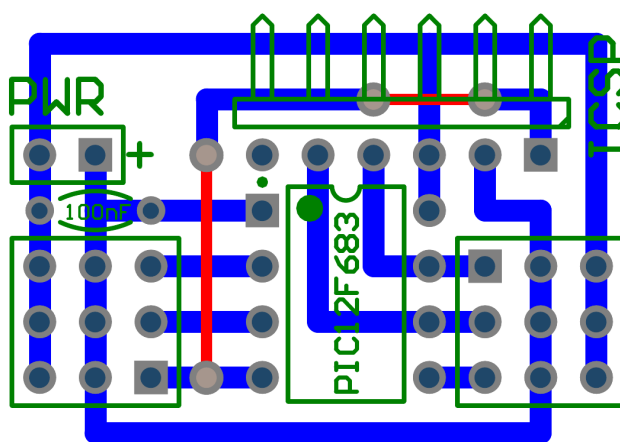
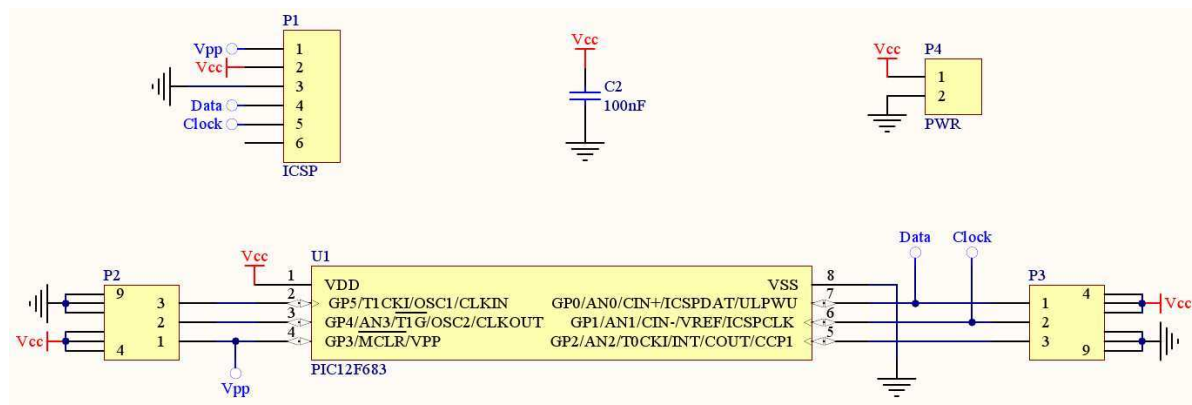
o Tonos grises: agujeros y bahías de cobre situadas alrededor de cada agujero.

Este tipo de esquemas, en caso de estar diseñados para placa perforada, se incluyen en los planos la imagen propiamente dicha y además la misma imagen invertida horizontalmente. Debido a que estas placas están diseñadas para ser construidas por los propios usuarios, ésta es la manera más cómoda para ellos de comprender el esquema y así ser capaces de construirlo. La imagen no invertida normalmente se utilizará para colocar los componentes y los cables que forman las pistas superiores, mientras que la imagen invertida se utilizará para construir las pistas inferiores mediante estaño.

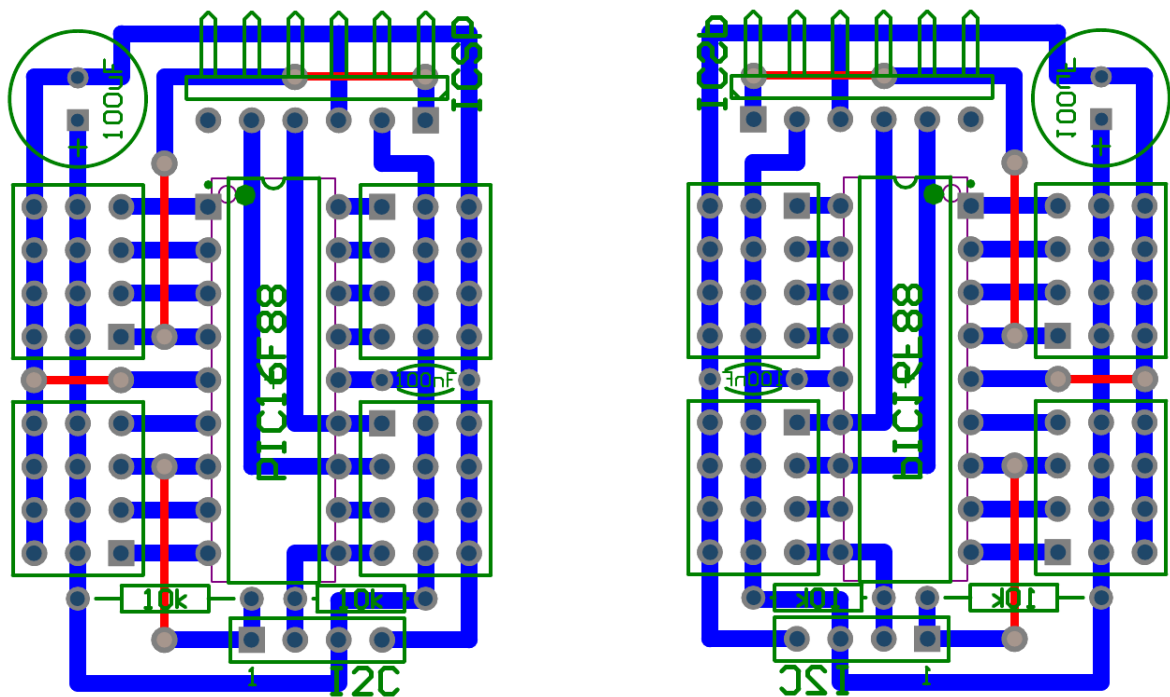
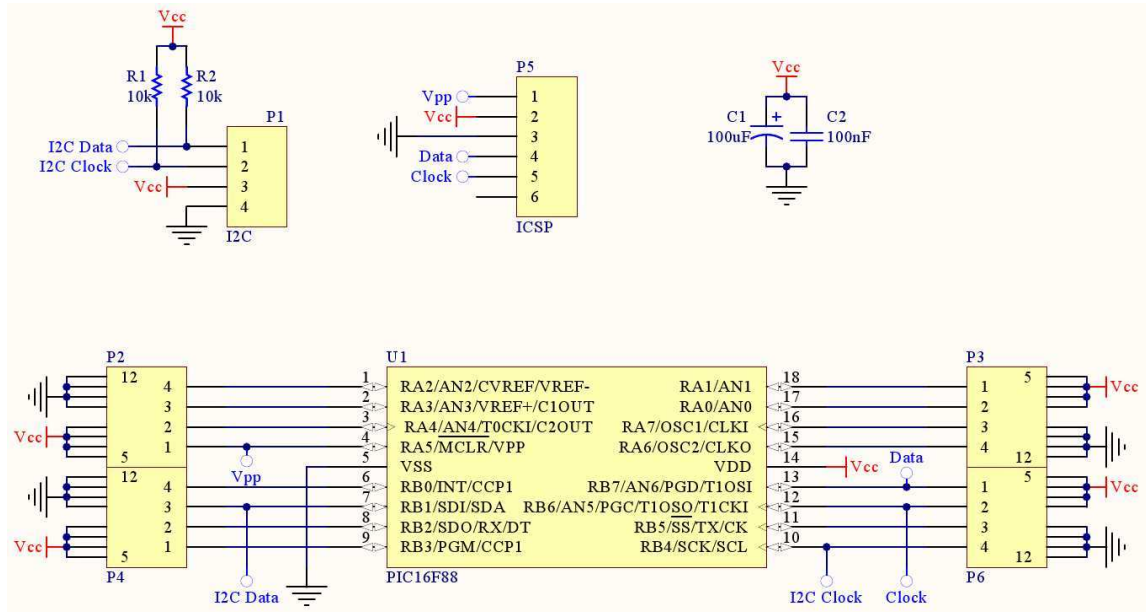
- Fotolitos: representación del cobre situado en la cara inferior de una placa de circuito impreso, utilizado para la construcción de la propia placa.

2. Lista de planos

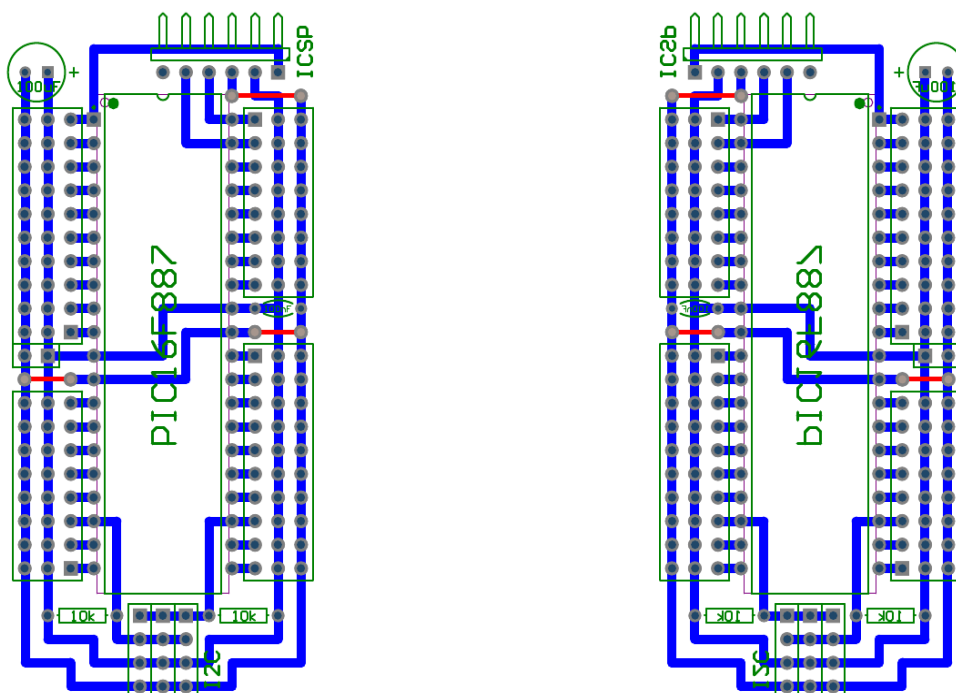
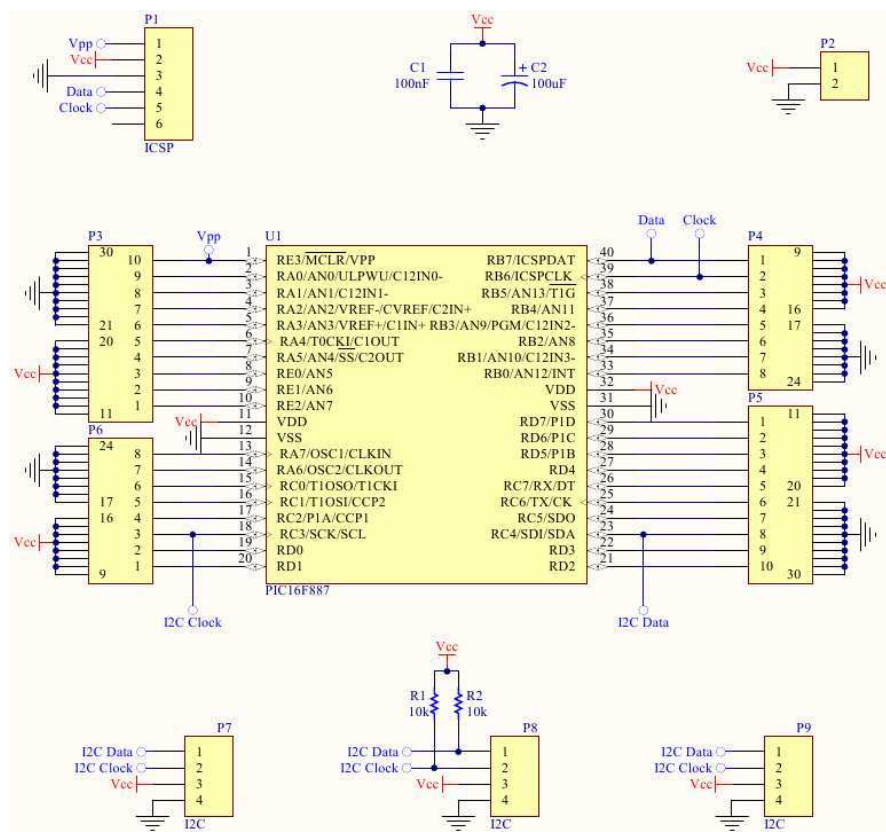
1. Placa Principal: MBP8
2. Placa Principal: MBP14
3. Placa Principal: MBP18
4. Placa Principal: MBP40
5. Módulo: LTIND-A
6. Módulo: SISW-SPST
7. Módulo: LTIL-A
8. Módulo: SIBW-1Y
9. Módulo: AO-SPK
10. Módulo: SIDST-GP2
11. Módulo: LT7S-HN
 - 11.1. Esquema para uno o varios segmentos y *displays*
 - 11.2. Esquemas para 7 segmentos y 6 *displays*
12. Módulo: CMIR-RC
13. Módulo: MC2A
 - 13.1. Esquema electrónico y PCB
 - 13.2. Diagrama de cableado y fotolito a escala real
14. Módulo: MC2B
15. Módulo: PWR-DD5
16. Módulo: PWR-AD5
17. Módulo: CMSP-MAX



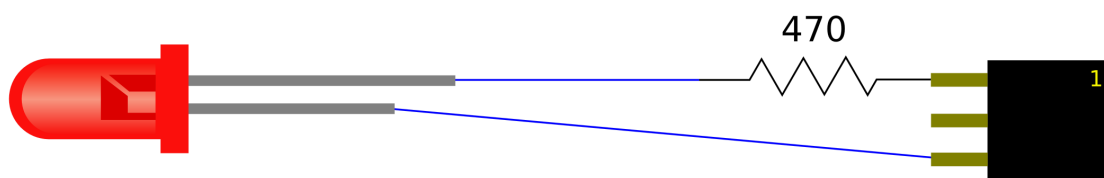
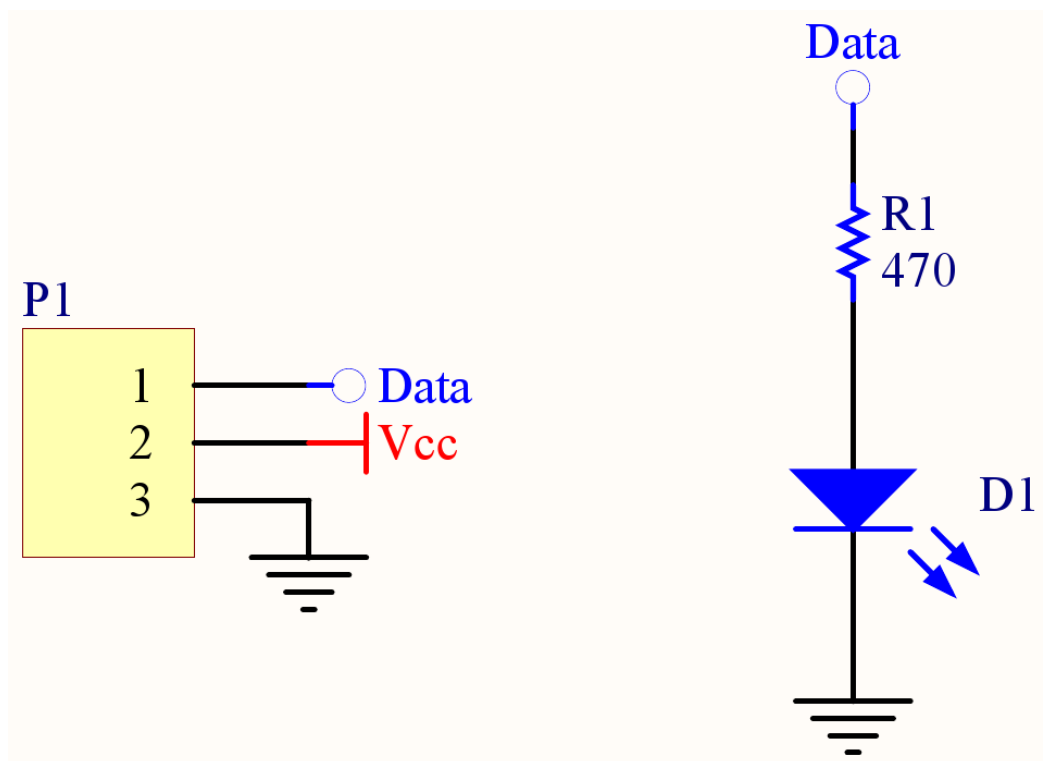
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON
Diseñado	13/10/2009	Adrián Bulnes		
Dibujado	15/10/2009	Adrián Bulnes		
Comprobado	29/06/2010	Adrián Bulnes		
Escala n/a	Placa Principal: MBP8			Plano núm: 1
	Esquema electrónico y placa perforada			Hoja 1 de 1



