Guía de programación de Microcontroladores AVR con SL (Codeblock-AVR-GCC Toolchain)

Tabla de contenido

1. Introducción

El objeto de esta guía es el de iniciar a los entusiastas de la electrónica a la programación y utilización de microcontroladores, específicamente de los microcontroladores AVR de Atmel.

En primer lugar, es necesario responder a la pregunta ¿Qué son los microcontroladores?. Los microcontroladores son básicamente dispositivos electrónicos capaces de llevar a cabo procesos lógicos. Estos procesos o acciones son programados usualmente en lenguaje C o en ensamblador por el usuario, programa que se compila y se transforma a código hexadecimal para luego ser introducido en el microcontrolador a través de un dispositivo

de hardware llamado programador, el cual sirve de interfaz entre el computador donde se genera el código hexadecimal y el microcontrolador.

Se dice que un controlador es un dispositivo que se emplea para manejar uno o varios procesos. Por ejemplo, para ver televisión, un controlador evalúa la señal que ingresa por la antena y la procesa para que esta señal llegue a la pantalla con el mismo nivel promedio, sin importar el nivel de la señal que ingresa, siempre que esté dentro de determinados parámetros. Hasta hace algún tiempo, los controladores se construían con componentes electrónicos de lógica discreta; posteriormente se emplearon los microprocesadores, apoyados con chips de memoria y dispositivos de E/S (Entrada y Salida) sobre una tarjeta de circuito impreso.

A medida que los avances científicos y tecnológicos en la fabricación de componentes electrónicos continuaba en un auge cada vez mayor, los elementos del controlador se han podido incluir en un solo circuito integrado, el cual recibe el nombre de microcontrolador. Es decir, un microcontrolador es un chip que posee en su interior a un microprocesador, memoria de programa, memoria de datos y puertos para comunicarse con el exterior. Por lo tanto, tomando en cuenta lo anterior, se puede decir que un microcontrolador es un circuito integrado o chip que incluye en su interior las tres unidades funcionales de una computadora: unidad central de procesamiento, memoria y unidades de E/S (entrada/salida). Son diseñados para reducir el costo económico y el consumo de energía de un sistema en particular. Un microcontrolador difiere de una CPU normal, debido a que es más fácil convertirla en una computadora en funcionamiento, con un mínimo de chips externos de apoyo. La idea es que el chip simplemente se coloque en el dispositivo, conectado a las fuentes de energía y de información o datos que necesite. Un microprocesador tradicional no nos permite hacer esto, debido a que éste espera que todas estas tareas sean manejadas por

otros chips, es decir, es necesario agregarle módulos de entrada/salida (puertos) y memoria para almacenamiento de información.

Por lo tanto, se puede decir que, desde un punto de vista básico, los microcontroladores están conformados por los siguientes componentes:

- Procesador
- Buses
- Memoria de programas
- Memoria de datos
- Periféricos
- Puertos de E/S

2. Referencias a microcontroladores AVR

2.1 Atmel AVR

Los microcontroladores AVR son microcontroladores RISC (*Reduced Instruction Set Computer* o Computadora de Juego de Instrucciones Reducido) del fabricante Atmel. La arquitectura de los AVR fue concebida por dos estudiantes en el Norwegian Institute of Technology, y posteriormente refinada y desarrollada en Atmel Norway, la empresa subsidiaria de Atmel, fundada por los dos arquitectos del chip.

El AVR es una CPU de arquitectura Harvard, lo que quiere decir que poseen buses dedicados para la memoria de datos y la memoria de programa. Tiene 32 registros de 8 bits. Algunas instrucciones sólo operan en un subconjunto de estos registros. La concatenación de los 32 registros, los

registros de entrada/salida y la memoria de datos conforman un espacio de direcciones unificado, al cual se accede a través de operaciones de carga/almacenamiento. A diferencia de los microcontroladores PIC, el stack o pila se ubica en este espacio de memoria unificado, y no está limitado a un tamaño fijo.

El AVR fue diseñado desde un comienzo para la ejecución eficiente de código C compilado. Al igual que los más conocidos microcontroladores PIC del fabricante Microchip Technology Inc., tiene una comunidad de seguidores, especialmente en foros de internet; el de mayor auge, en idioma inglés, es el llamado AVRFreaks. Estos microcontroladores están soportados por tarjetas de desarrollo de costo razonable, capaces de descargar el código al microcontrolador, y por versiones de las herramientas libres del proyecto GNU. Esto último es posible por su uniformidad en el acceso al espacio de memoria, propiedad de la que carecen los procesadores de memoria segmentada o por bancos, como es el caso de los microcontroladores PIC.

2.2 Programación ISP

La programación In-System (ISP) es la habilidad de ciertos dispositivos lógicos programables, microcontroladores y otros chips electrónicos programables, de ser programados mientras están instalados en un sistema completo, en lugar de que tengan que ser programados antes de ser instalados en un sistema.

La principal ventaja de esta característica es que permite a los fabricantes de dispositivos electrónicos integrar la programación y las pruebas dentro de una única fase de producción, en lugar de tener que separar el momento de la programación del ensamblado del sistema.

Permite también a los fabricantes programar los chips en su propia línea de producción en lugar de tener que comprar chips preprogramados, haciendo posible aplicar cambios de código o diseño en el medio del proceso de producción.

Generalmente, los chips que soportan ISP, tienen circuitos internos que generan cualquier voltaje de programación necesario a partir del voltaje normal que suministra el sistema, y se comunican con el programador a través de un protocolo serial.

3. Herramientas a utilizar

Para la programación de microcontroladores AVR, es necesario que se disponga de ciertas herramientas y dispositivos:

- Microcontrolador AVR
- Protoboard
- Cables
- Fuente de Poder
- Computadora
- Hardware Programador
- Software para la generación del código
- Diversos dispositivos electrónicos (para los ejemplos)

En cuanto al microcontrolador AVR, en esta guía se utiliza el AVR **ATMega88** en su presentación DIP (Dual In-line Package), debido a que, aparte de ser uno de los modelos más emblemáticos de la familia AVR, se puede conseguir con cierta facilidad en el territorio nacional.