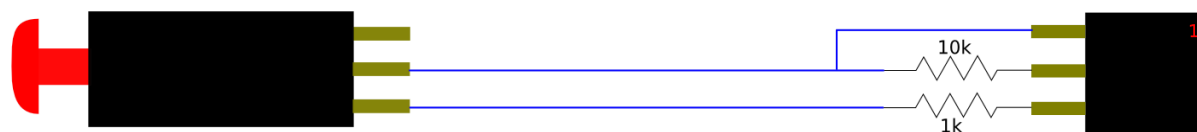
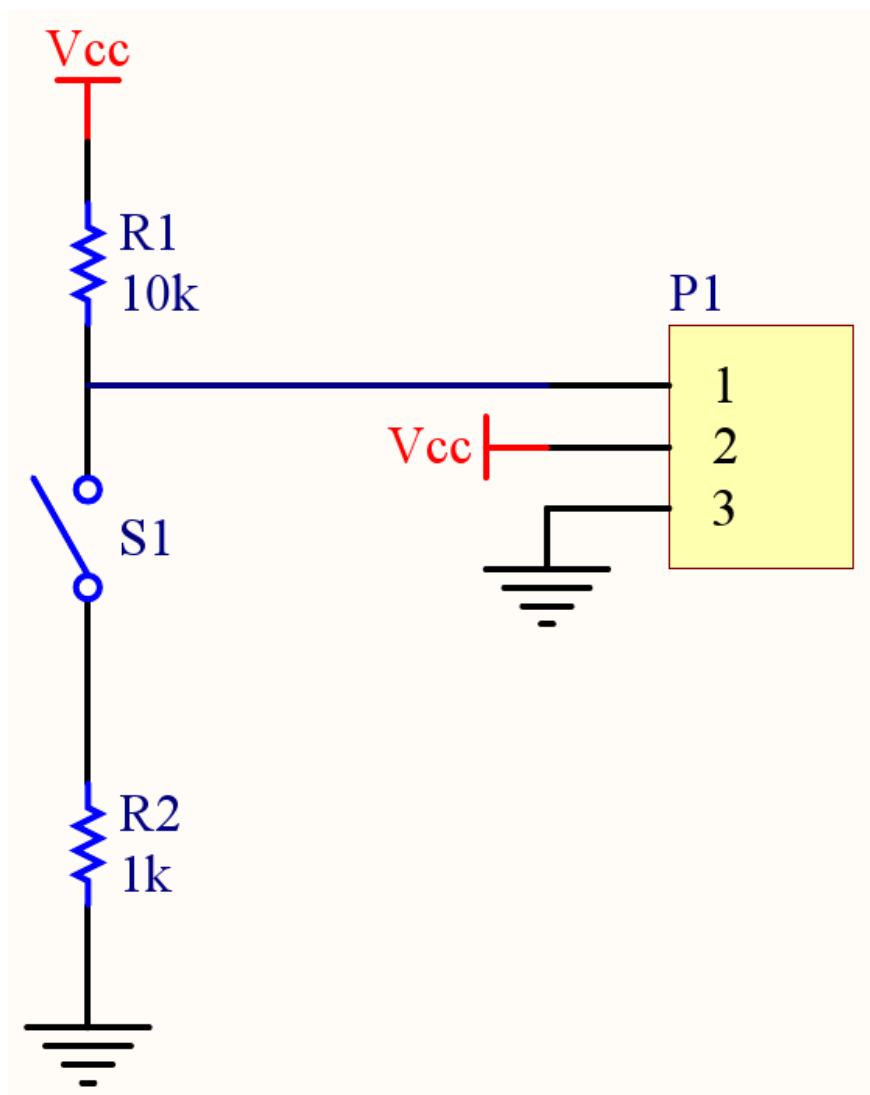
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON
Diseñado	21/09/2009	Adrián Bulnes		
Dibujado	23/09/2009	Adrián Bulnes		
Comprobado	29/06/2010	Adrián Bulnes		
Escala n/a	Placa Principal: MBP18 Esquema electrónico y placa perforada			Plano núm: 3 Hoja 1 de 1



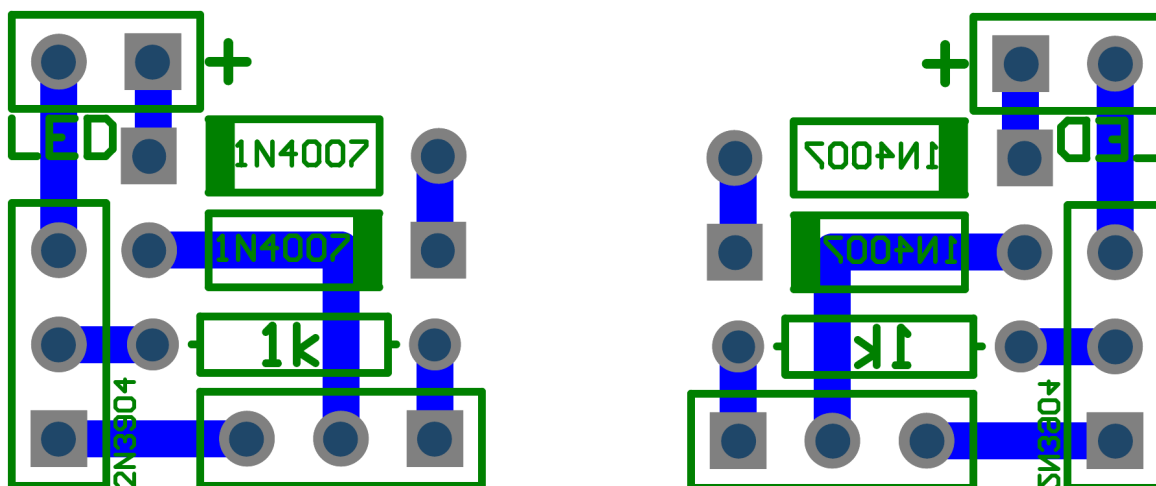
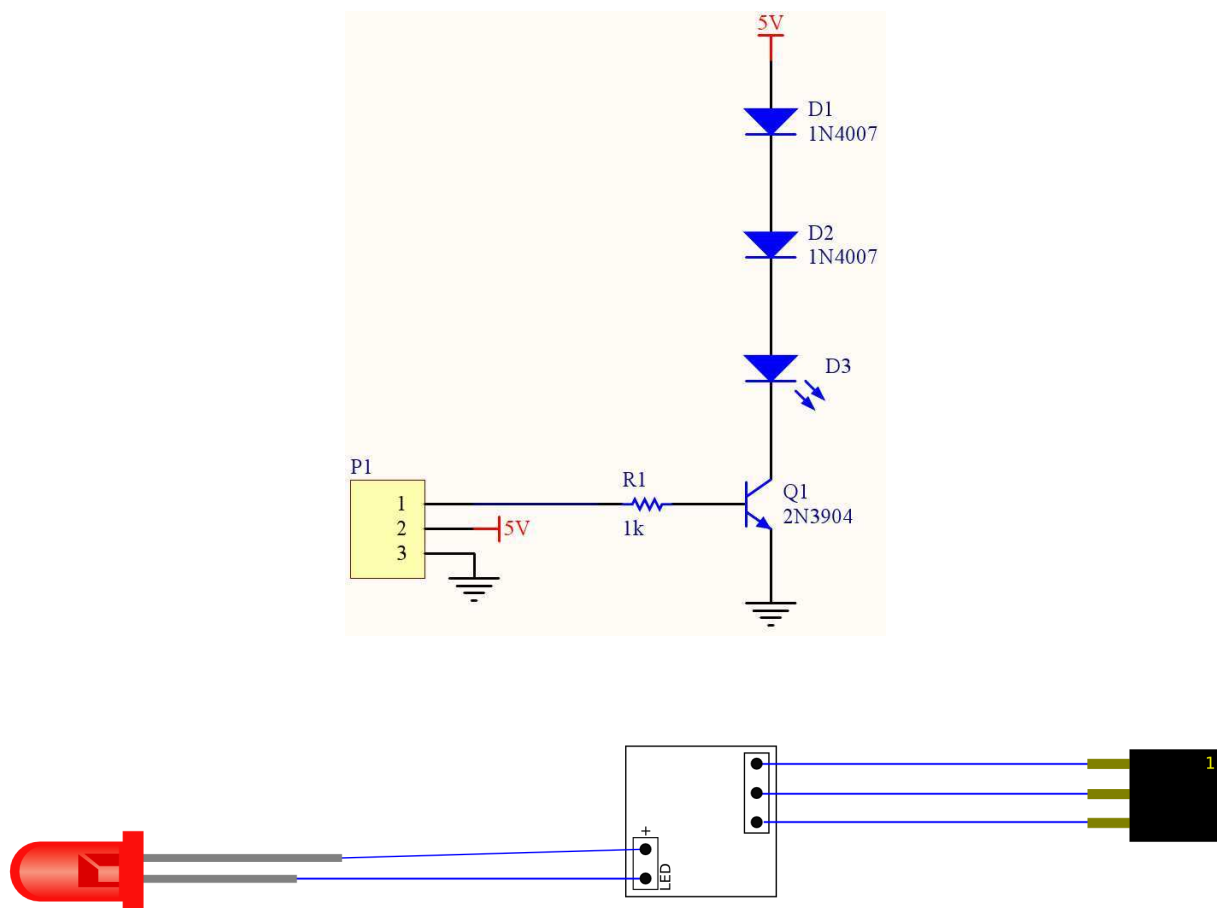
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON
Diseñado	13/11/2009	Adrián Bulnes		
Dibujado	15/11/2009	Adrián Bulnes		
Comprobado	29/06/2010	Adrián Bulnes		
Escala n/a	Placa Principal: MBP40			Plano núm: 4
	Esquema electrónico y placa perforada			Hoja 1 de 1



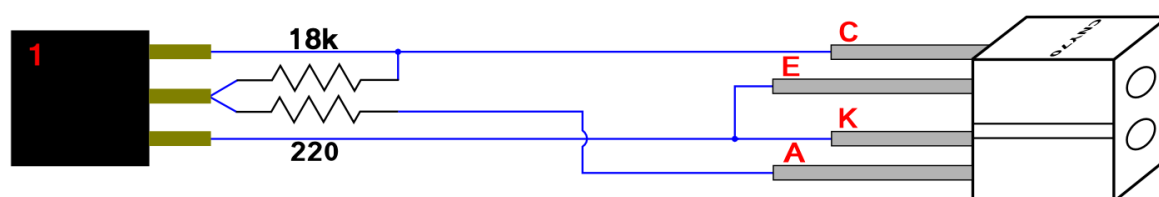
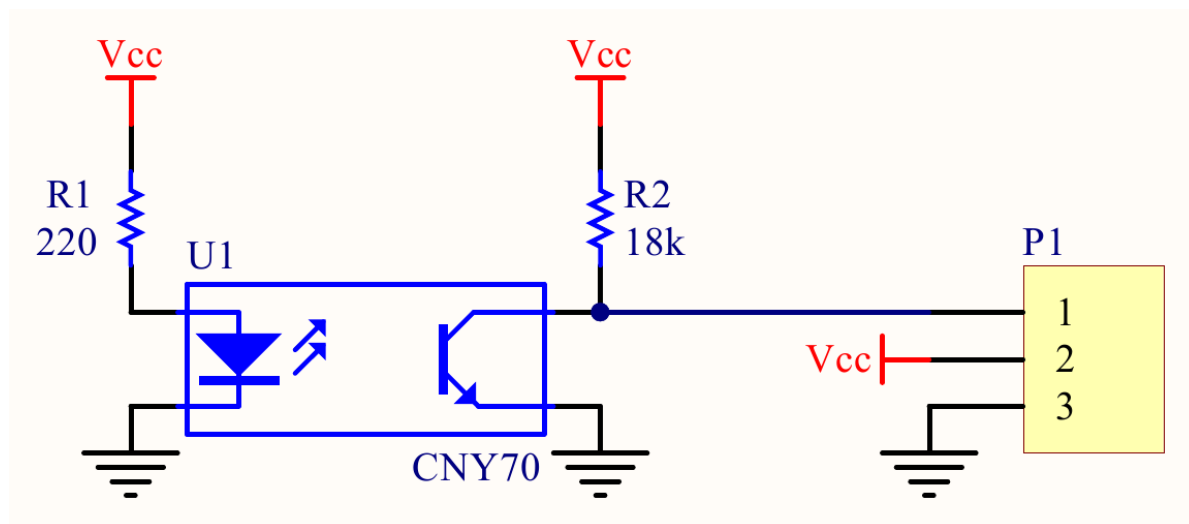
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	24/09/2009	Adrián Bulnes			
Dibujado	26/09/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: LTIND-A				Plano núm: 5
	Esquema electrónico y diagrama de cableado				Hoja 1 de 1



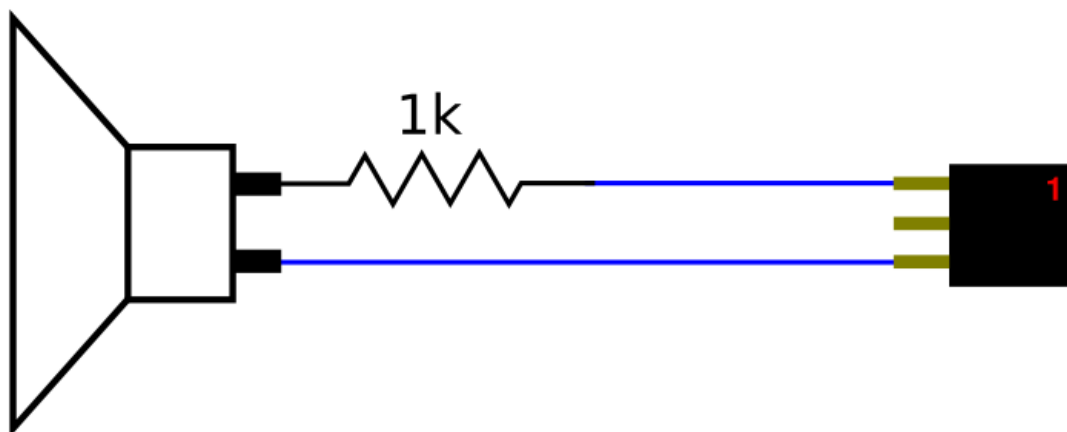
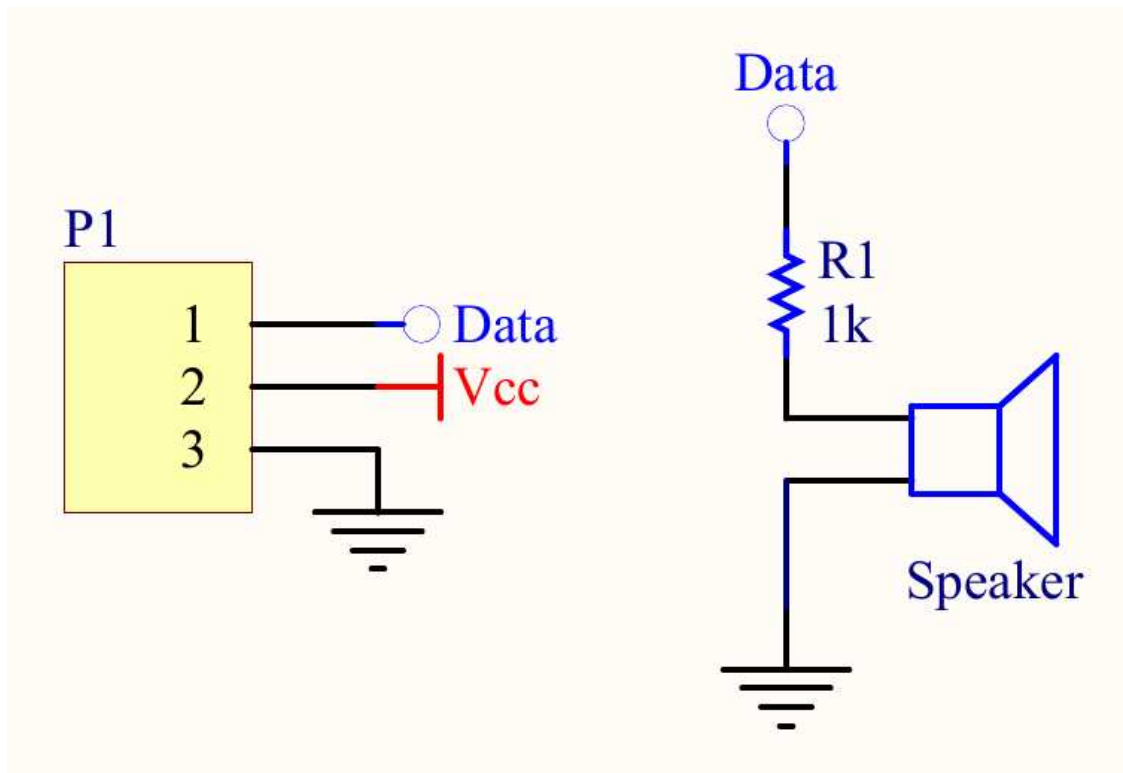
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	28/09/2009	Adrián Bulnes			
Dibujado	30/09/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: SISW-SPST				Plano núm: 6
	Esquema electrónico y diagrama de cableado				Hoja 1 de 1



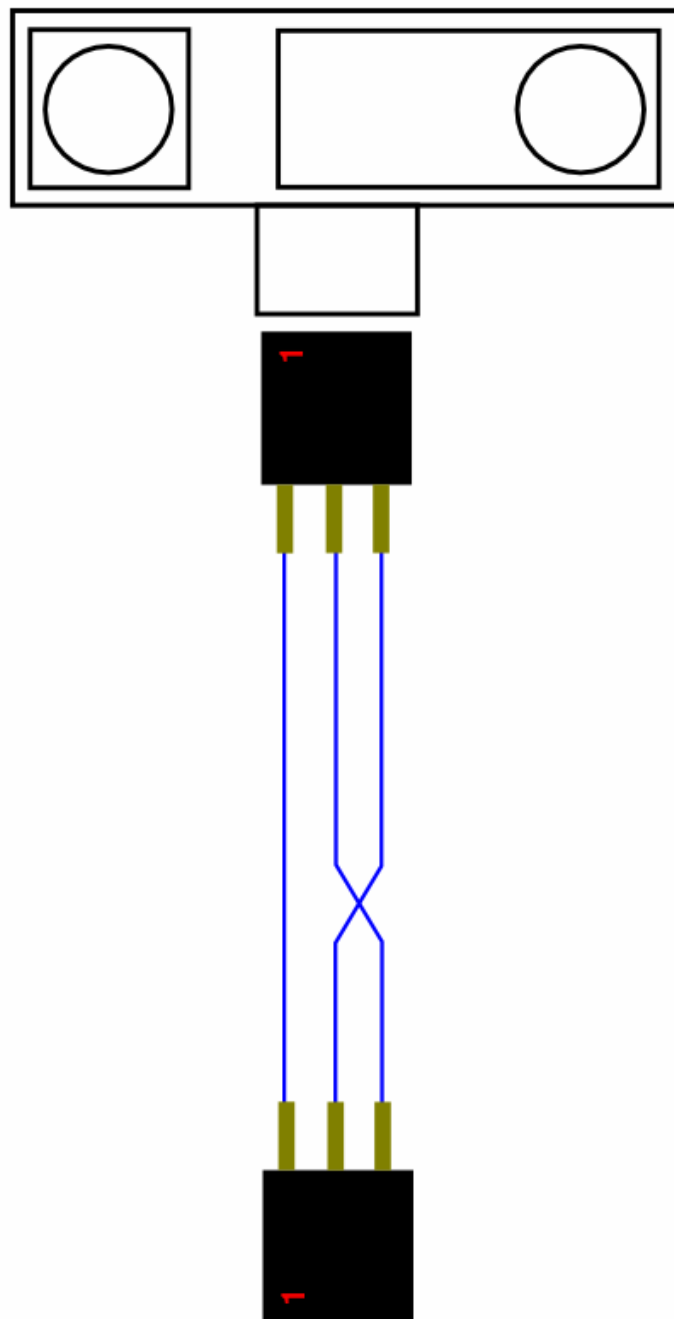
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	03/10/2009	Adrián Bulnes			
Dibujado	05/10/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: LTIL-A				Plano núm: 7
	Esquema electr., diag. de cableado y p. perforada				Hoja 1 de 1



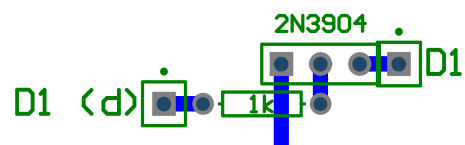
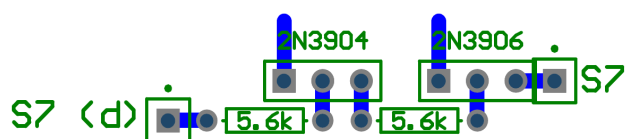
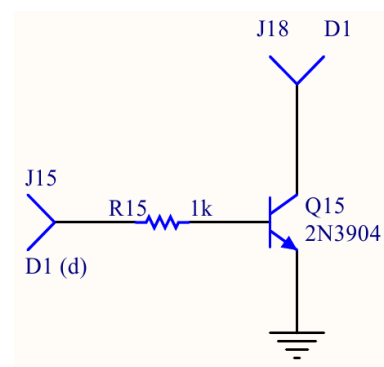
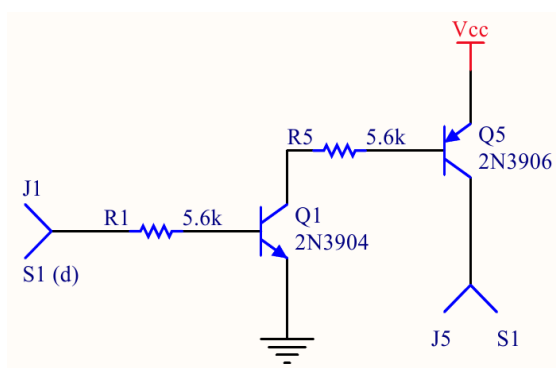
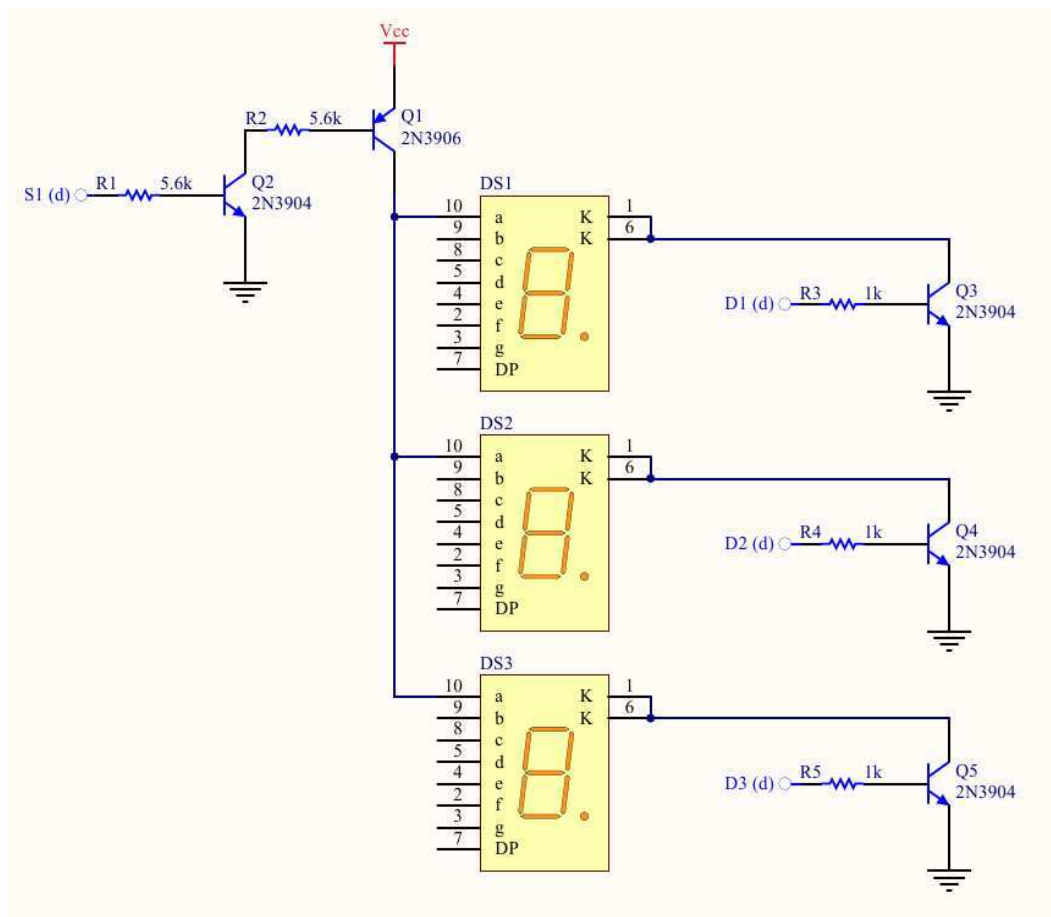
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	16/12/2009	Adrián Bulnes			
Dibujado	18/12/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: SIBW-1Y				Plano núm: 8
	Esquema electrónico y diagrama de cableado				Hoja 1 de 1



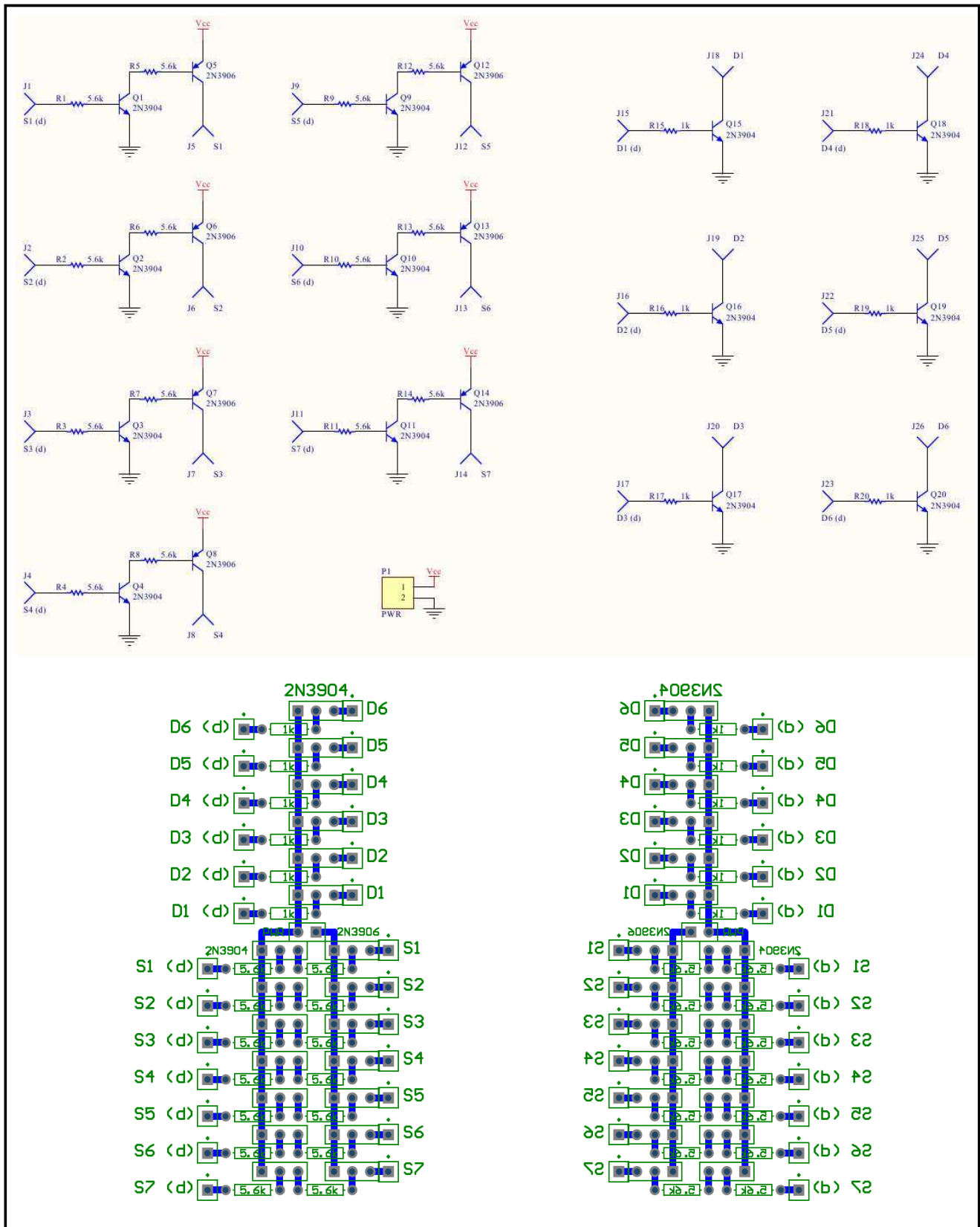
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	12/01/2010	Adrián Bulnes			
Dibujado	15/01/2010	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: AO-SPK				Plano núm: 9
	Esquema electrónico y diagrama de cableado				Hoja 1 de 1



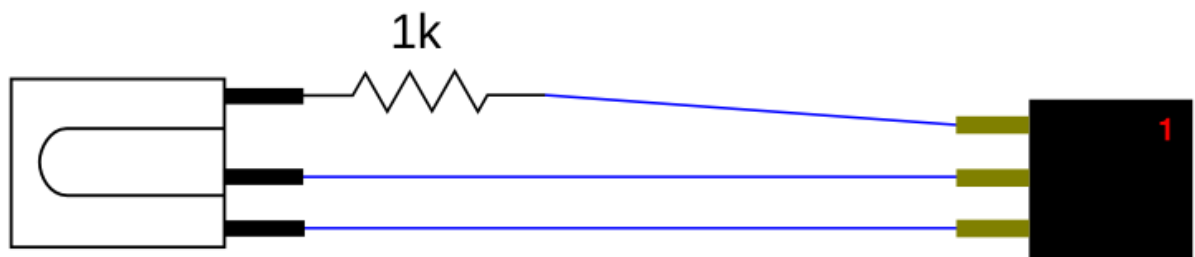
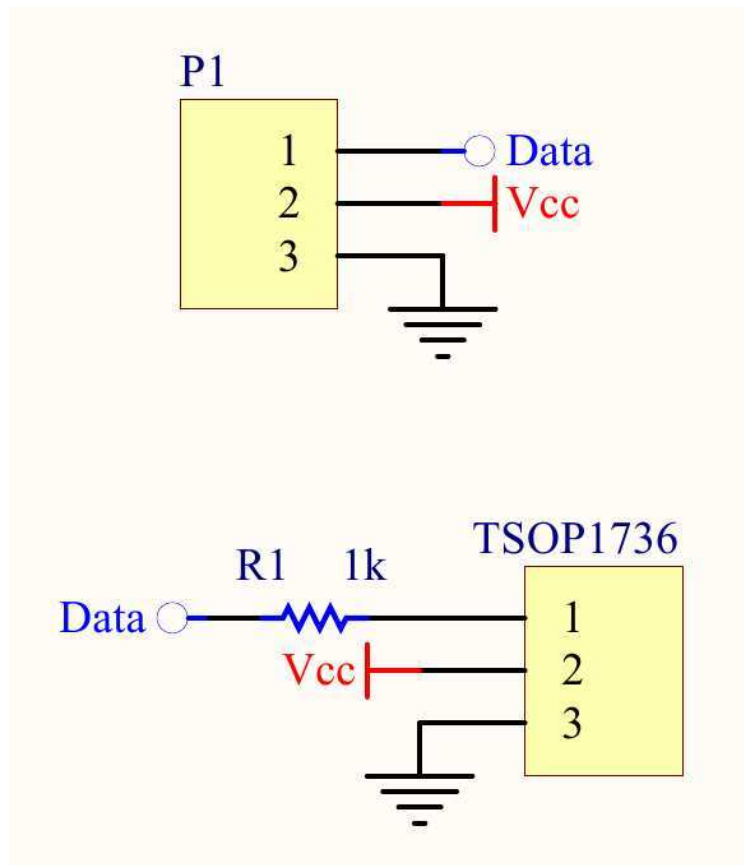
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	27/03/2010	Adrián Bulnes			
Dibujado	28/03/2010	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: SIDST-GP2				Plano núm: 10
	Diagrama de cableado				Hoja 1 de 1



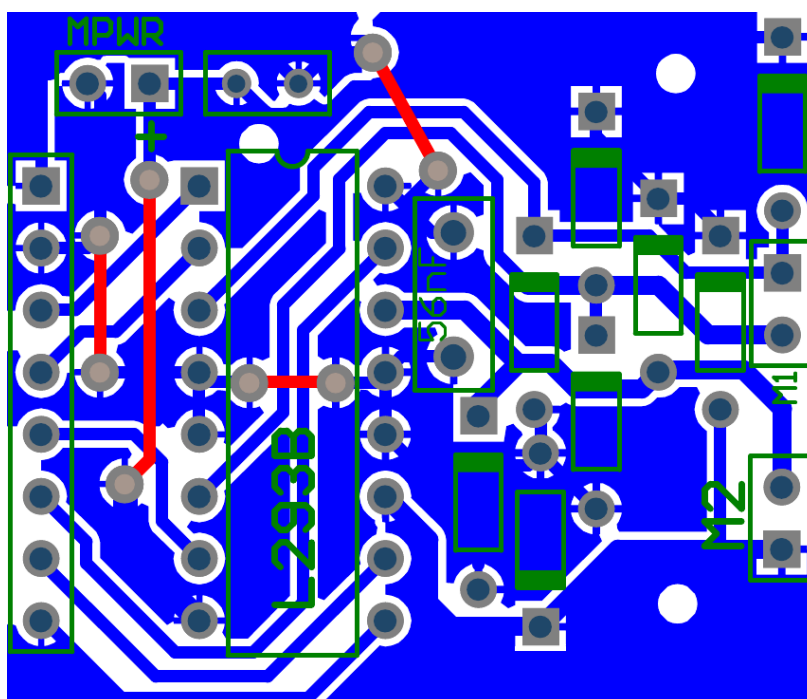
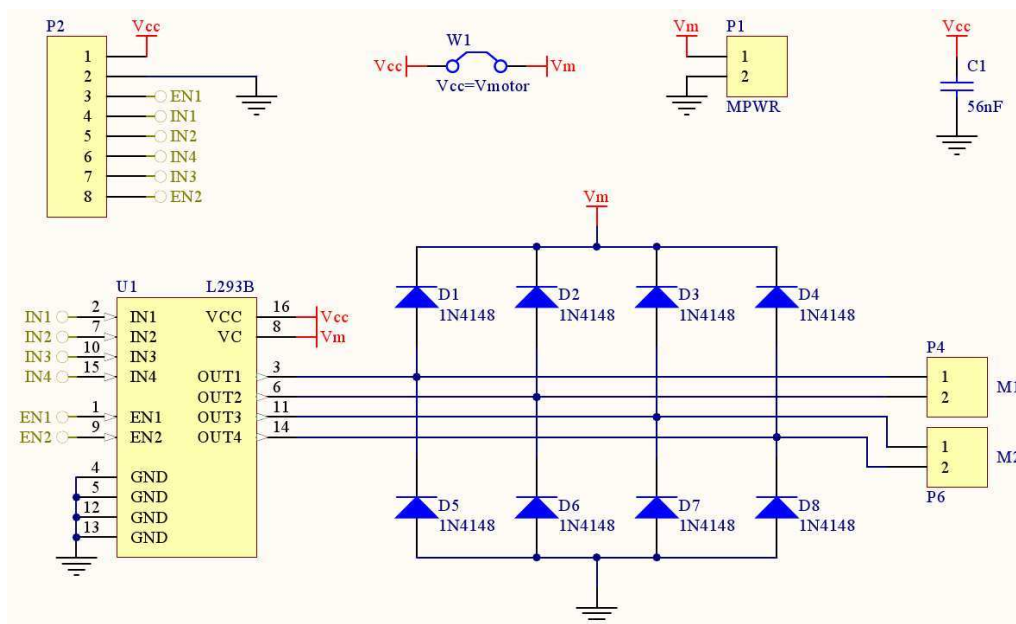
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	06/04/2010	Adrián Bulnes			
Dibujado	08/03/2010	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: LT7S-HN				Plano núm: 11
	Esquemas para uno o varios segmentos y displays				Hoja 1 de 2



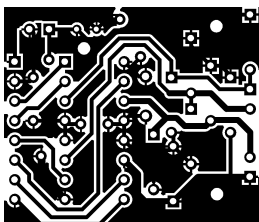
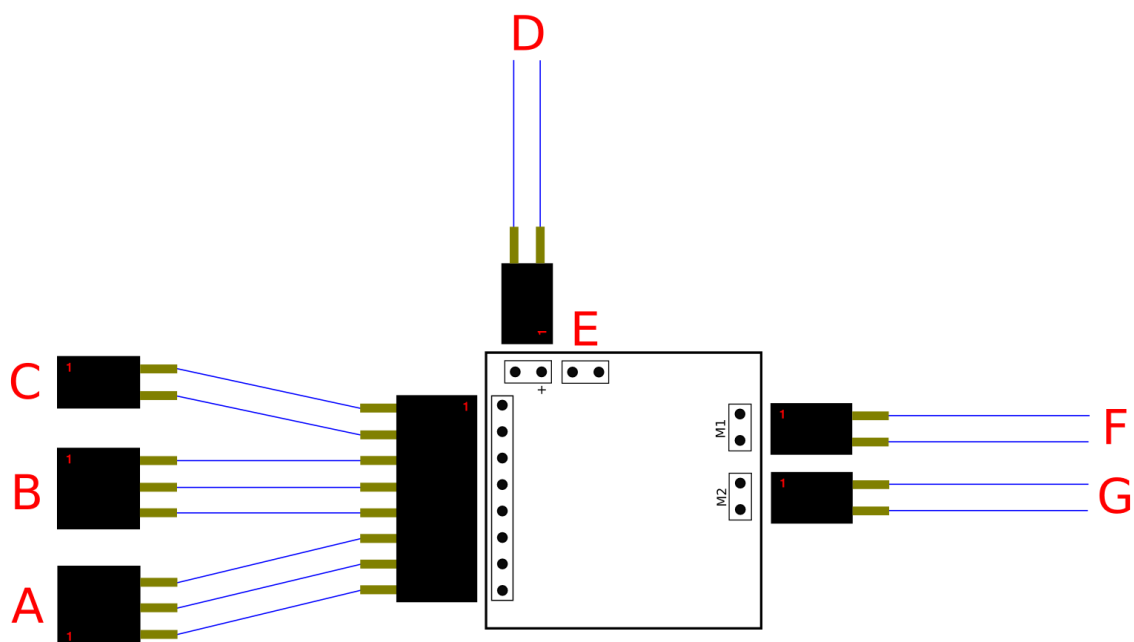
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	06/04/2010	Adrián Bulnes			
Dibujado	08/03/2010	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: LT7S-HN				Plano núm: 11
	Esq. elect. y placa perf. para 7 segmentos y 6 displays				Hoja 2 de 2