La características de este modelo son:

- Memoria Flash: 8Kbytes

- Memoria EEPROM: 512Bytes
- Memoria SRAM: 1Kbytes
- Tamaño del vector de interrupciones: 1 instrucción por palabra/vector
- Nro. de Pines: 28
- Nro. de Registros de Propósito General: 32
- Frecuencia de Operación Máxima: 20 MHZ
- Nro. Máximo de Pines de E/S: 23
- Interrupciones Externas: 24
- Canales ADC: 8
- Canales PWM: 6
- Timers: 3
- Oscilador RC Calibrado

Además se necesita disponer de un **protoboard**, así como cables y una fuente de poder, para el montaje de los circuitos, aparte de una computadora donde generar el código con el que posteriormente será programado el microcontrolador.

Para este proceso, se necesita una pieza de hardware llamado Programador. Este dispositivo sirve como medio de comunicación entre la computadora personal y el microcontrolador.

En esta guía se utiliza el programador **AVR-Prog**, de fabricación nacional, aunque es posible utilizar cualquier programador que utilice la programación ISP (In System Programming). Sin embargo, existe una diversidad de información sobre la fabricación de este tipo de dispositivos, por lo que los usuarios que dispongan conocimientos de electrónica pueden fabricar sus propios programadores.

4. **Circuito para la programación ISP**

En primer lugar, es necesario montar el siguiente circuito (figura 1), el cual establece las conexiones necesarias entre el microcontrolador ATMega88 y el programador AVR-Prog para que “suba” el código hex correctamente.

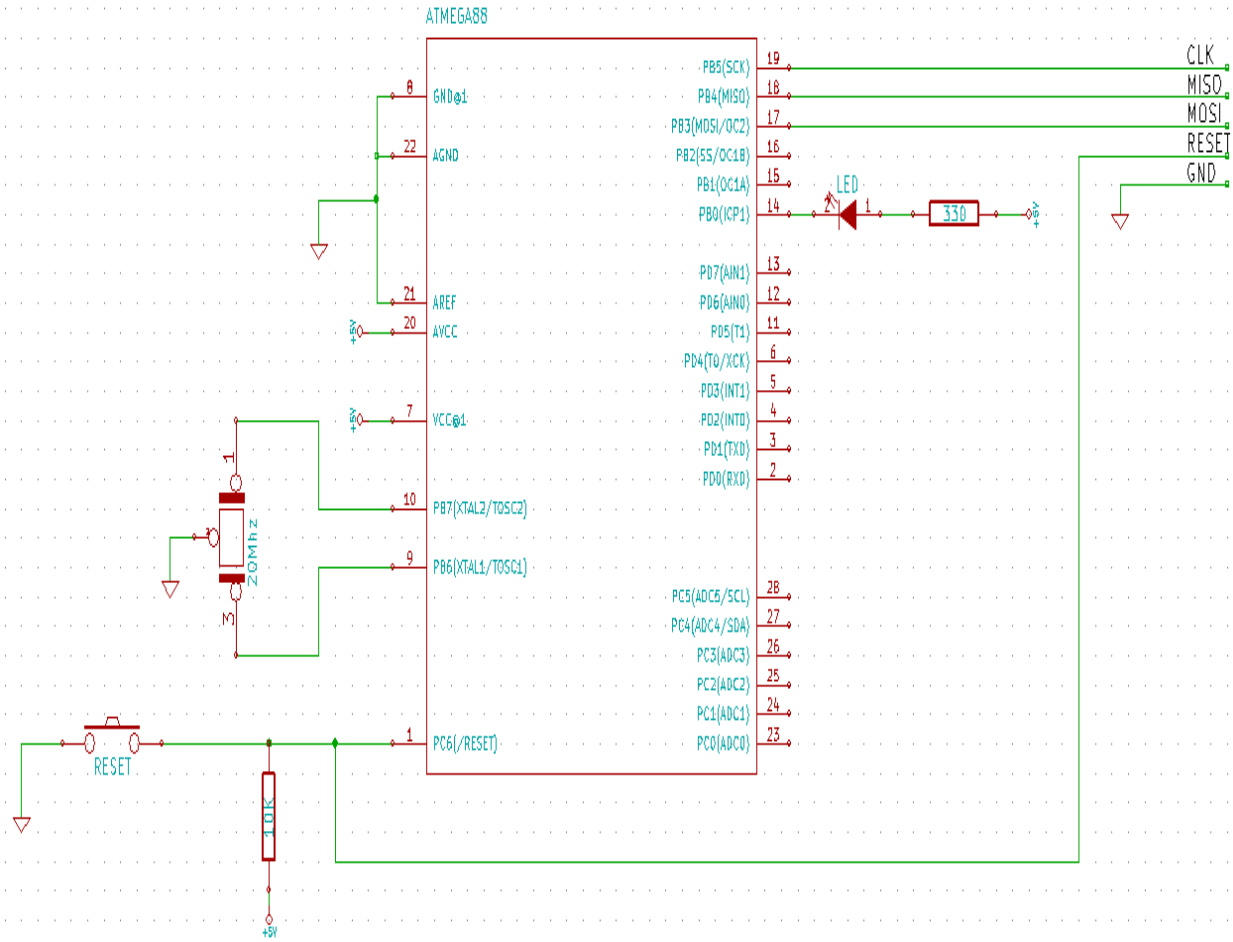

Figura 1. Conexiones para la programación del Atmega88

Además de las conexiones necesarias para el protocolo de programación ISP, se ha agregado un LED al pin 14 del microcontrolador. Esto se hizo para programar el microcontrolador para encender y apagar

intermitentemente dicho LED, con fines de comprobación de que la programación haya sido realizada correctamente.

5. Procedimiento para la programación del microcontrolador

5.1 Instalación de las herramientas de software

En esta guía se supone que el lector utiliza una versión de Ubuntu (o cualquiera de sus variantes) de 32 bits en una computadora personal, ya sea de escritorio o portátil.

Para la instalación del software necesario, se debe abrir el manejador de paquetes Synaptic, ya sea desde el menú o mediante la terminal escribiendo el siguiente comando:

```
Synaptic
```

Esta acción abre el gestor de paquetes, donde se pueden buscar los paquetes necesarios. Para buscar un paquete, escriba el nombre del mismo en la casilla que aparece en pantalla (figura 2).