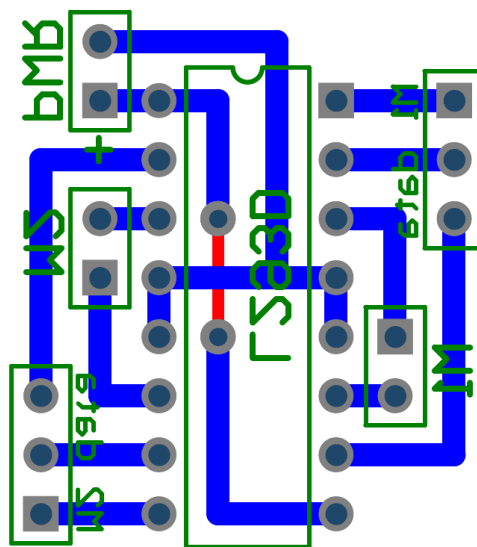
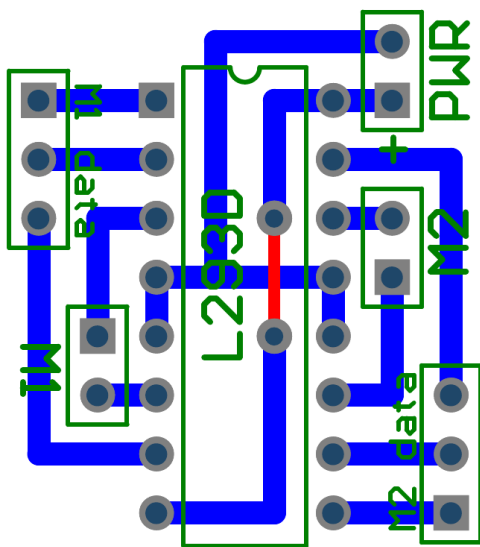
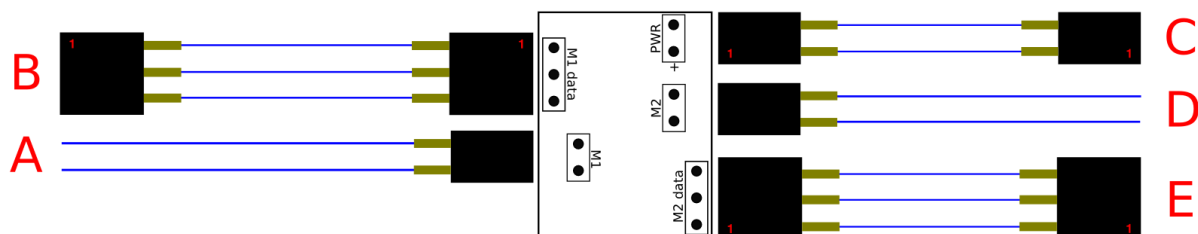
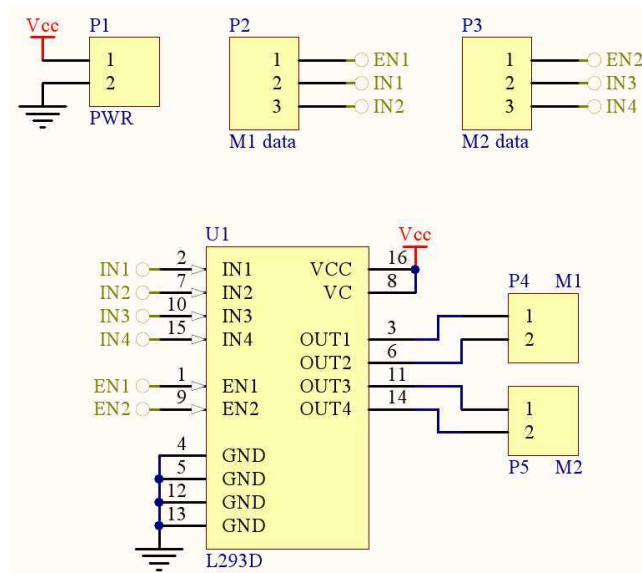
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	03/12/2009	Adrián Bulnes			
Dibujado	05/12/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: CMIR-RC				Plano núm: 12
	Esquema electrónico y diagrama de cableado para TSOPx				Hoja 1 de 1



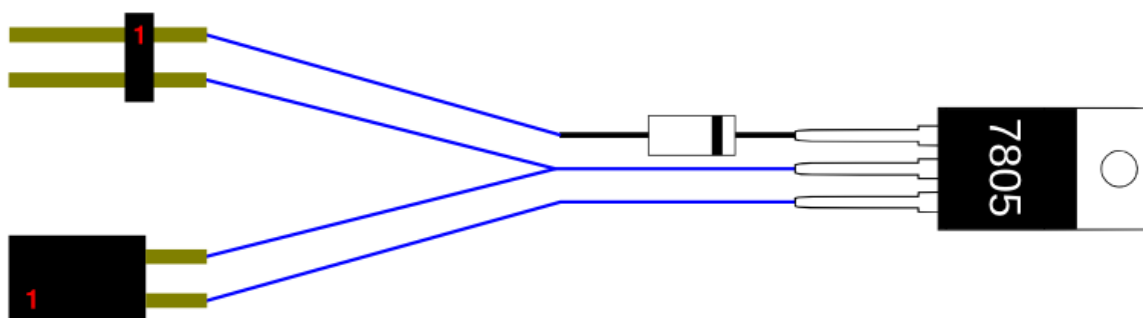
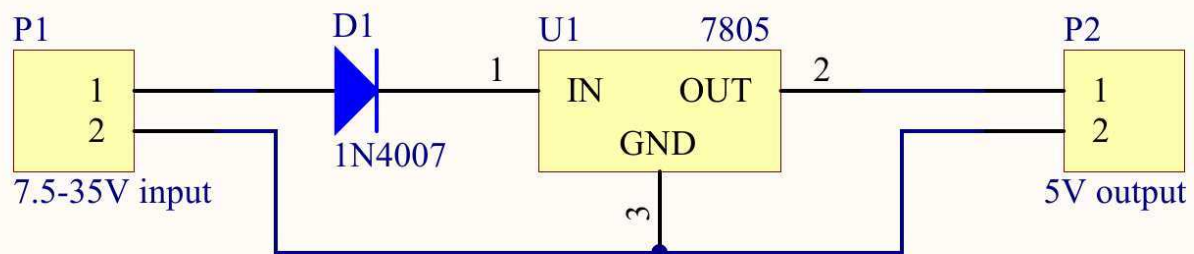
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON
Diseñado	04/10/2009	Adrián Bulnes		
Dibujado	07/10/2009	Adrián Bulnes		
Comprobado	29/06/2010	Adrián Bulnes		
Escala n/a	Módulo: MC2A			Plano núm: 13
	Esquema electrónico y PCB			Hoja 1 de 2



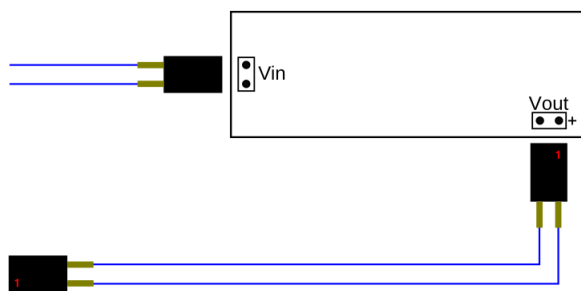
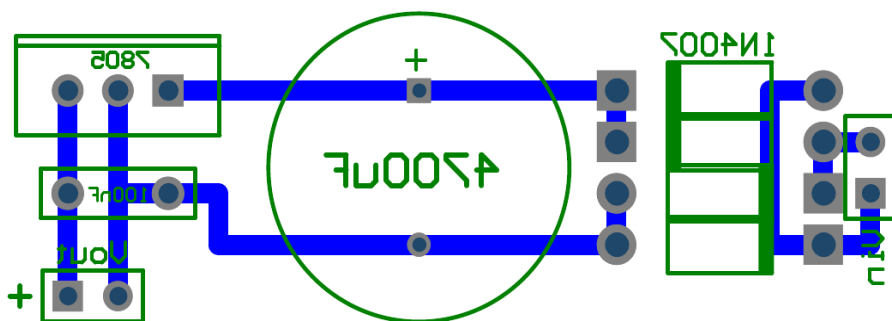
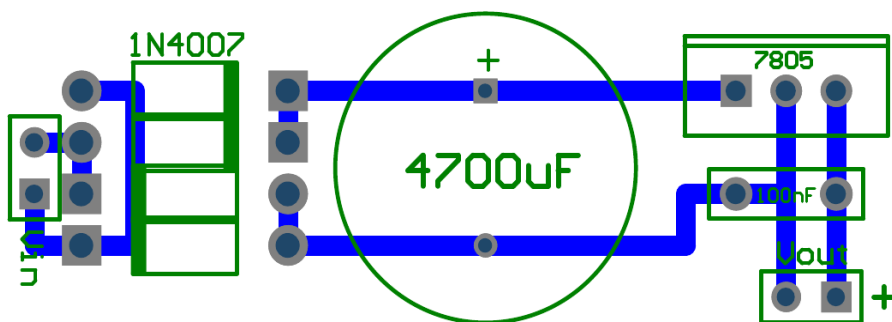
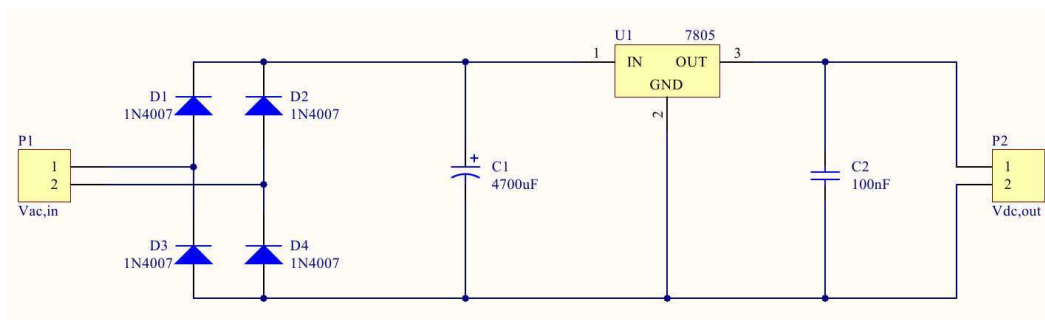
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON
Diseñado	10/10/2009	Adrián Bulnes		
Dibujado	15/10/2009	Adrián Bulnes		
Comprobado	29/06/2010	Adrián Bulnes		
Escala n/a	Módulo: MC2A			Plano núm: 13
	Diagrama de cableado y fotolito a escala real			Hoja 2 de 2



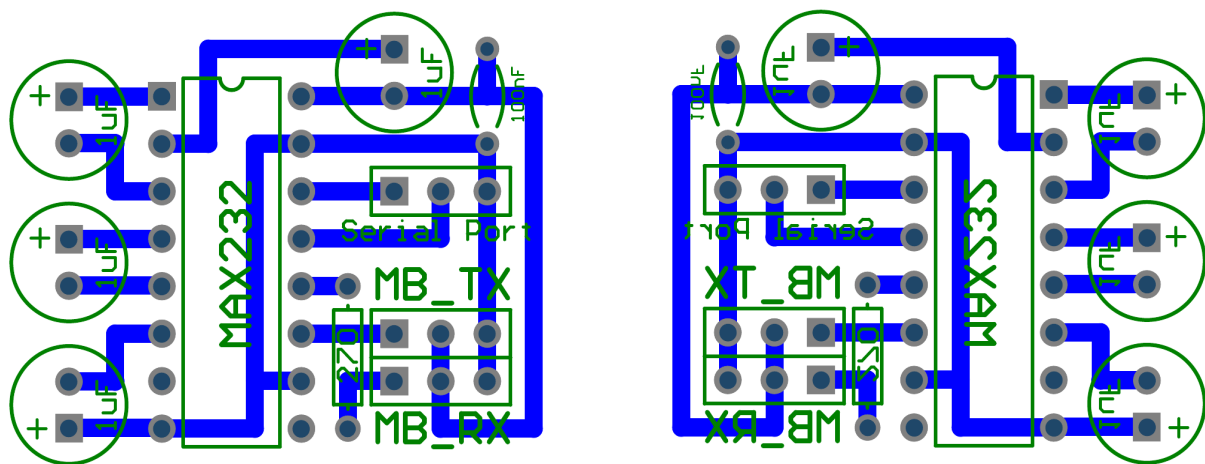
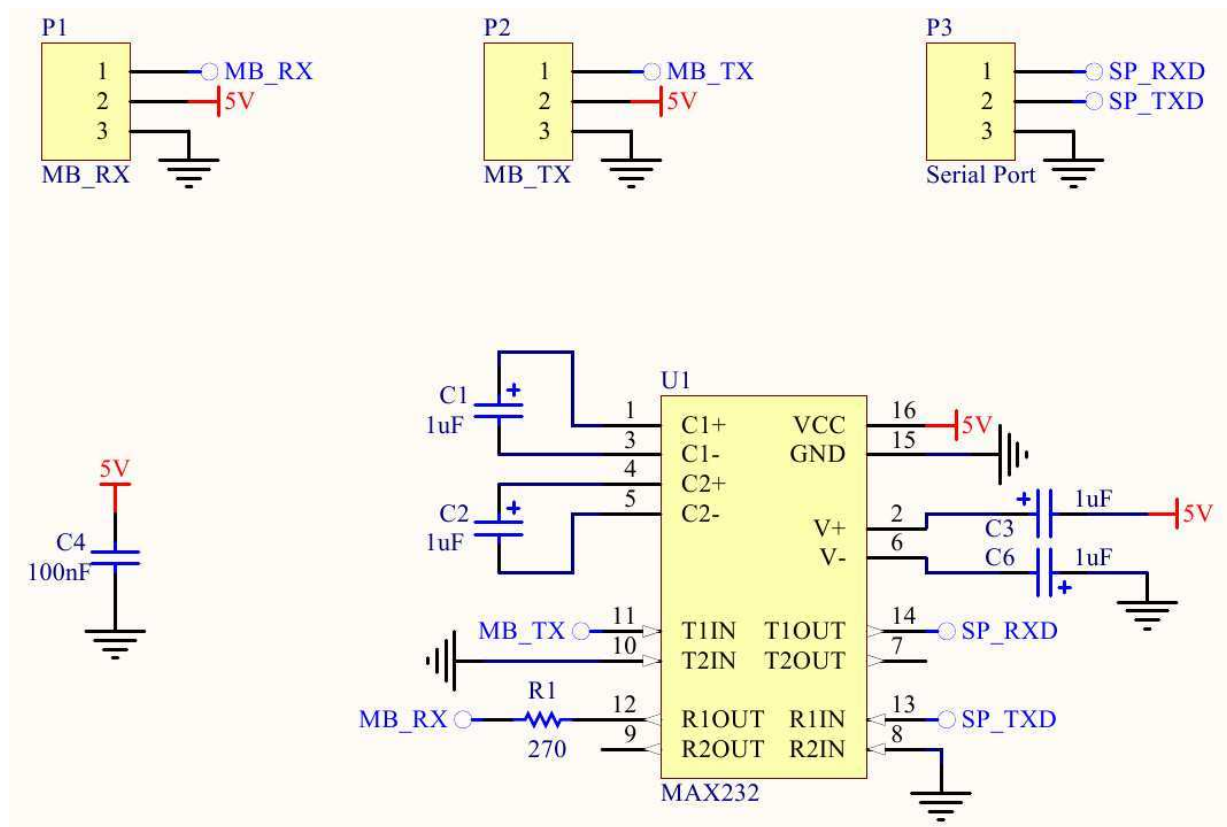
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	30/10/2009	Adrián Bulnes			
Dibujado	1/11/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: MC2B				Plano núm: 14
	Esquema electr., diag. de cableado y placa perforada				Hoja 1 de 1



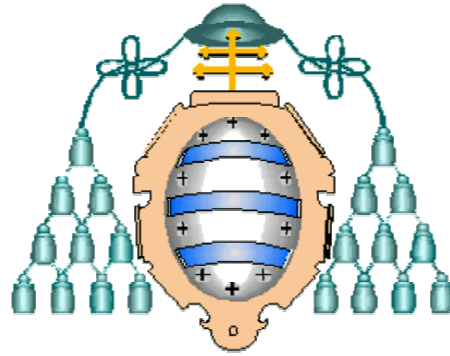
	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	28/10/2009	Adrián Bulnes			
Dibujado	31/10/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: PWR-DD5				Plano núm: 15
	Esquema electrónico y diagrama de cableado				Hoja 1 de 1



	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	15/11/2009	Adrián Bulnes			
Dibujado	17/11/2009	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: PWR-AD5				Plano núm: 16
	Esquema elect., diag. de cableado y placa perforada				Hoja 1 de 1



	Fecha	Nombre	Firma	ESCUELA UNIVERSITARIA DE INGENIERIA TECNICA INDUSTRIAL DE GIJON	
Diseñado	06/05/2010	Adrián Bulnes			
Dibujado	07/05/2010	Adrián Bulnes			
Comprobado	29/06/2010	Adrián Bulnes			
Escala n/a	Módulo: CMSP-MAX				Plano núm: 17
	Esquema electrónico y placa perforada				Hoja 1 de 1



Universidad de Oviedo

Escuela Universitaria de Ingeniería Técnica Industrial

Ingeniería Técnica Industrial, Especialidad Electrónica
Industrial

**Plataforma modular educativa para
Aprendizaje de microcontroladores**

Documento Pliego de Condiciones

Departamento de Ingeniería Eléctrica,
Electrónica, de Computadores y Sistemas
Autor: Adrián Bulnes Parra
Tutor: Miguel Ángel José Prieto

Gijón, 30 de junio de 2010

Fdo.: Adrián Bulnes Parra

1. CONDICIONES GENERALES	3
1.1. NORMATIVA.....	4
Norma UNE 82009-3.....	4
Norma UNE 20-621-85/3-1 ^{er} C.....	4
1.2. CONDICIONES LEGALES.....	5
1.3. CONDICIONES DE CARÁCTER FACULTATIVO.	6
2. ESPECIFICACIONES DE MATERIALES Y EQUIPOS	9
3. ESPECIFICACIONES DE EJECUCIÓN	10
3.1. OBLIGACIONES DE LA PROPIEDAD	10
3.2. ACCIONES DEL CONTRATISTA INSTALADOR	10
3.3. ESPECIFICACIONES TÉCNICAS.....	12
4. PLIEGO DE CONDICIONES ECONÓMICAS	13

1. Condiciones generales

▪ Artículo 1. Objeto del pliego de condiciones

El presente pliego de condiciones tiene por objeto regular el diseño y la construcción de una plataforma electrónica modular educativa para el aprendizaje de microcontroladores, la cual incluye un conjunto de circuitos electrónicos diseñados principalmente (pero no necesariamente) para ser contruidos por los usuarios o estudiantes, un conjunto de herramientas informáticas para facilitar la programación de los microcontroladores, y aplicaciones de ejemplo de uso de esta plataforma, incluyendo la configuración de componentes electrónicos y programas para ordenador y microcontroladores necesarios, pero excluyendo diseños mecánicos. Por lo tanto, la posible comercialización de productos completos utilizando esta plataforma no forma parte de este proyecto.

Por otro lado este documento establece las condiciones técnicas, facultativas, económicas y legales en las que han de basarse los trabajos a realizar. Por tanto, este pliego afectara a todas las obras que comprende el proyecto y que se especifican en los documentos adjuntos.

▪ Artículo 2. Contenido del pliego de condiciones

El siguiente documento expresa el acuerdo entre las partes contratantes de este proyecto. En el se exponen una serie de artículos, que la parte contratante del proyecto deberá hacer constar mediante las acciones pertinentes y que en caso de ser incumplidas, podrán dar lugar a las sanciones correspondientes según la legislación vigente, durante la ejecución de las obras.

Con este documento se intentan evitar posibles conflictos o desacuerdos entre las partes. Por lo tanto, este documento habrá de ser leído y notificado a todas aquellas partes, que tomaran parte en el desarrollo del proyecto, siempre en la medida que se considere oportuno y en el aspecto que a cada uno de ellos concierna.

Se deduce que los dos máximos interesados y por tanto imposible de excluirse de la lectura de este documento, son la parte contratante y el Ingeniero Técnico Director de Obra, por lo que ambos deberán de estar en posesión de una copia íntegra del presente pliego.

Ambas partes serán las responsables de velar por el cumplimiento de las ordenes y sugerencias suscritas, puesto que bajo su mando tal y como se indica en el presente documento, se hallara el personal que compone el resto del equipo de trabajo.

En este documento, junto con los demás que forman parte del presente proyecto, y en especial los Planos y Presupuestos, contemplan la localización y la descripción general de la “plataforma modular para el aprendizaje de microcontroladores”, condiciones que han de cumplir los mismos y las horas de mano de obra necesarias para la realización de esta, además de las instrucciones para su ejecución, medición y abono de los mismos; junto con las normas a seguir por el realizador del proyecto.

▪ Artículo 3. Descripción general de las obras

Las obras objeto del contrato son las que quedan especificadas en los restantes documentos del proyecto: memoria, presupuestos y planos; y que en resumen consisten en la

realización de un conjunto de circuitos electrónicos modulares, un entorno de desarrollo para trabajar con ellos, y una serie de dispositivos contruidos con los circuitos electrónicos anteriores, así como programas de ejemplo de funcionamiento para todos los circuitos y dispositivos, tanto para microcontroladores como para ordenador.

1.1. Normativa

▪ Artículo 1. Normativa vigente

Se tendrán en cuenta las Normas Oficiales Españolas (UNE) y Comunitarias o las equivalentes DIN, ISO, etc. así como la normativas EPA. Todos los materiales utilizados estarán homologados.

Tendremos en cuenta las particulares de este pliego y también las de aplicación a las obras e instalaciones como normas generales o supletorias.

El instalador tendrá en cuenta los reglamentos de seguridad e higiene en el trabajo, de actividades molestas, nocivas y peligrosas, perjudiciales a la salud, de incendios y de cualquier otra índole.

La consecución de todos los permisos y licencias que se estimen necesarias para la realización del citado proyecto corren a cargo de la empresa o sus representantes legales.

▪ Artículo 2. Normas de aplicación general

Además de las normas particulares de este pliego, serán de aplicación a este proyecto, como normas generales o supletorias las siguientes:

Norma UNE 82009-3

Esta parte de la Norma UNE 82009 especifica cuatro medidas intermedias de la precisión, debidas a cambios en las condiciones de observación (tiempo, calibración, operador y equipo) dentro de un laboratorio. Estas medidas intermedias pueden ser establecidas por un experimento dentro de un laboratorio concreto o un experimento interlaboratorios.

Norma UNE 20-621-85/3-1^{er} C.

El objeto de esta norma es complementar la norma UNE 20-621/3 en lo concerniente al apartado 4.1 que en el original aparece como es estudio, así como sustituir el texto del apartado 4.2 e incluir en el capítulo 11 las normas que aparecen en el desarrollo de estos apartados.

Se emplearan en lo posible las normas UNE o bien cualquier tipo de normas supletorias de las citadas. Siempre que los equipos se atengan a unas normas que no sean las descritas en este pliego, estas deberán ser notificadas y presentadas ante el organismo competente.

Además, las pruebas realizadas al final del proyecto serán en la medida de lo posible no destructivas y en caso contrario no serán incluidas en el presupuesto general, sino que correrán a cargo del contratista.

1.2. Condiciones legales

▪ Artículo 1. Documentos

Los Planos, Cuadros de Precios, Presupuesto y Pliego de Condiciones, que se incluyen en el presente proyecto, se consideran con carácter contractual. Los datos incluidos en la Memoria tienen carácter informativo.

Cualquier cambio en el planteamiento de la instalación que implique un cambio sustancial respecto de lo proyectado deberá ponerse en conocimiento del Proyectista para lo que apruebe, si procede y redacte el oportuno reformado del proyecto.

▪ Artículo 2. Compatibilidad entre documentos

En caso de contradicción entre los Planos y la Memoria o entre la Memoria y el Pliego de Condiciones, tiene prioridad lo mencionado en los Planos y en el Pliego de Condiciones.

▪ Artículo 3. Jurisdicción

Para los litigios o diferencias pudieran surgir durante o después de los trabajos, las partes se someterán a los tribunales de Justicia del lugar que radique la Entidad Propietaria.

La Empresa Instaladora es responsable de la ejecución de la instalación en las condiciones establecidas en el contrato y en los documentos que componen el Proyecto (la memoria no tendrá consideración contractual).

▪ Artículo 4. Seguro de los trabajos

La Empresa Instaladora está obligada a asegurar la instalación, durante todo el tiempo que dure su ejecución, hasta la verificación definitiva; la cuantía del seguro coincidirá en todo momento, con el valor que tengan los objetos asegurados. No se podrá disponer del importe del seguro para menesteres ajenos a los de la parte siniestrada de la instalación; la infracción de lo anteriormente expuesto será motivo suficiente para que la Entidad Propietaria pueda rescindir la Empresa Instaladora, sin devolución de la fianza.

Los riesgos asegurados y las condiciones que figuran en la póliza de seguros, los pondrá la Empresa Instaladora antes de contratarlos en conocimiento de la Entidad Propietaria, con el objeto de recabar de ésta su previa conformidad o reparos.

▪ Artículo 5. Accidentes de trabajo y daños a terceros

En caso de accidentes ocurridos con motivo y en el ejercicio de los trabajos para la ejecución de Instalación, la Empresa Instaladora se atendrá a lo dispuesto a estos respectos, en la legislación vigente, y siendo, en todo caso, único responsable de su cumplimiento y sin que, por ningún concepto pueda quedar afectada la Entidad Propietaria responsable por responsabilidad en cualquier aspecto.

La Empresa Instaladora está obligada a adoptar todas las medidas de seguridad que las disposiciones vigentes preceptúan para evitar, en lo posible, accidentes a los empleados.

De los accidentes o perjuicios de todo genero, que por no cumplir la Empresa Instaladora lo legislado sobre seguridad e higiene, pudieran sobrevenir, será esta única

responsable, ya que se considera que en los precios contratados están incluidos todos los gastos precisos para cumplimentar debidamente dichas disposiciones legales.

La Empresa Instaladora es responsable de todos los accidentes que, por inexperiencia o descuido, sobrevinieran en la Instalación.

▪ **Artículo 6. Causas de rescisión del contrato.**

Se consideran causas suficientes de rescisión las que a continuación se señalan.

- Por muerte o incapacidad de la Empresa Instaladora.
- La quiebra de la Empresa Instaladora
- Las alteraciones del contrato por la modificación del Proyecto, en forma tal que presente alteraciones fundamentales del mismo, a juicio del Ingeniero Director.
- El no dar comienzo la Empresa Instaladora a los trabajos dentro del plazo señalado en las condiciones particulares del Proyecto.
- El incumplimiento de las condiciones del Contrato, cuando implique descuido o mala fe, con perjuicio de los intereses de la instalación.
- La terminación del plazo de ejecución de la instalación, sin haberse llegado a esta.
- El abandono de la instalación sin causa justificada.
- La mala fe en la ejecución de los trabajos

1.3. Condiciones de carácter facultativo.

▪ **Artículo 1. Nombramiento del director de obra**

Se nombrará para el cargo de director de obra a un ingeniero técnico que será el encargado de dirigir el diseño y desarrollo del proyecto, la parte contratante deberá poner a disposición del mismo todas las facilidades posibles para que este y su equipo, lleven a cabo la labor de que es objeto el presente proyecto.

El ingeniero no será responsable de los trámites burocráticos o retrasos debidos a estos.

▪ **Artículo 2. Atribuciones del director de obra**

El director de obra será responsable de que se cumplan las especificaciones del presente proyecto, así como de cualquier modificación que éste autorice, según la legislación vigente.

Corresponde al director de obra la interpretación de los distintos documentos de este proyecto. Así mismo, impartirá las pertinentes órdenes al contratista instalador para la correcta interpretación y ejecución de lo proyectado.

El director de obra inspeccionará los materiales así como las condiciones de ejecución y puesta en obra de los mismos, pudiendo exigir todas las comprobaciones que crea pertinentes.

El director de obra podrá designar a un ayudante a cuyas indicaciones se ajustará el contratista instalador.

▪ **Artículo 3. Reclamaciones contra órdenes del director**

Las reclamaciones a las que pueda dar lugar el contratista sólo serán atendidas en el caso de que estas sean puramente económicas. En caso contrario, es decir si dichas reclamaciones son debidas a motivos técnicos, el responsable será única y exclusivamente el ingeniero director de obra.

▪ **Artículo 4. Remisión de solicitud de ofertas**

Con el fin de llevar a cabo la instalación del presente proyecto con los materiales y componentes citados en los documentos correspondientes, se ha procedido a hacer un estudio de mercado, llegándose a la conclusión de que dada la fluctuación a que se ven sometidos los materiales y componentes, se aconseja una inmediata compra de los mismos.

Debido a todo esto, se considera que la oferta y el presupuesto planteado en el presente proyecto tendrán una validez que no será superior a un mes.

▪ **Artículo 5. Copia de documentos**

El contratista dispondrá de una copia de los documentos, pudiendo ser requerida también por el ingeniero director tras la aprobación de la concesión de la obra.

▪ **Artículo 6. Libro de órdenes**

Este libro estará presente en la oficina de la obra y el ingeniero director podrá hacer uso del mismo para las notificaciones que crea correspondientes, estas notificaciones serán tan validas como puedan serlo las del pliego de condiciones.

▪ **Artículo 7. Comienzo de los trabajos y plazo de ejecución**

El plazo de ejecución será contado a partir del inicio de la obra que debe ser notificado por el contratista al ingeniero director, el cual llevará el peso y responsabilidad de la ejecución de la misma.

Las demoras producidas debido a trámites burocráticos serán objeto de revocación del plazo citado, pudiendo ampliarse este, siempre que la duración total de la instalación técnica no supere el plazo citado anteriormente.

▪ **Artículo 8. Condiciones generales de ejecución de los trabajos**

El contratista debe regirse por las “Especificaciones de materiales” y por las “Especificaciones de ejecución”. Sobre él recaen todas las responsabilidades fijadas en el Pliego de condiciones del proyecto, siendo el único responsable de las compras efectuadas durante el periodo de tiempo transcurrido entre la concesión de la obra y el final de la misma y por tanto, de la alteración del presupuesto del presente proyecto.

▪ **Artículo 9. Trabajos defectuosos**

Si durante la instalación surgiesen defectos o si las pruebas finales arrojasen un resultado negativo, el ingeniero director podría ordenar la repetición de las mismas o el desmonte de la instalación para posteriormente montarla correctamente. El ingeniero director es el único que tiene capacidad para tomar este tipo de decisiones.

En el caso de materiales defectuosos o de baja calidad, el ingeniero director podrá notificar su disconformidad al contratista, el cual debe encargarse de subsanar los defectos, sustituyendo los materiales defectuosos por otros que estén en perfectas condiciones.

▪ **Artículo 10. Recepciones provisionales**

Si tras las obras de instalación y en presencia del ingeniero director, el contratista y el propietario, las pruebas realizadas fueran llevadas a cabo con éxito, se dará por percibida la instalación y comenzará a correr el plazo de garantía de 6 meses.

En caso contrario, el ingeniero informará al contratista de los defectos y sus posibles remedios, efectuándose esta operación todas las veces necesarias para dar lugar a una óptima instalación.

Una vez terminado el plazo de garantía, se verificará la recepción definitiva siempre con las mismas características que antes.

▪ **Artículo 11. Liquidación en caso de rescisión**

Se incluirán en el documento de liquidación, el importe de las obras y elementos utilizados hasta el momento en que se produzca la rescisión del contrato.

2. Especificaciones de materiales y equipos

▪ Artículo 1. Especificaciones de materiales

Todos los materiales empleados en la construcción del aparato de este proyecto deben ser los que se detallan en la lista de materiales o en el Presupuesto, debiendo cumplir éstos los valores y tolerancias que se especifiquen para cada componente. El fabricante al cual se le haga el pedido de materiales no es relevante siempre que cumpla con lo anteriormente expuesto.

Para cualquier cambio en alguno de los componentes de los que se detallan en la lista de materiales será consultado con el proyectista. De hacer el cambio y no consultar con el proyectista, este declina toda responsabilidad debida al mal funcionamiento del producto, sobre quien haya autorizado dicho cambio.

Antes de proceder a la soldadura de los componentes en las tarjetas de circuito impreso, cada componente debe ser comprobado para verificar que cumple la tolerancia y valores que nos aseguran el fabricante de los mismos, de no cumplir los requisitos los componentes serán devueltos. No teniendo nada que ver en los pedidos a fabricantes el proyectante no se hará cargo de estas circunstancias.

▪ Artículo 2. Condiciones exigibles a materiales y equipos

Los materiales y equipos que se instalen, serán nuevos y vendrán provistos de la correspondiente certificación de calidad, sin ningún tipo de defecto que pueda afectar a la calidad, estética de la instalación o dificultar el montaje, mantenimiento, funcionamiento y revisiones de la misma.

En la ejecución de la instalación se atenderán las indicaciones de los fabricantes de los materiales y componentes.

Todo material o aparato usado en el montaje de la instalación estará marcado de un modo perdurable con la información de sus características técnicas, el nombre y marca del fabricante.

▪ Artículo 3. Instalaciones no especificadas

Si durante los trabajos de instalación surgiera algún tipo de inconveniente, el ingeniero jefe tendrá absoluta autoridad para realizar los cambios pertinentes, debiendo el contratista atenerse a los cambios que este dictamine y sin derecho a reclamación alguna.

▪ Artículo 4. Lista de materiales empleados

Se consideran materiales de este proyecto todos los que se exponen en las listas de materiales especificados en cualquiera de los documentos del proyecto, no permitiendo el proyectista utilizar otros, sin previo aviso al mismo.

3. Especificaciones de ejecución

3.1. Obligaciones de la propiedad

La construcción del sensor, del que es objeto el presente proyecto, se ejecutará por gestión directa de la propiedad, ya sea contratándola total o parcialmente a un instalador autorizado, que esté en posesión de la titulación correspondiente o en su defecto, del resguardo justificativo de haberle sido concedida; o bien realizándolas directamente en caso de contar con personal que reúna las citadas condiciones. En este último supuesto, se entenderá que la propiedad asume el papel de contratista. Por lo tanto, todas las obligaciones del contratista y en particular las expresadas en este pliego de condiciones, se harán extensivas a la propiedad.

La propiedad está obligada a comunicar al contratista instalador, con la antelación necesaria antes del comienzo de la obra, el nombre y dirección del director técnico de la obra, y a éste último el nombre y dirección del contratista instalador.

Se entiende que toda obra realizada sin la documentación exigida, ha sido realizada sin dirección técnica y bajo la dirección y responsabilidad de la propiedad, siendo ésta en consecuencia la única responsable de tales obras y asumiendo expresamente la responsabilidad que pudiera serle exigida o reclamada por terceros o estamentos oficiales.

3.2. Acciones del contratista instalador

El contratista o empresa instaladora, por ser el adjudicado de la fabricación del sensor, asume la obligación de materializar la obra proyectada, y reconoce que ha examinado y estudiado toda la documentación que forma el proyecto, y que así mismo acepta las responsabilidades de su función y las expresamente señaladas en este pliego de condiciones.

El contratista instalador será responsable único de la ejecución de las obras que se hayan contratado, no teniendo derecho a indemnización alguna por el mayor precio que pudieran costarle, ni por los errores cometidos durante la ejecución.

▪ Artículo 1. Condiciones específicas de ejecución.

Dentro de los trabajos de ejecución a realizar por el contratista, se especifican los siguientes:

- Diseño y confección de todos los documentos y requisitos necesarios para la adquisición y montaje de todos los elementos que intervienen en la instalación, tales como listas de materiales, esquemas, especificaciones, etc.
- Suministro, montaje y conexionado de todos los elementos que intervienen en la instalación salvo aquellos que sean aportados por terceros.

- Obtención de los permisos correspondientes, en caso necesario, para la realización de la instalación.

Las obras a realizar se muestran en los planos de construcción, que deben haber sido aprobados por la dirección técnica y que podrán ser entregados parcialmente. Después de la entrega de planos y durante la ejecución el contratista recibirá, si fuera necesario a juicio de la dirección técnica, revisiones de los planos de construcción. Para la ejecución, el contratista utilizará solamente los planos correspondientes al último número de revisión.

Finalmente, el instalador demostrará el correcto funcionamiento de la obra de acuerdo con lo especificado en este proyecto.

▪ **Artículo 2. Obligaciones adicionales no especificadas**

Es responsabilidad y obligación del instalador llevar a cabo cuanto sea necesario para la buena ejecución y aspecto de las obras, aunque no se halle expresamente determinado en este proyecto y siempre que así lo disponga el director técnico.

▪ **Artículo 3. Mano de obra y herramientas.**

El contratista instalador deberá utilizar en la obra personal debidamente autorizado y cualificado, con experiencia en los campos correspondientes y una antigüedad mínima en la empresa instaladora de tres meses.

En caso de que el contratista tuviera que emplear personal en horas fuera de las estipuladas (horario nocturno, festivo, etc.) por necesidad de cumplimiento de los plazos de ejecución fijados, no se abonará cantidad suplementaria alguna.

Se supone que el contratista posee, toda la herramienta y utillaje necesario para realizar el montaje y las pruebas de la instalación.

▪ **Artículo 4. Accidentes**

El contratista está obligado a conocer, cumplir y hacer cumplir la normativa vigente en materia de seguridad e higiene laboral. Así mismo será responsable ante las autoridades y tribunales de los accidentes que, por inexperiencia o descuido, sobreviniesen.

La dirección técnica declinará toda responsabilidad sobre los accidentes que pudieran originarse por defectos de ejecución, por errónea manipulación de los materiales o de la maquinaria, o los debidos a la deficiente calidad de los materiales.