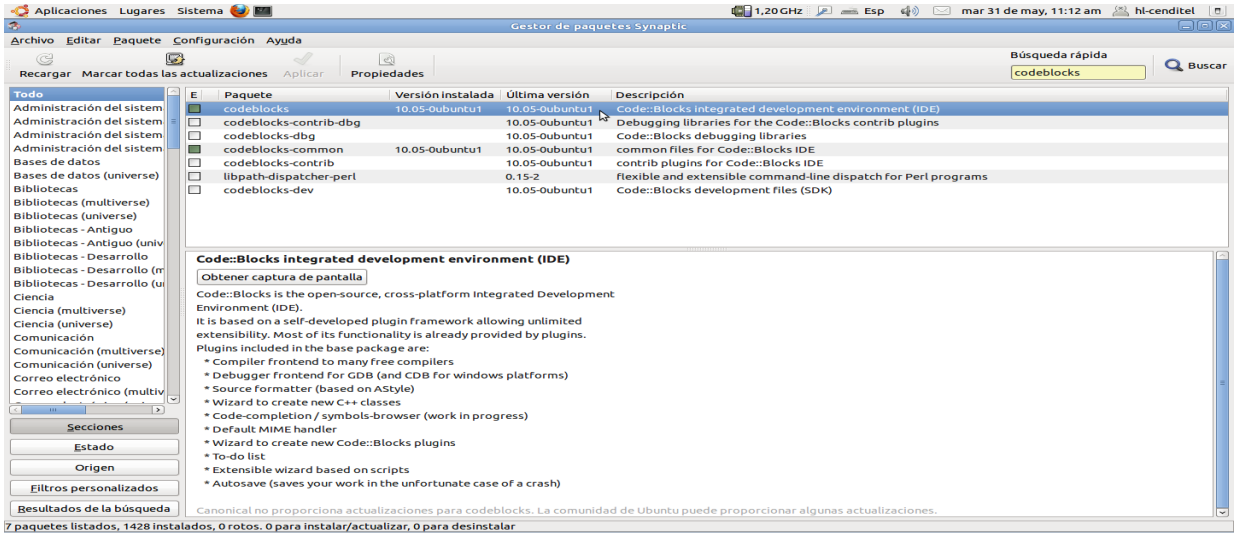

Figura 2. Ventana del gestor de paquetes Synaptic

El primer paquete a buscar es `codeblocks`. Una vez que haya sido procesada la consulta y encontrado el paquete, se procede a hacer clic derecho en el mismo y se selecciona “Marcar para instalación”.

Nota: si al hacer esto aparece **una ventana informando sobre otros paquetes necesarios para la instalación,** se hace clic **en “Aceptar”** en **esa ventana.**

Luego, se debe repetir este procedimiento para los siguientes paquetes:

- `Codeblocks-common`
- `Avrdude`
- `Avr-libc`
- `Gcc-avr`
- `Binutils-avr`

Una vez que se han marcado estos paquetes para su instalación, se hace clic en el botón “Aplicar” en la ventana principal de Synaptic (figura 3).

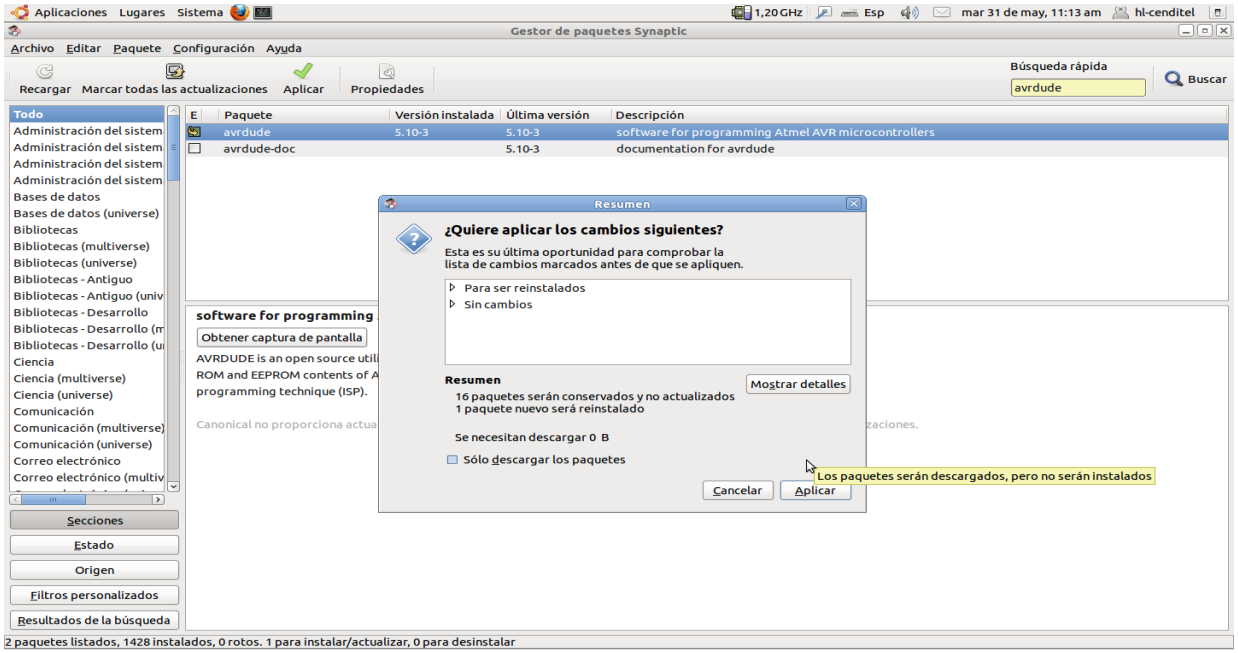

Figura 3. Ventana de aplicación de cambios

Esto inicia la descarga e instalación de los paquetes seleccionados.

Una vez finalizado el proceso, se le informa al usuario a través de una ventana de diálogo.

Con esto se finaliza la instalación del software y ya pueden ser utilizados el Code::Blocks y las herramientas para el proceso de programación del microcontrolador.