▪ **Artículo 5. Limpieza y reparación de desperfectos**

Toda la instalación estará cuidadosamente limpia en su presentación final. Si el contratista causase algún desperfecto, tendrá que repararlo por su cuenta, dejándolo todo en el mismo estado en el que estaba en el momento del inicio de la obra.

▪ **Artículo 6. Dudas, ambigüedades y aclaraciones**

En caso de existir falta de información o surgir alguna diferencia de interpretación por contradicción y/o ambigüedad acerca de lo expuesto en este proyecto (los planos y el pliego de condiciones prevalecen sobre los demás documentos), el contratista pondrá tal extremo en

conocimiento del director técnico, con el fin de aclarar la discrepancia. El contratista siempre aceptará la opinión del director técnico. Todo ello se llevará a cabo antes de ejecutar la parte de la obra en conflicto y procediendo, con posterioridad a la decisión de la dirección técnica, a efectuar la correspondiente anotación en el libro de órdenes.

▪ **Artículo 7. Modificación del proyecto**

Si en el transcurso de los trabajos la dirección técnica juzgase necesario realizar alguna modificación no especificada en este proyecto, el contratista la ejecutará con arreglo a las órdenes de modificación y a los planos de construcción y sus eventuales revisiones si las hubiera.

▪ **Artículo 8. Obras complementarias**

Aun cuando este pliego sólo abarca las obras e instalaciones incluidas en este proyecto, se ejecutarán todas las obras que ordene el director técnico como ampliaciones o mejoras, siempre que estén debidamente autorizadas por el propietario.

▪ **Artículo 9. Ampliaciones irresponsables**

El autor del presente proyecto y el director técnico de la obra, declinan cualquier tipo de responsabilidad que se derive de las ampliaciones de la instalación, no incluidas en este proyecto y que se realicen a iniciativa de la propiedad o del contratista, sin ser comunicadas de forma oficial para su inserción en este proyecto mediante el correspondiente anexo.

▪ **Artículo 10. Obras defectuosas**

De todos los materiales y elementos de construcción se presentarán muestras al director técnico y con arreglo a ellas se ejecutará el trabajo. Toda obra ejecutada que a juicio del director técnico sea defectuosa o no esté en consonancia con lo indicado en este pliego, será deshecha y rehecha por el contratista, sin que pueda servir de excusa el que el director técnico haya examinado la instalación en el transcurso de la obra, ni el que esta haya sido abonada total o parcialmente.

3.3. Especificaciones técnicas

La tensión de alimentación de cualquier circuito electrónico nunca será superior a 5V ni inferior a 3V, salvo que se indique lo contrario en su documentación.

4. Pliego de condiciones económicas

Si hubiese una empresa que decida fabricar en serie y comercializar el producto deberá tener en cuenta las siguientes condiciones económicas:

▪ Artículo 1. Importe de la instalación

La valoración de la fabricación será la asignada en el Presupuesto.

Como base fundamental de este Pliego de condiciones económicas, se establece el principio de que la Empresa Instaladora debe percibir el importe de todos los trabajos ejecutados, siempre que éstos se hayan realizado con arreglo a las condiciones que se estipulan en el proyecto.

▪ Artículo 2. Informes financieros

El Proyectista podrá exigir a la Empresa Instaladora la presentación de informes bancarios o de otras entidades o personas, al objeto de cerciorarse de si ésta reúne todas las condiciones requeridas para el exacto cumplimiento del Contrato; dichas referencias, si le son pedidas, las presentará la Empresa Instaladora antes de la firma del Contrato.

▪ Artículo 3. Precios y revisiones

Si ocurriese algún caso por virtud de la cual fuese necesario fijar un nuevo precio se procederá a estudiarlo y convenirlo de la siguiente forma:

- El Adjudicatario formulará por escrito, bajo su firma el precio que considere oportuno.
- El Proyectista estudiará el precio, que según su criterio deba aplicarse.
- Si ambos son coincidentes se formulará por la Dirección Técnica el Acta de Avenencia.
- Si no fuera posible conciliar por simple discusión de los resultados, el Ingeniero Director propondrá al Proyectista que adopte la resolución que estime conveniente, que podrá ser aprobatoria del precio exigido por el Adjudicatario o en otro caso, dará lugar a la segregación de la instalación o a una instalación nueva, para ser ejecutada por otro adjudicatario distinto.

▪ Artículo 4. Reclamaciones de aumento de precios

Si la Empresa Instaladora, antes de la firma del contrato, no hubiese hecho la reclamación u observación oportuna, no podrá bajo ningún pretexto de error u omisión, reclamar aumento de los precios fijados en el cuadro correspondiente del presupuesto que sirve de base para la ejecución de las Instalación.

▪ **Artículo 5. Elementos comprendidos en el presupuesto**

Al fijar los precios en el presupuesto, se ha tenido en cuenta el importe del material, así como impuestos. Por esta razón no se abonará a la Empresa Instaladora cantidad alguna por dichos conceptos.

Los gastos de patente se harán a cargo de la Empresa Contratada, en dichos papeles figurará el proyectista como parte asociada a la patente, previa firma de un contrato de silencio profesional.

▪ **Artículo 6. Equivocación en el presupuesto**

Se supone que la Empresa Instaladora ha hecho un detenido estudio de los documentos que componen el Proyecto y por lo tanto, al no haber hecho ninguna observación sobre posibles errores o equivocaciones en el mismo, se entiende que no hay lugar a disposición alguna en cuanto a lo que a precios se refiere.

▪ **Artículo 7. Fianzas**

Se exigirá a la Empresa Instaladora, para que responda del cumplimiento de lo contratado, una fianza del 10% del presupuesto de las unidades adjudicadas.

▪ **Artículo 8. Ejecución de los trabajos con cargo a al fianza**

Si la Empresa Instaladora, se negase a hacer por su cuenta, los trabajos precisos para utilizar la instalación en las condiciones contratadas, el Ingeniero Director, en nombre y representación de la Entidad Propietaria, lo ordenará ejecutar a un tercero o directamente por administración, abonando su importe con la fianza depositada, sin perjuicio de las acciones legales a que tenga derecho el propietario, en el caso de que el importe de la fianza, no baste para abonar el importe de los gastos efectuados en la instalación, que no fueran de recibo.

▪ **Artículo 9. Devolución de la fianza**

La fianza depositada será devuelta a la Empresa Instaladora en un plazo que no excederá de 8 días, una vez firmada el acta de verificación definitiva de la instalación.

▪ **Artículo 10. Liquidación final**

Terminada la Instalación, se procederá a la liquidación fijada, que incluirá los importes de la instalación realizada y los que constituyen modificaciones del proyecto, siempre y cuando estas hayan sido previamente aprobadas por la Dirección Técnica con sus precios.

De ninguna manera tendrá derecho la Empresa Instaladora, a formular reclamaciones a la Entidad propietaria por aumentos de instalación, si estos no estuviesen autorizados por escrito y con el visto bueno del Ingeniero Director.

▪ **Artículo 11. Liquidación en caso de rescisión**

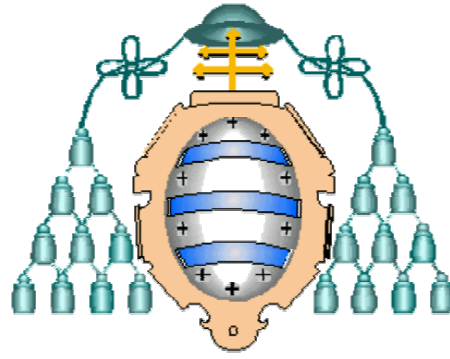
En este caso, la liquidación se hará mediante un contrato liquidatorio, que se redactará por acuerdo de ambas partes. Incluirá el importe de la instalación realizada hasta la fecha de la rescisión.

▪ **Artículo 12. Pagos**

Los pagos se efectuarán por el Propietario en los plazos previamente establecidos.

▪ **Artículo 13. Suspensión por retraso de los pagos**

En ningún caso podrá la Empresa Instaladora, alegando retraso en los pagos, suspender trabajos, con arreglo al plazo en que deben terminarse.



Universidad de Oviedo

Escuela Universitaria de Ingeniería Técnica Industrial

Ingeniería Técnica Industrial, Especialidad Electrónica
Industrial

**Plataforma modular educativa para
Aprendizaje de microcontroladores**

Documento Prevención de Riesgos Laborales

Departamento de Ingeniería Eléctrica,
Electrónica, de Computadores y Sistemas
Autor: Adrián Bulnes Parra
Tutor: Miguel Ángel José Prieto

Gijón, 30 de junio de 2010

Fdo.: Adrián Bulnes Parra

1. ESTUDIO DE SEGURIDAD Y SALUD.....	3
1.1. INTRODUCCIÓN	3
1.2. OBJETIVO DEL ESTUDIO.....	3
1.3. DESCRIPCIÓN DE LAS TAREAS LLEVAS A CABO.....	3
1.3.1 Diseño de las distintas partes del sistema.....	3
1.3.2 Elaboración de las placas perforadas y de circuito impreso.....	4
1.3.3 Construcción de las placas	4
1.3.4 Montaje de los distintos componentes.....	4
1.3.5 Pruebas de funcionamiento y comprobación	4
1.4. IDENTIFICACIÓN DE LOS DISTINTOS RIESGOS EVITABLES	5
1.4.1 Riesgos asociados al diseño de las distintas partes del sistema	5
1.4.2 Riesgos asociados a la elaboración de las placas	6
1.4.3 Riesgos asociados a la construcción del equipo	7
1.4.4 Riesgos asociados a la realización de las pruebas.....	7
1.4.5 Formación.....	8
1.5. MEDIDAS TÉCNICAS PARA EVITAR ESTOS RIESGOS	8
1.5.1 Pantallas de visualización y puesto de trabajo asociado.....	8
1.5.2 Elaboración de PCBs.....	13
1.5.3 Montaje de PCBs	13

1. ESTUDIO DE SEGURIDAD Y SALUD

1.1. Introducción

La Ley de Prevención de Riesgos Laborales 54/2003 del 12 de Diciembre tiene por objeto promover la seguridad y la salud los trabajadores mediante la aplicación de medidas y el desarrollo de las actividades necesarias para la prevención de riesgos derivados del trabajo. A tales efectos, esta Ley establece los principios generales relativos del trabajo, los riesgos profesionales para la protección de la seguridad y de la salud, la eliminación o disminución de los riesgos derivados del trabajo, la información, la consulta, la participación equilibrada y la formación de los trabajadores en materia preventiva, en los términos señalados en la presente disposición. Para el cumplimiento de dichos fines, regula las actuaciones a desarrollar por las administraciones públicas, así como por los empresarios, los trabajadores y sus respectivas organizaciones representativas.

Las disposiciones de carácter laboral contenidas en ella y en sus normas reglamentarias tendrán, en todo caso, el carácter de derecho necesario mínimo indisponible, pudiendo ser mejoradas y desarrolladas en los convenios colectivos.

1.2. Objetivo del estudio

Este estudio de seguridad y salud establece, durante la construcción de la *plataforma modular educativa para aprendizaje de microcontroladores*, las prevenciones respecto a la prevención de riesgos de accidentes y enfermedades profesionales, así como los derivados de los trabajos de reparación, conservación, entretenimiento y mantenimiento, además de las instalaciones respectivas de higiene y bienestar de los trabajadores.

Servirá para dar unas directrices básicas a la empresa correspondiente para llevar a cabo sus obligaciones en el campo de la prevención de los riesgos profesionales, facilitando su desarrollo bajo el control de la dirección facultativa, de acuerdo con el Real Decreto 1.627/1.997

1.3. Descripción de las tareas llevas a cabo

1.3.1 Diseño de las distintas partes del sistema

Para el diseño de las diferentes partes y etapas que componen el sistema, se realizan una serie de cálculos, una consulta digital de información (hojas de características de componentes, página web de fabricantes, etc.) y editar toda la documentación y circuitos que se diseñan con ayuda de programas informáticos. Este diseño lo realizará el ingeniero técnico principalmente en un laboratorio del Área de Tecnología Electrónica, que es donde se encuentran estos equipos.

1.3.2 Elaboración de las placas perforadas y de circuito impreso

En la elaboración de las placas perforadas y de circuito impreso entran en juego varias tareas que se describen a continuación:

- Diseño de las placas con programas de diseño electrónico asistido por ordenador (CAE) y obtención de planos de distribución de pistas y componentes, así como fotolitos.
- Obtención de la PCB a partir de los fotolitos (si procede).
- Taladrado para la inserción de los componentes (si procede).
- Montaje de los distintos componentes, cables, etc.
- Soldado de los componentes anteriores.

Estas tareas las llevan a cabo distintas personas como el ingeniero técnico y el responsable del taller de elaboración de circuitos impresos, y en diferentes emplazamientos como los laboratorios del Área de Tecnología Electrónica, el taller de elaboración de circuitos impresos y el taller mecánico, donde se encuentran localizados los materiales para el mecanizado de las placas.

1.3.3 Construcción de las placas

Su construcción también está compuesta de una serie de tareas que se describen a continuación:

- Mecanizado de las piezas de todos los circuitos y las conexiones entre sí según sean necesarias, labores a realizar por empresas de servicio mecánicos que, al tratarse de una subcontrata, también tendrá que velar por su seguridad.
- Ensamblado de todas las piezas anteriores. Esta tarea se llevará a cabo en el laboratorio del Área de Tecnología Electrónica.

1.3.4 Montaje de los distintos componentes

El montaje de los distintos componentes como las placas consiste tanto en su colocación como en su cableado.

Esta tarea será realizada por operarios y serán realizadas en un laboratorio del Área de Tecnología Electrónica.

1.3.5 Pruebas de funcionamiento y comprobación

Las pruebas se realizarán, como norma general, comprobando el correcto estado de las entradas y salidas analógicas y digitales de los microcontroladores, circuitos integrados específicos y conectores de las placas utilizando voltímetros, amperímetros y osciloscopios. Dado lo delicado de estas pruebas y la importancia que tienen en la estimación del comportamiento del producto final, serán realizadas por el ingeniero técnico en un laboratorio especializado que disponga de los equipos necesarios para llevarlas a cabo.

1.4. Identificación de los distintos riesgos evitables

1.4.1 Riesgos asociados al diseño de las distintas partes del sistema

Los principales riesgos de esta tarea están asociados a la utilización de pantallas de visualización que componen un sistema informático.

El artículo 2 del Real Decreto 488/1997 sobre disposiciones mínimas de seguridad y salud relativas al trabajo que incluye pantallas de visualización establece las siguientes definiciones:

1. Pantalla de visualización

Una pantalla alfanumérica o gráfica independientemente del método de visualización utilizado

2. Puesto de trabajo

El constituido por un equipo con pantalla de visualización previsto en su caso de un teclado o dispositivo de adquisición de datos, de un programa para la interconexión persona/máquina, de accesorios ofimáticos y de un asiento y mesa o superficie de trabajo así como el entorno laboral inmediato.

3. Trabajador

Cualquier trabajador que habitualmente y durante una parte relevante de su trabajo normal utilice un equipo con pantalla de visualización.

Los riesgos asociados a su uso pueden ser de tres tipos:

- Fatiga visual.
- Alteración músculo-esquelética.
- Fatiga mental o psicología.

Fatiga visual:

El trabajo con una pantalla de visualización implica generalmente, cuatro tipos de tareas visuales:

- Tareas de diálogo.
- Tareas de introducción de datos.
- Tareas de programación.
- Tareas de tipo mixto.

La fatiga visual se agudiza en trabajos que incluyen una pantalla de visualización de datos, ya que al tener que moverse continuamente para visualizar los elementos de comunicación los ojos tienen que adaptar continuamente su enfoque siendo este proceso de acomodación y convergencia lo que puede plantear problemas en los trabajadores.

Alteración músculo-esquelética:

Suele deberse a posturas incorrectas ante la pantalla.

- Inclinación excesiva de la cabeza
- Inclinación del tronco
- Flexión de manos
- Estatismo postural

Un factor de gran incidencia en los dolores y los trastornos musculares es la contracción muscular mantenida durante horas y que suele ir asociada a la inmovilización corporal.

Fatiga mental o psicológica

Los factores que inciden en su manifestación pueden ser tanto de tipo laboral (rutina, carga mental) como personal (trastornos del sueño, alteraciones psico-somáticas).

La fatiga mental supone la alteración temporal (disminución) de la eficiencia funcional mental y física. Esta alteración está en función de la intensidad y duración de la actividad precedente y del esquema temporal de la presión mental. La disminución de la eficiencia funcional se manifiesta, por ejemplo, mediante una impresión de fatiga y una peor relación esfuerzo/resultados, a través de la naturaleza y frecuencia de los errores.

1.4.2 Riesgos asociados a la elaboración de las placas

Los riesgos relacionados con la elaboración de PCBs tienen su raíz en el uso de los productos químicos que habitualmente se emplean para este fin. Aunque el sistema es totalmente automático y se desarrolla en un equipo estanco y la agresividad de los productos es relativamente baja, es necesario tener en cuenta los factores de riesgo ligados al manejo de productos químicos, aunque este riesgo sólo se limita a los momentos de reposición y sustitución de los mismos. Según indica la Ley, todos los productos químicos que puedan ser potencialmente peligrosos, deben estar marcados en sus etiquetas con indicadores como los que se muestran en la figura 1.



Figura 1.

Los riesgos, para el caso de los productos que se manejan son los siguientes: tóxico, corrosivo e irritante.

1.4.3 Riesgos asociados a la construcción del equipo

Los riesgos ligados a la construcción del equipo se pueden identificar fácilmente a partir de las tareas que deben ejecutarse para este fin. Son los siguientes:

- Durante el taladrado de la PCB: el mecanizado de la PCB se realiza mediante un taladrado con brocas de 0,8 y 1 mm en un taladro vertical de precisión situado sobre una bancada. Los riesgos que se pueden producir corresponden a cortes producidos por el propio taladro o por impacto de virutas o esquirlas que se proyecten durante el proceso.
- Durante la colación de los componentes se procede al soldado y al corte de las patillas. El soldado se realiza mediante un estañador de temperatura controlada, situada esta en torno a los 350 – 400°C lo que supone un riesgo de quemadura directo por contacto con el estañador y un riesgo indirecto por salpicaduras de estaño fundido o de la resina que suele incluir. También es necesario indicar que el soldado produce vaporización parcial de la sustancia empleada para la soldadura (estaño y resinas) que pueden ser inhalados por el operario que realiza esta acción. Finalmente, hay que indicar el riesgo de corte al efectuar esta acción sobre las patillas sobrantes de los componentes y el de proyección de trozos también durante el corte.

1.4.4 Riesgos asociados a la realización de las pruebas

Las pruebas se realizan con sistemas de alimentación eléctrica conectados a la red y otros equipos también conectados. Aunque las tensiones del sistema objeto del presente proyecto son bajas y, por tanto, seguras, es necesario asumir el riesgo de descarga eléctrica en la conexión de los equipos a la red.