5.2 Codificación y generación del archivo hexadecimal (Utilizando el Code::Blocks)

El procedimiento para la programación de microcontroladores se puede dividir en dos fases. En la primera se debe escribir el código fuente en lenguaje C con las instrucciones que se desean ingresar al microcontrolador, para luego ser compilado y generar un archivo hexadecimal del mismo. Para esto es necesario una herramienta de edición y compilación de código. Al tener el fichero hexadecimal se procede a programarlo dentro del microcontrolador a través del uso de una interfaz de hardware conocido como programador, que sirve de vía de comunicación entre la computadora y el microcontrolador. Para realizar esta transferencia del archivo hexadecimal se utiliza otra herramienta en la cual, mediante comandos en una terminal, se le envían las instrucciones a la herramienta con los parámetros referentes al archivo hexadecimal, el microcontrolador, el puerto de comunicación y el programador hardware.

La herramienta que se utiliza en esta guía para la edición/compilación del código en C y la generación del código hex a partir de éste, se llama Code::Blocks, el cual es un IDE (Entorno Integrado de Desarrollo) que posee funcionalidades para proyectos de Atmel AVR.

Para inicializar el Code::Blocks, se busca en el menú Aplicaciones->Programación->Code::Blocks (figura 4).

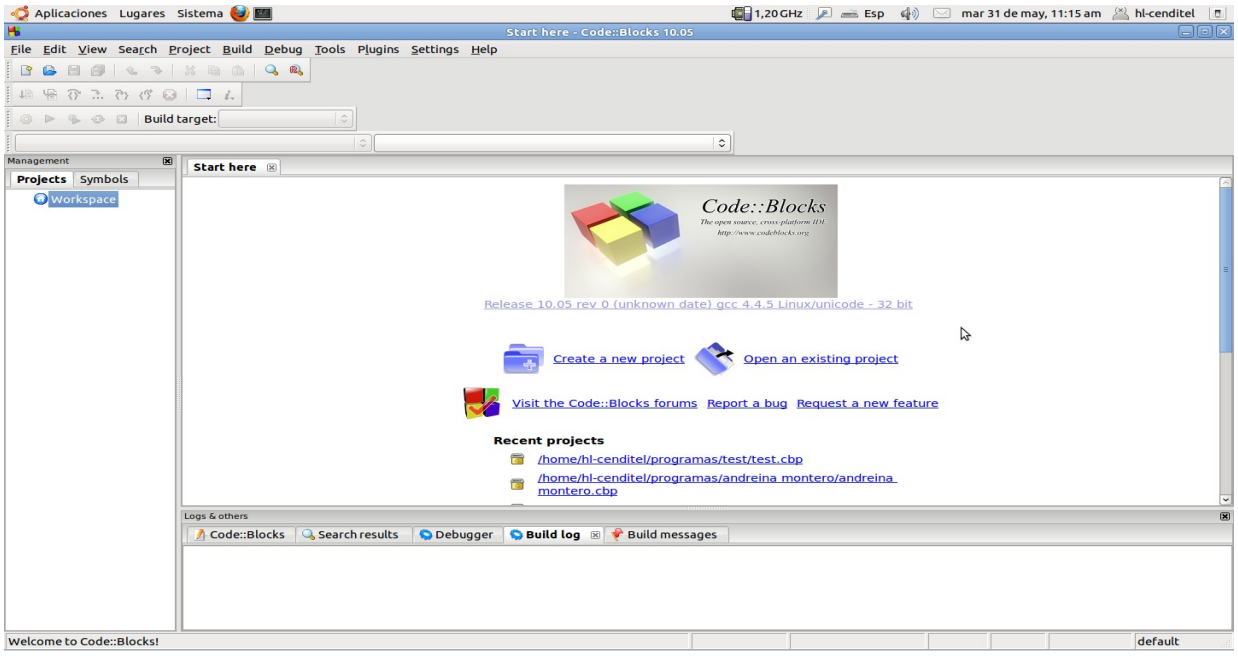

Figura 4. Ventana de inicio del Code::Blocks

También es posible abrir el IDE a través de un terminal con el comando:

```
Codeblocks
```

Ya con el Code:Blocks en ejecución, procedemos a ingresar al menú Settings->Compiler and Debugger. En este menú se debe colocar la opción Compiler en GNU AVR GCC Compiler.

Luego, se ubica la pestaña Toolchain Executables (figura 5), donde hay que confirmar que la información mostrada allí se encuentre de la siguiente forma:

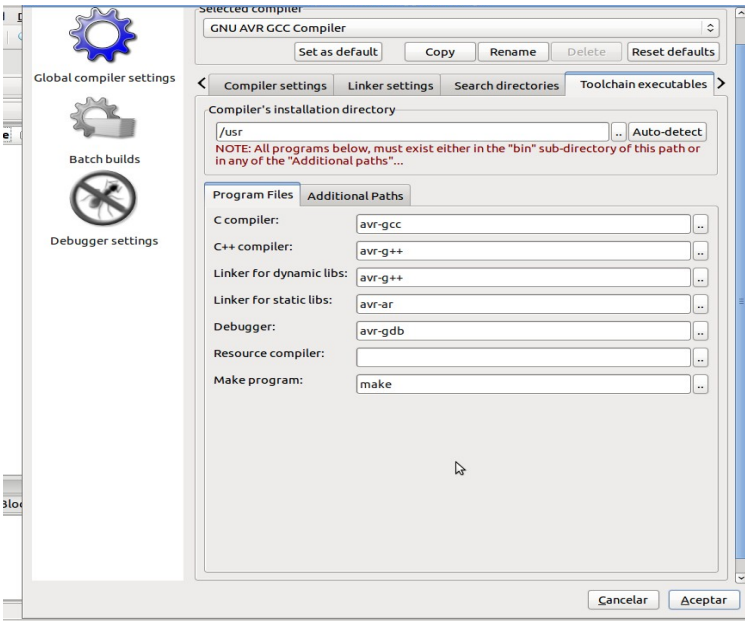

Figura 5. Configuración de Toolchain executables

Sin salir de esta ventana, se ubica la pestaña Debugger Settings del lado izquierdo, donde se desmarca la casilla Auto-build Project to ensure up-to-date y se hace clic en Aceptar (figura 6).