Durante las pruebas se usarán un computador para la toma de datos y su procesamiento por lo que los riesgos citados en el apartado 5.4.1 quedan identificados como posibles dentro de esta fase.

El manejo de recipientes de vidrio fino durante las pruebas, hace posible la producción de cortes en caso de rotura.

1.4.5 Formación

Todo el personal que realizará todas las tareas anteriores debe recibir una exposición de los métodos de trabajo y los riesgos que estos pudieran entrañar, junto con las medidas de seguridad que deberá emplear.

Eligiendo al personal más cualificado y se impartirán cursillos de socorrismo y primeros auxilios de forma que todas dispongan de algún socorrista.

1.5. Medidas técnicas para evitar estos riesgos

1.5.1 Pantallas de visualización y puesto de trabajo asociado

Existen una serie de medias técnicas para evitar los posibles riesgos que se describieron en el apartado anterior. La norma UNE-EN29241.3 proporciona una serie de recomendaciones para las pantallas de ordenador.

Elección del tipo de pantalla

Quizás el primer parámetro a tener en cuenta es la elección de una pantalla de visualización correcta para el tipo de trabajo que se vaya a realizar. El cuadro siguiente muestra el tamaño, resolución y frecuencia de imagen que debe tener una pantalla para cada tipo de trabajo principal:

TRABAJO PRINCIPAL	TAMAÑO (DIAGONAL)	RESOLUCIÓN (PÍXELES)	FRECUENCIA DE IMAGEN
<i>Oficina</i>	35 cm (14'')	640 x 480	70 Hz
<i>Gráficos</i>	42 cm (17'')	800 x 600	70 Hz
<i>Proyecto</i>	50 cm (20'')	1024 x 768	70 Hz

Las tareas que se van a realizar para el presente proyecto se pueden considerar como trabajo de proyecto, por lo tanto la pantalla tendrá que tener un tamaño de 550 cm (20''), una resolución de 1027 x 768 píxeles y una frecuencia de 70 Hz.

Colocación de la pantalla

En lo que concierne a la colocación de la pantalla se recomienda situarla a una distancia superior a 400 mm respecto a los ojos del usuario entre la línea de visión horizontal y la trazada a 60° bajo la horizontal, según se representa la figura 2.

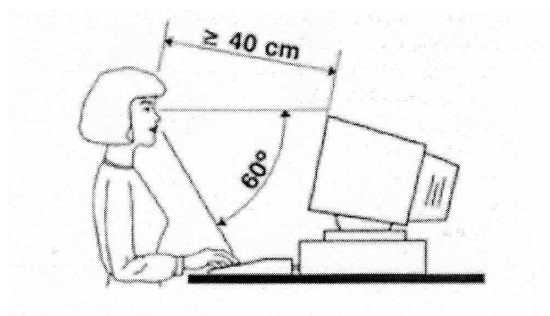


Figura 2

Colocación del teclado

Algunas características del teclado, como su altura, grosor e inclinación, pueden influir en la adopción de posturas incorrectas y originar trastornos en los usuarios. Para prevenir estos riesgos, el diseño del teclado debería cumplir, al menos, los siguientes requisitos:

El cuerpo del teclado debe ser suficientemente plano; se recomienda que la altura de la 3ª fila de teclas (fila central) no exceda de 30 mm respecto a la base de apoyo del teclado y la inclinación de éste debería estar comprendida entre 0 y 25° respecto a la horizontal (véase figura 3)

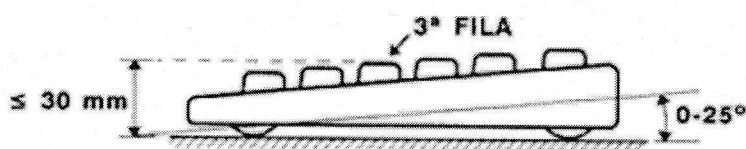


Figura 3

Si el diseño incluye un soporte para las manos, su profundidad debe ser al menos de 10 cm. Si no existe dicho soporte se debe habilitar un espacio similar en la mesa delante del teclado. Este reposo es muy importante para reducir la tensión estática en los brazos y la espalda del usuario.

Con el fin de asegurar la legibilidad de los símbolos de las teclas se recomienda la impresión de caracteres oscuros sobre fondo claro, asimismo, las superficies visibles del teclado no deberían ser reflectantes. Por otro lado, se recomienda que las principales secciones del teclado (bloque alfanumérico, bloque numérico, teclas de cursor y teclas de función) estén claramente delimitadas y separadas entre sí por una distancia de, al menos, la mitad de la anchura de una tecla. Así mismo, la forma, tamaño y fuerza de accionamiento de las teclas deben ser adecuados para permitir un accionamiento cómodo y preciso.

Finalmente, los teclados deberán incluir la letra ñ y los demás caracteres del idioma castellano, de acuerdo con lo establecido en el Real Decreto 564/1993, de 16 de abril.

La silla de trabajo

Las sillas destinadas a los puestos de trabajo con pantallas de visualización deberían cumplir los siguientes requisitos de diseño:

- Altura del asiento ajustable en el rango necesario para la población de usuarios.
- Respaldo con una suave prominencia para dar apoyo a la zona lumbar y con dispositivos para poder ajustar su altura e inclinación.
- Profundidad del asiento regulable, de tal forma que el trabajador pueda utilizar el respaldo sin que el borde del asiento le presione las piernas.
- Mecanismos de ajuste fácilmente manejables en posición sentado y contruidos a prueba de cambios no intencionados.
- Las sillas deberán estar dotadas de 5 apoyos para el suelo.

También deberían incluir ruedas, especialmente cuando se trabaje sobre superficies muy amplias. Las ruedas deben ser adecuadas al tipo de suelo existente, con el fin de evitar desplazamientos involuntarios en suelos lisos y con actividades de tecleo intensivo (véase figura 4).

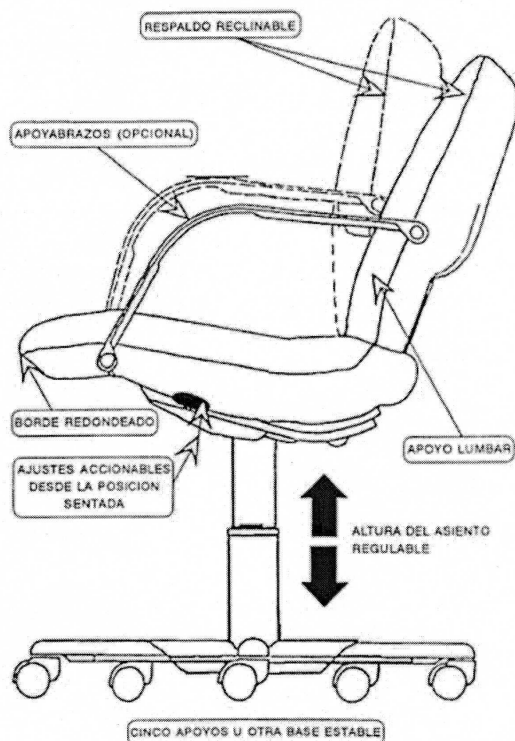


Figura 4

El reposapiés (figura 5) se hace necesario en los casos donde no se puede regular la altura de la mesa y la altura del asiento no permite al usuario descansar los pies en el suelo. Cuando sea utilizado debe reunir las siguientes características:

- Inclinación ajustable entre 0° y 15° sobre el plano horizontal.
- Dimensiones mínimas de 45 cm de ancho por 35 cm de profundidad.
- Tener superficies antideslizantes, tanto en la zona superior para los pies como en sus apoyos para el suelo.

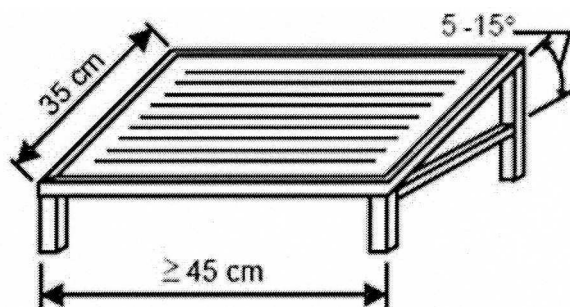


Figura 5

Configuración y definición de los caracteres del texto.

En lo que concierne a la configuración y definición de los caracteres alfanuméricos, la matriz de representación de los caracteres de la pantalla debe estar constituida por un mínimo de 5 x 7 píxeles; elementos más pequeños de la pantalla, direccionales, que forman la trama de la imagen.

No obstante, cuando se requiera una lectura frecuente de la pantalla, o sea importante garantizar la legibilidad del texto, la matriz de representación de los caracteres debe tener al menos 7 x 9 píxeles, según la figura 6.

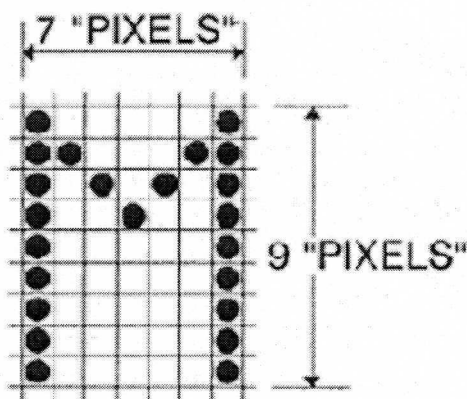


Figura 6

Este requisito lo cumplen sobradamente las actuales pantallas de ordenador tipo VGA o SVGA.

Por lo que se refiere a los aspectos concernientes a la luminiscencia y al contraste de los caracteres, la pantalla debe ser capaz de proporcionar al menos una luminiscencia de 35 Cd/m² (para los caracteres, en polaridad negativa, o para el fondo de pantalla, en polaridad positiva), si bien el nivel preferido es de 100 Cd/m².

La mesa o superficie de trabajo

Las dimensiones de la mesa deben ser suficientes para que el usuario pueda colocar con holgura los elementos de trabajo y, más concretamente, para que pueda situar la pantalla a la distancia adecuada (a 40 cm como mínimo) y el teclado de manera que exista un espacio suficiente delante del mismo para apoyar las manos y los brazos.

Por otro lado, el acabado de las superficies de trabajo debería tener aspecto mate, con el fin de minimizar los reflejos y su color no debería ser excesivamente claro u oscuro. Asimismo, las superficies del mobiliario con las que pueda entrar en contacto el usuario deben ser de baja transmisión térmica y carecer de esquinas o aristas agudas.

Iluminación

En el recinto donde se ubiquen los puestos de trabajo con pantallas de visualización debe existir una iluminación general. Si se utilizan fuentes de iluminación individual complementaria, éstas no deberían ser usadas en las cercanías de la pantalla en el caso de que produzcan deslumbramiento directo o reflexiones. Tampoco deben ser usadas en el caso de que produzcan desequilibrios de luminiscencia que perturben al propio usuario o a los operadores del entorno. En todo caso, el nivel de iluminación debe ser suficiente para el tipo de tarea que se realice en el puesto (por ejemplo, lectura de documentos), pero no debe alcanzar valores que reduzcan el contraste de la pantalla por debajo de lo tolerable.

La pérdida de contraste originada por la iluminación general en las antiguas pantallas de visualización hacía difícil alcanzar el nivel de luz necesario para compatibilizar la lectura de la pantalla con otras tareas. La mayoría de las actuales pantallas de visualización, con tratamiento antirreflejo y mayor rango de regulación del contraste, permiten utilizar un nivel de iluminación de 500 lux, que es el mínimo recomendable para la lectura y escritura de impresos y otras tareas habituales de oficina.

Con el fin de limitar el deslumbramiento directo producido por las luminarias instaladas en el techo, no se debería sobrepasar el límite de 500 Cd/m² para las que son vistas bajo un ángulo inferior a 45° sobre el plano horizontal, siendo recomendable no sobrepasar las 200 Cd/m².

Con el fin de asegurar un equilibrio adecuado de luminiscencias en el campo visual del usuario, se recomienda que entre los componentes de la tarea la relación de luminiscencias no sea superior a 10:1 (por ejemplo, entre pantalla y documento). La relación de luminiscencias entre la tarea y el entorno alejado se considera un aspecto menos crítico (se podrían presentar problemas con relaciones de luminiscencia del orden de 100:1).

Ubicación del puesto de trabajo y de la pantalla

El puesto de trabajo se debe orientar adecuadamente respecto a las ventanas, con el fin de evitar los reflejos que se originarían si la pantalla se orientara hacia ellas, o el deslumbramiento que sufriría el usuario, si fuera este quien se situara frente a las mismas.

Estas medidas pueden ser completadas mediante la utilización de cortinas o persianas que amortigüen la luz, o mediante mamparas en las salas que dispongan de ventanas en más de una pared.

1.5.2 Elaboración de PCBs

La realización de las tarjetas de circuito impreso conlleva una serie de riesgos descritos en el apartado 5.4.2 y que se pueden evitar siguiendo las siguientes precauciones:

- El local donde se realicen debe estar bien ventilado para evitar concentraciones excesivas de vapores nocivos en la atmósfera del local. La ventilación debe ser forzada y el operario utilizará el correspondiente respirador con filtros adecuados.
- Durante el manejo del equipo, la introducción de las placas de circuito impreso, su retirada o la sustitución y reposición de los agentes químicos, el operario usará ropas, gafas protectoras y guantes capaces de soportar las posibles salpicaduras.
- El local estará dotado de un sistema de ducha y, en especial, una ducha de ojos con el fin de paliar los efectos de cualquier salpicadura potencialmente peligrosa.
- La entrada al local deberá estar estrictamente controlada, evitando la entrada de cualquier persona ajena a los trabajos o sin la suficiente formación en las tareas de prevención.

1.5.3 Montaje de PCBs

Riesgos derivados del mecanizado de la tarjeta

El personal que realice estas tareas utilizará guantes de seguridad y gafas protectoras, capaces de evitar el impacto sobre los ojos de virutas y esquirlas.

El taladro se usará adecuadamente y se comprobará que está desconectado antes de cualquier tarea de sustitución de la broca o de limpieza de la zona de trabajo. Asimismo, las brocas deberán estar correctamente situadas en la mordaza del taladro evitando movimientos laterales que puedan producir cortes o la propia rotura de la broca.

Las brocas deberán presentar un buen estado para el corte, evitando presiones excesivas del taladro. El estado del corte de la broca se deberá comprobar cada un cierto número de taladros que dependerá de la naturaleza del material. Para las brocas habituales para circuito impreso y para PCBs de fibra de vidrio con taladros menores de 1 mm, se estima en 50 -100 taladros el periodo de comprobación.

Riesgos derivados del montaje de la tarjeta

La soldadura de los componentes se realiza mediante estaño libre de plomo con alma de resina para facilitar la soldadura. La ausencia de plomo en los actuales estaños para soldadura de PCBs, si bien ha supuesto un incremento en la dificultad de la soldadura, evita la producción de gases muy tóxicos que, anteriormente, eran inhalados por el operario, produciendo un efecto de acumulación en el organismo ya que el plomo es una sustancia difícil de eliminar por los sistemas biológicos.

No obstante, durante el estañado de los componentes se producen algunos vapores menos nocivos pero cuya inhalación debe evitarse por ser, cuando menos, desagradable. Para ello, es conveniente que las tareas de soldadura se realicen en locales bien ventilados y con un extractor de gases en las proximidades.

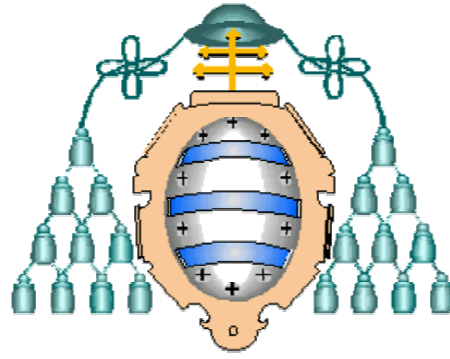
Los riesgos de quemadura se producen por la necesidad de elevar la temperatura por encima de los 300°C para una buena soldadura de los componentes. El estañador, cuya punta debe mantener este valor de temperatura, debe ser termorregulado para evitar que adquiera mayores temperaturas si está conectado durante algún tiempo sin uso, deberá disponer de una empuñadura de material térmicamente aislante, de tamaño suficiente como para poderlo coger sin riesgo y deberá estar situado sobre un soporte rígido y cerrado por todos los lados excepto por la parte superior. El usuario deberá dejar el estañador sobre su soporte cuando no lo esté usando y deberá evitar que quede, en ningún caso, sobre la mesa o colgando de su cable.

La soldadura se realizará con guantes resistentes al calor y con gafas protectoras para evitar cualquier quemadura por gotas de estaño o proyecciones de resina candente.

Riesgos derivados de las pruebas de los circuitos electrónicos construidos como parte de la plataforma modular para aprendizaje:

Las conexiones de todos los equipos a la red eléctrica deberán estar dotadas de toma de tierra y se verificará que estas conexiones son correctas antes de conectar ningún otro sistema.

Las fuentes de alimentación deberán ser fijadas a las tensiones correspondientes de alimentación, a saber, +5V salvo que se indique lo contrario, extremo que será comprobado antes de realizar el conexionado de cualquiera de los circuitos.



Universidad de Oviedo

Escuela Universitaria de Ingeniería Técnica Industrial

Ingeniería Técnica Industrial, Especialidad Electrónica
Industrial

**Plataforma modular educativa para
Aprendizaje de microcontroladores**

Documento Presupuesto

Departamento de Ingeniería Eléctrica,
Electrónica, de Computadores y Sistemas
Autor: Adrián Bulnes Parra
Tutor: Miguel Ángel José Prieto

Gijón, 30 de junio de 2010

Fdo.: Adrián Bulnes Parra

1. MATERIAL.....	3
1.1. MBP8	3
1.2. MBP14	3
1.3. MBP18	4
1.4. MBP40	4
1.5. LTIND-A.....	4
1.6. SISW-SPST	4
1.7. LTIL-A	5
1.8. SIBW-1Y	5
1.9. AO-SPK	5
1.10. SIDST-GP2.....	5
1.11. LT7S-HN (7 SEG., 6 DISP.)	6
1.12. CMIR-RC	6
1.13. MC2A	6
1.14. MC2B	6
1.15. PWR-DD5	7
1.16. PWR-AD5	7
1.17. CMSP-MAX.....	7
1.18. ELECTROOCULÓGRAFO.....	8
1.19. “KETJO” – OCTÓPODO	8
1.20. CLAQUETA DIGITAL	8
1.21. “PCBot”	9
2. MANO DE OBRA.....	10
2.1. FACTURACIÓN POR HORA	10
2.2. MANO DE OBRA TOTAL.....	10
3. PRESUPUESTO GENERAL.....	13

1. Material

A continuación se detalla el coste del material empleado en la construcción de cada uno de los circuitos electrónicos.