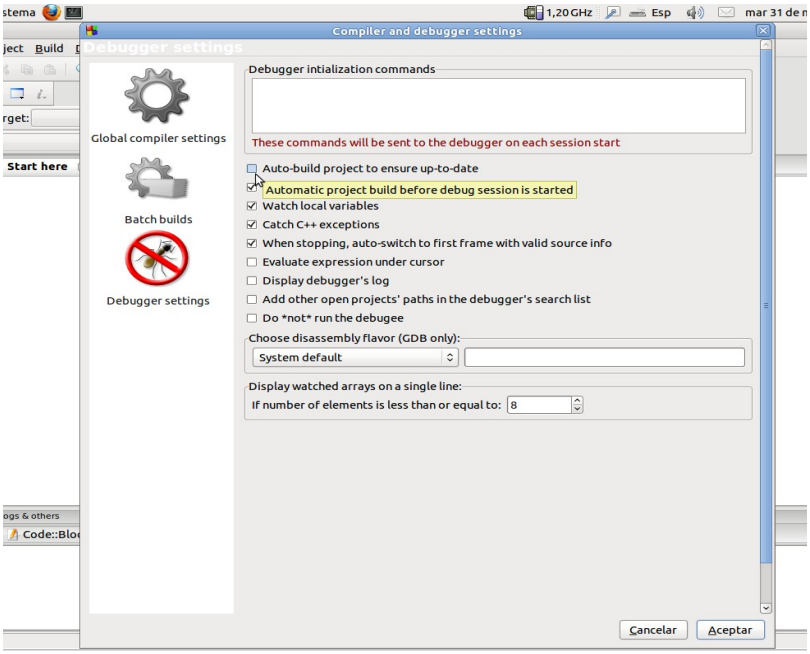

Figura 6. Configuraciones del Depurador

Ya realizadas las configuraciones que posibiliten la creación de ficheros hex a partir de la compilación de código en C para microcontroladores AVR, se puede proceder a crear un proyecto AVR.

Para crear un proyecto AVR, se ubica el menú File->New->Project (figura 7).

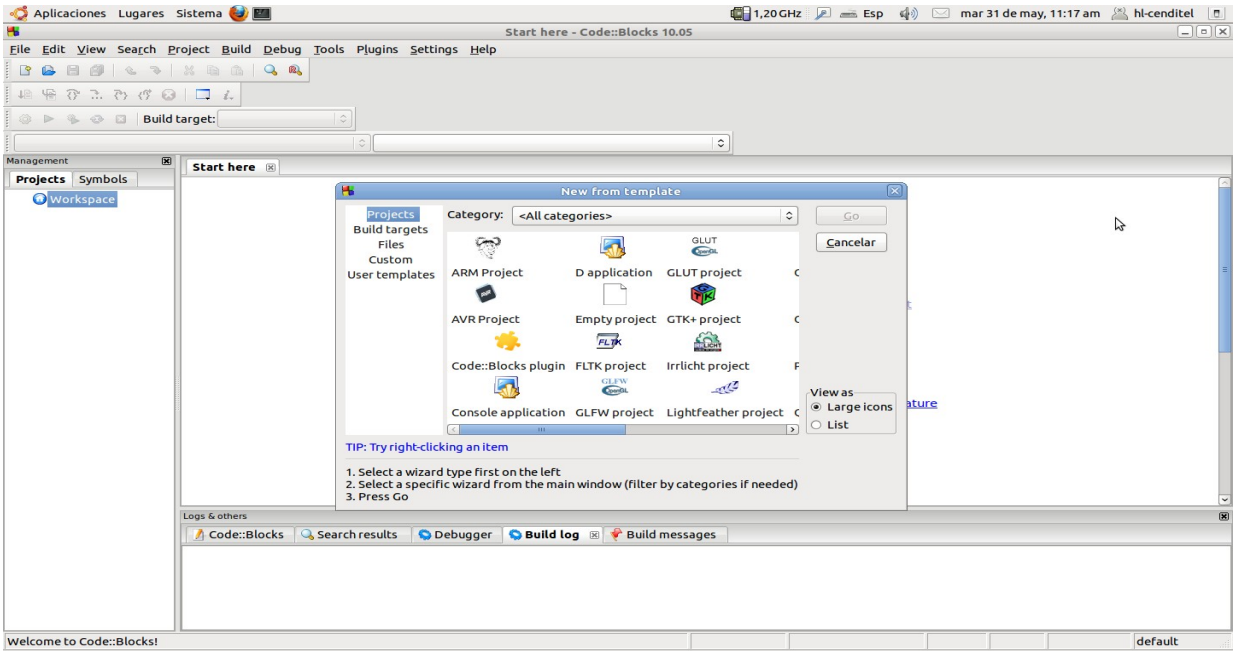

Figura 7. Creación de proyectos en Code::Blocks

Se seleccionamos AVR Project (figura 8)