Se ha construido una Placa Principal y un Módulo de cada tipo para hacer pruebas, fotografías y para ser utilizados como ejemplos. Se han construido unidades adicionales de algunos de estos circuitos debido a que fueron necesarios para su utilización en alguna de las aplicaciones de ejemplo.

Se ha separado el coste del material de cada circuito individual para simplificar los cálculos en el futuro, a la hora de utilizar y vender los componentes necesarios para construir cada uno de los circuitos.

Los precios de las placas perforadas, placas de circuito impreso, longitud de cable y funda termorretráctil se han ajustado automáticamente al tamaño real de cada uno.

1.1. MBP8

Cantidad	Componente	Precio u.	Precio total
1	PIC12F683-I/P	1,50 €	1,50 €
1	Placa perforada	0,41 €	0,41 €
1	Zócalo PDIP 8 pines	0,80 €	0,80 €
6	Pines macho acodados	0,02 €	0,11 €
20	Pines macho	0,01 €	0,26 €
1	Condensador cerámico 100nF	0,15 €	0,15 €
	Total:		3,23 €

1.2. MBP14

Cantidad	Componente	Precio u.	Precio total
1	Placa perforada	0,56 €	0,56 €
1	PIC16F688-I/P	2,00 €	2,00 €
1	Zócalo PDIP 14 pines	0,80 €	0,80 €
6	Pines macho acodados	0,02 €	0,11 €
36	Pines macho	0,01 €	0,47 €
1	Condensador electrolítico 100uF	0,15 €	0,15 €
1	Condensador cerámico 100nF	0,15 €	0,15 €
	Total:		4,24 €

1.3. MBP18

Cantidad	Componente	Precio u.	Precio total
1	Placa perforada	0,82 €	0,82 €
1	PIC16F88-I/P	3,00 €	3,00 €
1	Zócalo PDIP 18 pines	0,80 €	0,80 €
2	Resistencia 10kΩ, 1/4W	0,03 €	0,06 €
6	Pines macho acodados	0,02 €	0,11 €
52	Pines macho	0,01 €	0,68 €
1	Condensador electrolítico 100uF	0,15 €	0,15 €
1	Condensador cerámico 100nF	0,15 €	0,15 €
	Total:		5,77 €

1.4. MBP40

Cantidad	Componente	Precio u.	Precio total
1	Placa perforada	1,76 €	1,76 €
1	PIC16F887-I/P	2,50 €	2,50 €
1	Zócalo PDIP 40 pines	0,80 €	0,80 €
2	Resistencia 10kΩ, 1/4W	0,03 €	0,06 €
6	Pines macho acodados	0,02 €	0,11 €
122	Pines macho	0,01 €	1,59 €
1	Condensador electrolítico 100uF	0,15 €	0,15 €
1	Condensador cerámico 100nF	0,15 €	0,15 €
	Total:		7,12 €

1.5. LTIND-A

Cantidad	Componente	Precio u.	Precio total
1	LED	0,10 €	0,10 €
1	Resistencia 470Ω, 1/4W	0,03 €	0,03 €
3	Pines hembra	0,02 €	0,06 €
2	Cable. 30cm	0,02 €	0,04 €
3	Funda termorretráctil 1,5mm	0,03 €	0,08 €
1	Funda termorretráctil 3mm	0,02 €	0,02 €
	Total:		0,33 €

1.6. SISW-SPST

Cantidad	Componente	Precio u.	Precio total
1	Pulsador	0,58 €	0,58 €
1	Resistencia 1kΩ, 1/4W	0,03 €	0,03 €
1	Resistencia 10kΩ, 1/4W	0,03 €	0,03 €
3	Pines hembra	0,02 €	0,06 €
2	Cable. 30cm	0,02 €	0,04 €
1	Funda termorretráctil 1,5mm	0,03 €	0,03 €
3	Funda termorretráctil 3mm	0,02 €	0,05 €
1	Funda termorretráctil 6mm	0,03 €	0,03 €
	Total:		0,85 €

1.7. LTIL-A

Cantidad	Componente	Precio u.	Precio total
1	LED 3V, 50-100mA	2,00 €	2,00 €
1	Placa perforada	0,12 €	0,12 €
1	2N3904	0,55 €	0,55 €
1	Resistencia 1kΩ, 1/4W	0,03 €	0,03 €
2	1N4007	0,10 €	0,20 €
3	Pines hembra	0,02 €	0,06 €
4	Cable. 15cm	0,01 €	0,04 €
5	Funda termorretráctil 1,5mm	0,03 €	0,14 €
1	Funda termorretráctil 6mm	0,03 €	0,03 €
Total:			3,16 €

1.8. SIBW-1Y

Cantidad	Componente	Precio u.	Precio total
1	CNY70	1,15 €	1,15 €
1	Resistencia 220Ω, 1/4W	0,03 €	0,03 €
1	Resistencia 18kΩ, 1/4W	0,03 €	0,03 €
3	Pines hembra	0,02 €	0,06 €
3	Cable. 30cm	0,02 €	0,06 €
3	Funda termorretráctil 1,5mm	0,03 €	0,08 €
1	Funda termorretráctil 3mm	0,02 €	0,02 €
1	Funda termorretráctil 6mm	0,03 €	0,03 €
Total:			1,46 €

1.9. AO-SPK

Cantidad	Componente	Precio u.	Precio total
1	Altavoz	0,65 €	0,65 €
1	Resistencia 1kΩ, 1/4W	0,03 €	0,03 €
3	Pines hembra	0,02 €	0,06 €
2	Cable. 30cm	0,02 €	0,04 €
3	Funda termorretráctil 1,5mm	0,03 €	0,08 €
1	Funda termorretráctil 3mm	0,02 €	0,02 €
Total:			0,88 €

1.10. SIDST-GP2

Cantidad	Componente	Precio u.	Precio total
1	GP2D-12 o GP2Y0A21YK	13,00 €	13,00 €
3	Pines hembra	0,02 €	0,06 €
3	Cable. 30cm	0,02 €	0,06 €
6	Funda termorretráctil 1,5mm	0,03 €	0,16 €
Total:			13,29 €

1.11. LT7S-HN (7 seg., 6 disp.)

Cantidad	Componente	Precio u.	Precio total
13	2N3904	0,55 €	7,15 €
7	2N3906	0,55 €	3,85 €
6	Resistencia 1kΩ, 1/4W	0,03 €	0,16 €
14	Resistencia 5.6kΩ, 1/4W	0,03 €	0,38 €
2	Pines macho	0,01 €	0,03 €
13	Pines hembra	0,02 €	0,26 €
26	Cable. 15cm	0,02 €	0,55 €
1	Resistencia 270Ω, 1/4W	0,03 €	0,03 €
13	Funda termorretráctil 1,5mm	0,03 €	0,35 €
Total:			12,75 €

1.12. CMIR-RC

Cantidad	Componente	Precio u.	Precio total
1	TSOP17xx, SFH5110 o equivalente	1,33 €	1,33 €
1	Resistencia 1kΩ, 1/4W	0,03 €	0,03 €
3	Pines hembra	0,02 €	0,06 €
3	Cable. 30cm	0,02 €	0,06 €
5	Funda termorretráctil 1,5mm	0,03 €	0,14 €
1	Funda termorretráctil 3mm	0,02 €	0,02 €
Total:			1,64 €

1.13. MC2A

Cantidad	Componente	Precio u.	Precio total
1	L293B o L293NE	2,30 €	2,30 €
1	Zócalo PDIP 16 pines	0,80 €	0,80 €
8	1N4148	0,12 €	0,96 €
1	Jumper	0,03 €	0,03 €
16	Pines macho	0,01 €	0,22 €
22	Pines hembra	0,02 €	0,44 €
1	Condensador cerámico 56nF	0,15 €	0,15 €
14	Cable. 30cm	0,02 €	0,29 €
22	Funda termorretráctil 1,5mm	0,03 €	0,59 €
1	Circuito impreso	4,00 €	4,00 €
Total:			9,78 €

1.14. MC2B

Cantidad	Componente	Precio u.	Precio total
1	L293D	3,60 €	3,60 €
1	Zócalo PDIP 16 pines	0,80 €	0,80 €
1	Placa perforada	0,34 €	0,34 €
12	Pines macho	0,01 €	0,16 €
20	Pines hembra	0,02 €	0,40 €
12	Cable. 30cm	0,02 €	0,25 €
20	Funda termorretráctil 1,5mm	0,03 €	0,54 €
Total:			6,09 €

1.15. PWR-DD5

Cantidad	Componente	Precio u.	Precio total
1	LM7805	0,75 €	0,75 €
1	1N4007	0,10 €	0,10 €
2	Pines hembra	0,02 €	0,04 €
2	Pines macho	0,01 €	0,03 €
4	Cable. 15cm	0,01 €	0,04 €
6	Funda termorretráctil 1,5mm	0,03 €	0,16 €
1	Funda termorretráctil 3mm	0,02 €	0,02 €
Total:			1,14 €

1.16. PWR-AD5

Cantidad	Componente	Precio u.	Precio total
1	LM7805	0,75 €	0,75 €
1	Condensador electrolítico 4700uF	0,60 €	0,60 €
1	Condensador cerámico 100nF	0,15 €	0,15 €
4	1N4007	0,10 €	0,40 €
6	Pines hembra	0,02 €	0,12 €
4	Pines macho	0,01 €	0,05 €
4	Cable. 30cm	0,02 €	0,08 €
6	Funda termorretráctil 1,5mm	0,03 €	0,16 €
Total:			2,32 €

1.17. CMSP-MAX

Cantidad	Componente	Precio u.	Precio total
1	MAX232	1,85 €	1,85 €
1	Placa perforada	0,50 €	0,50 €
4	Condensador electrolítico 1uF	0,15 €	0,60 €
1	Condensador cerámico 100nF	0,15 €	0,15 €
9	Pines macho	0,01 €	0,12 €
15	Pines hembra	0,02 €	0,30 €
6	Cable. 15cm	0,01 €	0,06 €
1	Resistencia 270Ω, 1/4W	0,03 €	0,03 €
1	Hembra DB-15 para cable	1,80 €	1,80 €
Total:			5,41 €

1.18. Electrooculógrafo

Cantidad	Componente	Precio u.	Precio total
1	Placa perforada	0,14 €	0,14 €
5	Pines macho	0,01 €	0,07 €
6	Pines hembra	0,02 €	0,12 €
1	AD620AN	18,00 €	18,00 €
1	Zócalo PDIP 8 pines	0,80 €	0,80 €
2	Pinzas de cocodrilo	0,15 €	0,30 €
3	Cable. 15cm	0,01 €	0,03 €
2	Cable. 50cm	0,04 €	0,07 €
1	MBP18	5,77 €	5,77 €
1	AO-SPK	0,88 €	0,88 €
2	LTIND-A	0,33 €	0,66 €
1	SISW-SPST	0,85 €	0,85 €
	Total:		27,69 €

1.19. “KETJO” – Octópodo

Cantidad	Componente	Precio u.	Precio total
1	MC2A	9,78 €	9,78 €
1	CMIR-RC	1,64 €	1,64 €
1	MBP18	5,77 €	5,77 €
2	Motor DC	12,00 €	24,00 €
1	Piezas K'NEX	48,00 €	48,00 €
4	Pilas AA recargables Ni-MH alta capacidad	3,50 €	14,00 €
1	Portapilas 4 AA	0,55 €	0,55 €
	Total:		103,74 €

1.20. Claqueta digital

Cantidad	Componente	Precio u.	Precio total
1	MBP40	7,12 €	7,12 €
6	Display 7 segmentos (gran tamaño)	3,84 €	23,04 €
1	LTIND-A	0,33 €	0,33 €
7	SISW-SPST	0,85 €	5,95 €
1	LT7S-HN (para 6 displays, 7 segmentos)	12,75 €	12,75 €
1	PWR-DD5	1,14 €	1,14 €
1	Pila 9V	1,50 €	1,50 €
1	Portapilas 9V	0,60 €	0,60 €
1	Conmutador DPDT	1,20 €	1,20 €
53	Cable. 30cm	0,02 €	1,11 €
15	Cable. 15cm	0,01 €	0,16 €
1	Carcasa	46,00 €	46,00 €
	Total:		100,90 €

1.21. “PCBot”

Cantidad	Componente	Precio u.	Precio total
1	MBP40	7,12 €	7,12 €
2	LTIL-A	3,16 €	6,32 €
4	LTIND-A	0,33 €	1,32 €
2	SISW-SPST	0,85 €	1,70 €
1	MC2A	9,78 €	9,78 €
4	Pilas AA Ni-MH alta capacidad	3,50 €	14,00 €
1	Portapilas AA	0,55 €	0,55 €
1	CMSP-MAX	5,41 €	5,41 €
1	Adaptador USB-Serie	18,45 €	18,45 €
1	Webcam	35,95 €	35,95 €
1	Ordenador portátil	350,00 €	350,00 €
2	Motor DC	12,00 €	24,00 €
1	Carcasa	22,00 €	22,00 €
	Total:		496,60 €

2. Mano de obra

Coste de la mano de obra empleada para el desarrollo del proyecto, incluyendo planificación del mismo, diseño de circuitos, construcción de una unidad de cada circuito electrónico diseñado, desarrollo de programas para microcontroladores y ordenadores, documentación, etc.

2.1. Facturación por hora

Tipo	Coste por hora
Ingeniería	40,00 €
Construcción	25,00 €
Documentación	15,00 €

2.2. Mano de obra total

(siguiente página)

Horas	Tarea	Precio total
40	Planificación del proyecto	1.600,00 €
12	Planificación de las Placas Principales	480,00 €
7	Planificación de los Módulos	280,00 €
10	Diseño MBP18	400,00 €
7	Construcción MBP18	175,00 €
6	Desarrollo prog ejemplo MBP18 y pruebas	240,00 €
5	Diseño MBP8	200,00 €
4	Construcción MBP8	100,00 €
5	Desarrollo prog ejemplo MBP8 y pruebas	200,00 €
5	Diseño MBP14	200,00 €
4	Construcción MBP14	100,00 €
5	Desarrollo prog ejemplo MBP14 y pruebas	200,00 €
5	Diseño MBP40	200,00 €
4	Construcción MBP40	100,00 €
5	Desarrollo prog ejemplo MBP40 y pruebas	200,00 €
6	Diseño LTIND-A	240,00 €
3	Construcción LTIND-A	75,00 €
5	Desarrollo prog ejemplo LTIND-A y pruebas	200,00 €
4	Diseño SISW-SPST	160,00 €
3	Construcción SISW-SPST	75,00 €
4	Desarrollo prog ejemplo SISW-SPST y pruebas	160,00 €
4	Diseño LTIL-A	160,00 €
3	Construcción LTIL-A	75,00 €
1	Desarrollo prog ejemplo LTIL-A y pruebas	40,00 €
5	Diseño SIBW-1Y	200,00 €
4	Construcción SIBW-1Y	100,00 €
4	Desarrollo prog ejemplo SIBW-1Y y pruebas	160,00 €
2	Diseño AO-SPK	80,00 €
2	Construcción AO-SPK	50,00 €
3	Desarrollo prog ejemplo AO-SPK y pruebas	120,00 €
2	Diseño SIDST-GP2	80,00 €
2	Construcción SIDST-GP2	50,00 €
4	Desarrollo prog ejemplo SISW-SPST y pruebas	160,00 €
4	Diseño CMIR-RC	160,00 €
3	Construcción CMIR-RC	75,00 €
2	Desarrollo prog ejemplo CMIR-RC y pruebas	80,00 €

Horas	Tarea	Precio total
9	Diseño MC2A	360,00 €
8	Construcción MC2A	200,00 €
7	Desarrollo prog ejemplo MC2A y pruebas	280,00 €
4	Diseño MC2B	160,00 €
3	Construcción MC2B	75,00 €
4	Desarrollo prog ejemplo MC2B y pruebas	160,00 €
7	Diseño CMSP-MAX	280,00 €
3	Construcción CMSP-MAX	75,00 €
8	Desarrollo prog ejemplo CMSP-MAX y pruebas	320,00 €
5	Diseño LT7S-HN	200,00 €
4	Construcción LT7S-HN	100,00 €
3	Desarrollo prog ejemplo LT7S-HN y pruebas	120,00 €
3	Diseño PWR-DD5	120,00 €
3	Construcción PWR-DD5	75,00 €
1	Pruebas PWR-DD5	40,00 €
5	Documentación PWR-DD5	75,00 €
3	Diseño PWR-AD5	120,00 €
3	Construcción PWR-AD5	75,00 €
1	Pruebas PWR-AD5	40,00 €
5	Documentación PWR-AD5	75,00 €
12	Diseño Electrooculógrafo	480,00 €
27	Construcción Electrooculógrafo (y circuitos necesarios)	675,00 €
8	Programación y pruebas Electrooculógrafo	320,00 €
7	Diseño Octópodo	280,00 €
32	Construcción Octópodo (y circuitos necesarios)	800,00 €
4	Programación y pruebas Octópodo	160,00 €
5	Diseño Claqueta	200,00 €
39	Construcción Claqueta (y circuitos necesarios)	975,00 €
4	Programación y pruebas Claqueta	160,00 €
14	Diseño PCBot	560,00 €
51	Construcción PCBot	1.275,00 €
49	Programación y pruebas PCBot	1.960,00 €
26	Documentación: fotografías	390,00 €
62	Documentación: memoria, presupuesto y maquetado	930,00 €
Total:		19.290,00 €

Nota: El tiempo de desarrollo de MBP18 ha sido más largo que el de otras placas debido a que fue el primer modelo que se desarrolló y sentó la base de desarrollo para el resto del proyecto.

3. Presupuesto general

Resumen del presupuesto total del proyecto, es decir, la suma de los apartados anteriores.

Descripción	Precio total
Material MBP8	3,23 €
Material MBP14	4,24 €
Material MBP18	5,77 €
Material MBP40	7,12 €
Material LTIND-A	0,33 €
Material SISW-SPST	0,85 €
Material LTIL-A	3,16 €
Material SIBW-1Y	1,46 €
Material AO-SPK	0,88 €
Material SIDST-GP2	13,29 €
Material LT7S-HN	12,75 €
Material CMIR-RC	1,64 €
Material MC2A	9,78 €
Material MC2B	6,09 €
Material PWR-DD5	1,14 €
Material PWR-AD5	2,32 €
Material CMSP-MAX	5,41 €
Material Electrooculógrafo	27,69 €
Material Octópodo	103,74 €
Material Claqueta	100,90 €
Material PCBot	496,60 €
Mano de obra	19.290,00 €
Total:	20.098,39 €

Descripción	Precio total
Presupuesto de ejecución material	20.098,39 €
Gastos generales (16%)	3.215,74 €
Beneficio industrial (6%)	1.205,90 €
Presupuesto de ejecución por contrata	24.520,04 €
IVA (18%)	4.413,61 €
Total:	28.933,64 €