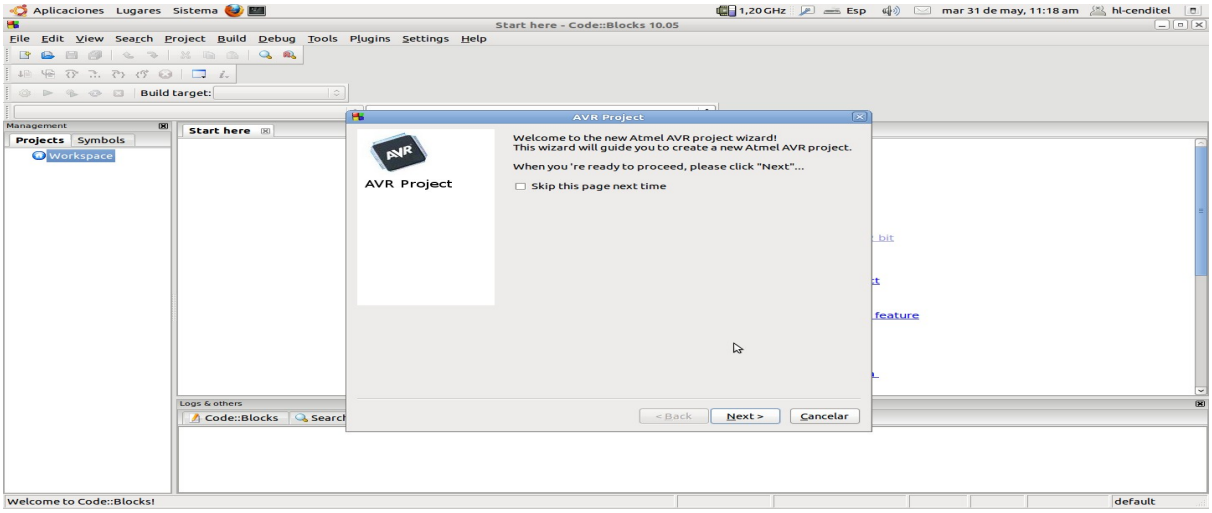

Figura 8. Proyecto AVR en Code::Blocks

Se procede a llenar la información referente al proyecto figura (9).

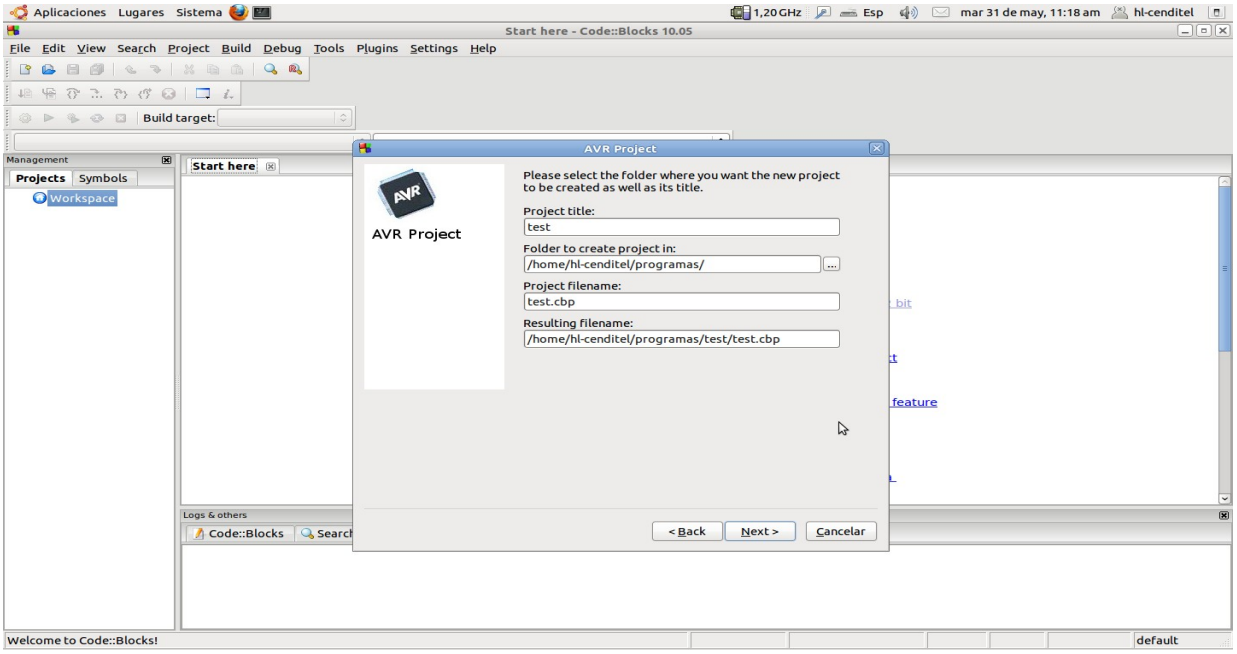

Figura 9. Información del proyecto AVR

En la pantalla siguiente se debe verificar que el compilador seleccionado sea GNU AVR GCC Compiler, y a la vez verificar que las opciones Create Debug y Create Release estén marcadas (figura 10).

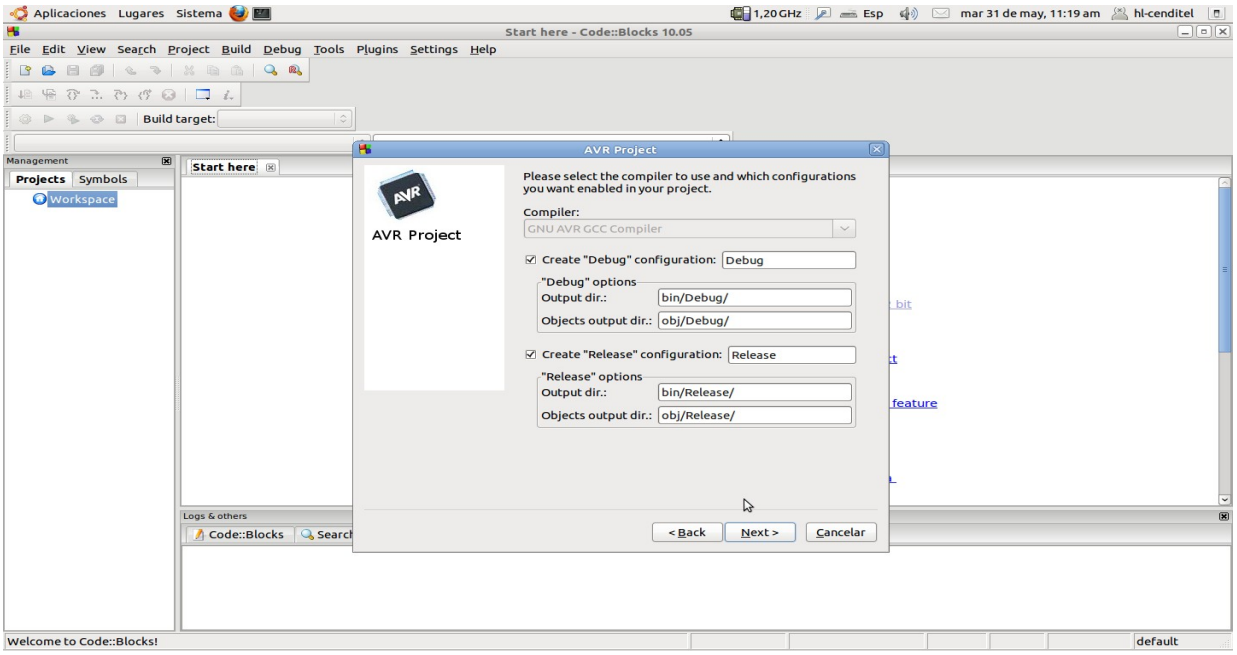

Figura 10. Opciones Create Debug y Create Release

Se selecciona de la lista el microcontrolador que se va a programar (ATMega88 en esta guía) y se marca la opción Run avr-size after build (figura 11).

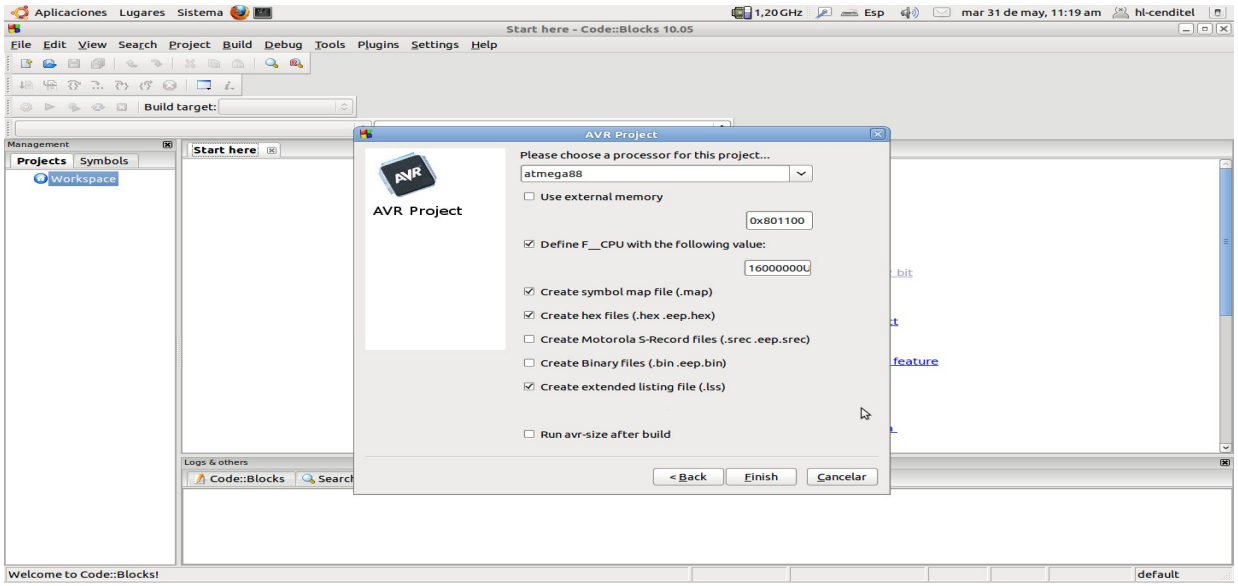

Figura 11. Selección del modelo del microcontrolador

Nota: Es importante destacar que, si en un futuro el lector desea trabajar con una frecuencia de reloj diferente a la predeterminada por el microcontrolador, es necesario modificar dicho campo con el valor correspondiente. Por ejemplo, si se desea trabajar con el ATMega88 a 8 Mhz, sustituimos 16000000UL por 8000000UL.

El IDE crea un proyecto nuevo con un árbol del proyecto de la siguiente forma (figura 12):

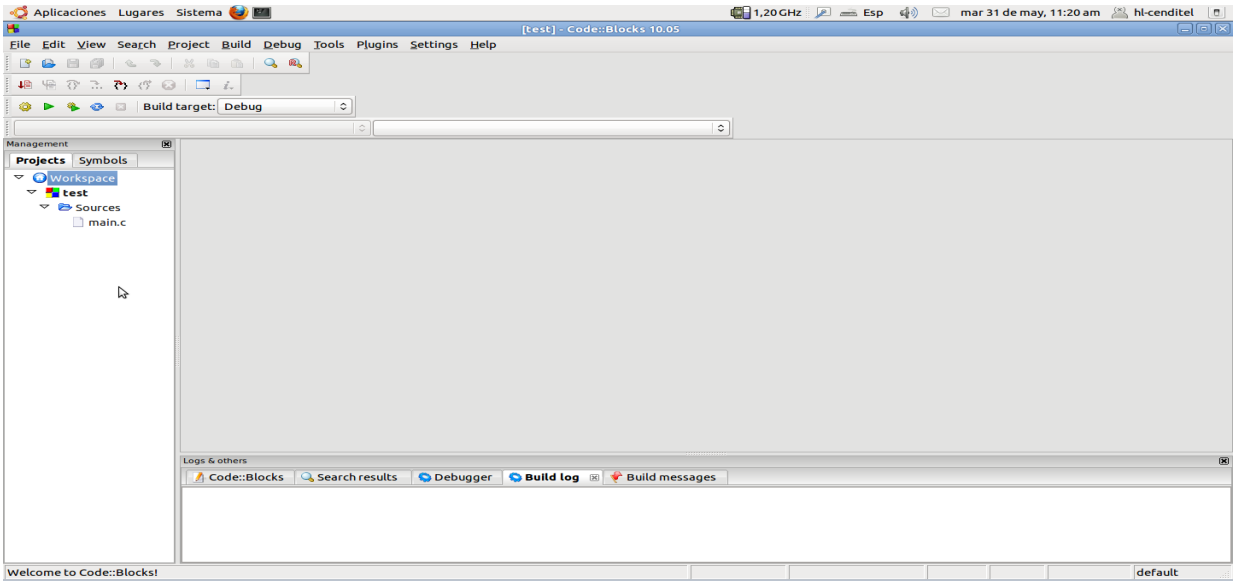

Figura 12. Proyecto AVR en blanco

Haciendo doble clic en el archivo main.c se muestran los contenidos del mismo, donde se va a escribir el siguiente código de ejemplo:

```
// Programa para controlar LED usando el ATMega88
#include<avr/io.h> //inclusión de la librería básica de entrada/salida
#include<util/delay.h>  //inclusión de la librería del delay
int main(void)   //función principal del programa
{
      DDRB=0xFF;  //Pines del puerto B del Atmega88 en alto
      while(1)   //Lazo de ejecución indefinida
      {
      PORTB=~PORTB;  //Cambia el estado de los pines del puerto B
                     //Si están en 0 pasan a 1, y viceversa
      _delay_ms(1000);   //Retraso de 1 segundo
      }
}
```

Una vez escrito, se compila el programa, haciendo clic en el ícono en forma de engranaje que se puede apreciar en la siguiente captura de pantalla (figura 13):

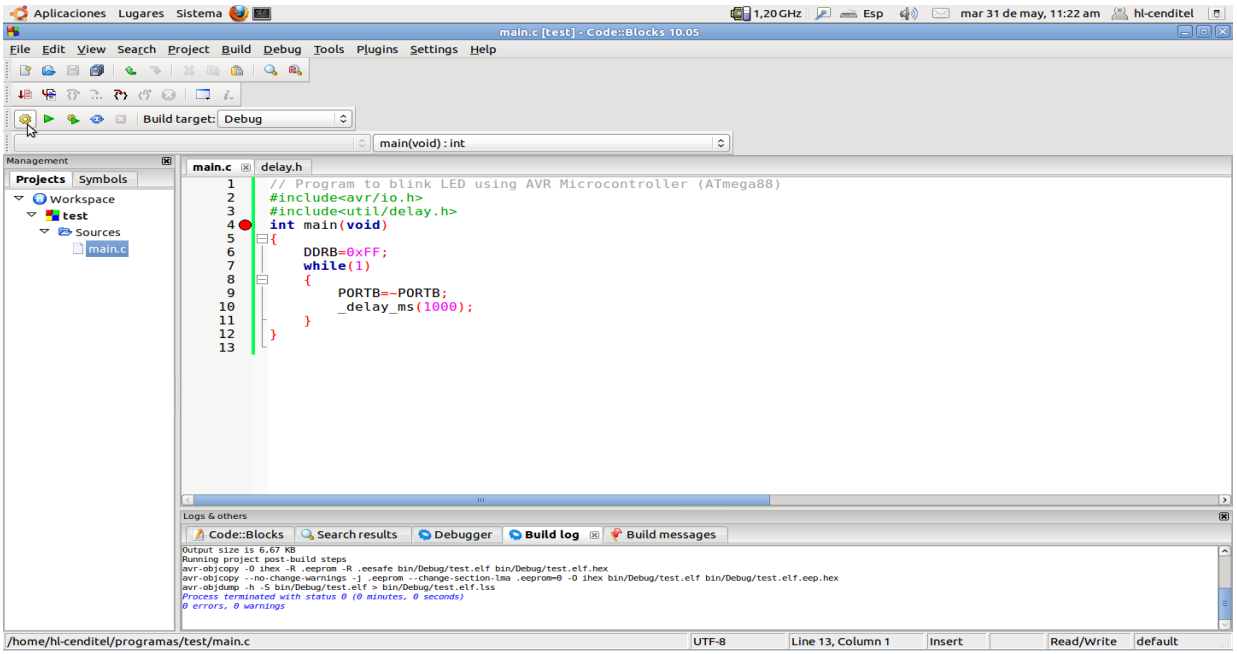

Figura 13. Compilación del código

Nota: Como se puede notar en la captura, la opción Build Target se encuentra Debug de manera predeterminada. Esta opción se utiliza mientras se comprueban errores en el código, pero cuando el código se encuentre totalmente depurado, se debe cambiar a Release para que generar el fichero hex.

Al hacer clic en el botón de compilar con la opción Build Target en Release, se genera un fichero con el formato *`nombre_del_proyecto`*`.elf.hex`. Este archivo está ubicado en la ruta donde fue creado el proyecto, dentro de la cual se encuentra una carpeta `Bin`, y dentro las carpetas `Debug` y `Release`; dentro de ésta última se ubica el hex.

Con un nombre de proyecto `test` creada en una carpeta `programas` la ruta sería de la siguiente forma:
`/programas/test/bin/release/test.elf.hex`

Con esto ya se tiene el archivo hexadecimal deseado. A continuación, solo nos falta realizar la subida de este fichero al microcontrolador.

5.3 Programación del microcontrolador (Utilizando AVRdude)

Para subir este fichero al microcontrolador, se utiliza un programa de línea de comandos llamado Avrdude.

Para esto, y tomando en cuenta que se utiliza un programador AVR-Prog, el cual es del tipo USBasp, y que se va a programar un microcontrolador ATMega88, se abre una terminal, ubicándose en la ruta donde fue creado el fichero hex y se escribe el siguiente comando:

```
sudo avrdude -p m88 -P usb -c usbasp -F -U flash:w:test.elf.hex
```

Nota: es importante que el comando anterior sea escrito por el usuario en una terminal, debido a que si se copia y pega es posible que la terminal no reconozca el guión corto correctamente. También debe ser **root** al ingresar el comando**, o** en su defecto **agregar** sudo **al inicio del** mismo **e ingresar** la **clave de superusuario. Esto s**e **deb**e **a que para la ejecución del Avrdude, se necesitan permisos de root.**

Si el proceso se realiza de forma correcta, se observa un mensaje como el siguiente (figura 14):

Figura 14. Proceso de subida con AVRdude

Cabe recordar que es necesario que el montaje del circuito sea correcto y que esté alimentado cuando se realice la subida del fichero al microcontrolador.

Nota: Los parámetros que se **pas**an **al Avrdude se construyen** con una sintaxis específica, **de acuerdo al microcontrolador y al programador que** se **utili**ce**n, de la siguiente forma:**

`-p` Nombre del microcontrolador

`-P` Puerto al que está conectado el programador

`-c` Modelo del programador

`-F` Comando para saltarse el chequeo de la firma del dispositivo

`-U` Tipo de Memoria donde se alojará el programa (`flash`) : `w` (Write-Escribir) : Nombre del fichero hex

Ya con esto se tiene el fichero dentro del microcontrolador. Basta comprobarlo observando el comportamiento del led conectado al microcontrolador, encendiendo y apagando durante intervalos de 1 segundo.