Universidad Autónoma de Baja California
Facultad de Ciencias Químicas e Ingeniería
Ingeniería en computación

MICROCONTROLADORES

Reporte de practica #2

Configuración del ambiente de desarrollo

GRUPO: 561

AUTOR: Rico López Tamara Illian (1270673)

MAESTRO: Ricardo Castro Gonzalez

21-Febrero-2021

Objetivo:

El alumno instalará el ambiente de desarrollo y se familiarizará con las herramientas.

Material:

- Computadora personal.
- Simulador en línea: https://wokwi.com/projects/new/esp32
- Analizador PulseView Sigrok

Teoría:

IDEs para sistemas embebidos

Para el desarrollo de sistemas embebidos es necesario un sistema de software que pueda controlar las funciones del hardware externo de la computadora. El desarrollo de sistemas embebidos necesita de diferentes herramientas, como un editor de texto, compilador, ensamblador, emulador, encadenador, y debugger; por ello, se suele hacer uso de IDE (*Integrated Development Environment*), los cuales incluyen las herramientas mencionadas anteriormente.

Uno de los principales IDEs es **PyCharm**, el cual pertenece a la compañía JetBrains y es utilizado especialmente para el desarrollo en Python, así como el soporte de otros lenguajes de programación como JavaScript, CoffeeScript, TypeScript, Cython, SQL, HTML/CSS, AngularJS, Node.js y otros. Asimismo, es una plataforma que puede usarse en sistemas operativos Windows, macOS y Linux.

Los beneficios de PyCharm es que implementa el reconocimiento de errores, completación inteligente de código, desarrollo remoto en máquinas virtuales y una gran comunidad que ayuda a solucionar problemas. Por el contrario, este IDE también presenta algunas desventajas, ya que puede ser lento, en especial si los proyectos crecen; así como las configuraciones preestablecidas suelen necesitar ajustes para el desarrollo de proyectos.

Otro de los IDEs más populares es **Qt Creator**, el cual cuenta con múltiples librerías, APIs y herramientas para crear software para sistemas embebidos en C + +, JavaScript y QML. Las ventajas de este IDE incluyen cross-compiling, autocompletación del código, señalización de los errores, 3D/2D interfaces de usuario; que colocan esta herramienta de software como líder en la industria del Internet of Things, aplicaciones móviles, industria médica y automatización. Sin embargo, esta plataforma también tiene desventajas, por ejemplo, la completación del código no siempre funciona, la plataforma es costosa y requiere de demasiado espacio de memoria.

Para el desarrollo en microcontroladores PIC, el mejor IDE es **MPLAB**, el cual pertenece a Microchip Technology, el cual es un software libre, que ayuda a la edición, debugging y programación de los sistemas embebidos. Esta plataforma incluye compiladores de C, funciones macros, interrupciones complejas, cambio de variables en la ventana de vista; así como facilidad de uso, autocompletación de código, revisión de errores de sintaxis, entre otros. Por otro lado, el IDE tiene algunas limitaciones, como el soporte único para Microchip que no soportan el desarrollo orientado a objetos.

Eclipse es un IDE que era utilizado únicamente para el desarrollo de aplicaciones en Java y, por ello, es la plataforma más popular de los programadores en dicho lenguaje. Sin embargo, Eclipse también soporta otro lenguaje de programación, tales como Ada, ABAP, C, C++, C#, Python y Php mediante el uso de extensiones. No obstante, su principal ventaja radica en un paquete especial (*Eclipse for Automotive Software Developers*) que contiene herramientas y frameworks para el fácil desarrollo de sistemas embebidos. Por otro lado, una desventaja de este IDE es que es difícil de organizar y de manejar y puede hacer que la memoria RAM se vuelva lenta pero la mayor desventaja es que suele ser complicada de utilizar en lenguajes diferentes a Java.

NetBeans es un IDE libre y gratis para el desarrollo de Java, la cual también incluye herramientas para Php y C/C + + que ayuda a crear aplicaciones con CSS, JavaScript y HTML. Las ventajas que ofrece esta plataforma incluye una gran selección de extensiones, soporte en diferentes plataformas, completación de código y que es gratis. Por otro lado, las desventajas incluyen la necesidad de instalar JDK, que es lento para iniciar y suele presentar problemas con la memoria caché al desarrollar programas.

Uno de los IDEs más utilizados para el desarrollo de programas para microcontroladores es **Arduino**, la cual es una plataforma que permite manejar los dispositivos de Arduino, provee un amplio rango de funcionalidades, librerías y apoyo en el desarrollo de sistemas embebidos. Las principales ventajas de este software son: librerías con códigos de ejemplo, es fácil de aprender a usar, es de código abierto con software y hardware extensible, soporte a través de plataformas Windows, MacOS y Linux, así como el apoyo de una extensa comunidad. Asimismo, una de las desventajas de esta plataforma es su falta de flexibilidad, ya que la plataforma está predefinida con ciertas funcionalidades y puede facilitar o dificultar el desarrollo de proyectos.

ARM Keil es una herramienta para desarrolladores que provee un entorno completo para la creación de aplicaciones basadas en sistemas embebidos para dispositivos ARM. El paquete de software incluye compiladores de C/C + +, simuladores, debuggers, encadenadores, etc. Sus ventajas son: fácil de aprender a usar, interfaz intuitiva, integración de herramientas externas, ejemplos de proyectos, plantillas de proyectos, soporte técnico de expertos en ARM, buena opción para expertos y principiantes.

Una de las herramientas populares para desarrollo es **Visual Studio** de Microsoft, ya que se usa para crear programas de computadora, aplicaciones móviles y sistemas embebidos. Las extensiones de C/C + + para el desarrollo de IoT (*Internet of Things*) permite a los programadores debug código de C/C + + en computadoras Windows, microcontroladores o máquinas virtuales en Linux. Esta plataforma incluye una versión gratuita, una terminal integrada, APIs para conectarse con otras herramientas, paquetes para desarrolladores que incluyen a integración de Git para el manejo de branches y la resolución de problemas, así como una extensiva lista de extensiones que permiten expandir las funcionalidades de este IDE. Sin embargo, entre sus desventajas se encuentra un alto precio por la versión profesional de esta plataforma, así como alto requerimientos de hardware y la falta de una versión para Linux.

Finalmente, la elección de un IDE depende de muchos factores, entre los que se encuentran el sistema operativo de la computadora, el lenguaje de programación a utilizar y la clase de plataformas que se desean desarrollar; así como de las habilidades del programador y preferencias, necesidades del proyecto y el hardware/equipo físico como el que se está trabajado.

Técnicas de depuración para sistemas embebidos

La depuración es una parte integral en el desarrollo de sistemas embebidos, igualmente importante que otras operaciones como el diseño. Un sistema embebido consiste en hardware, firmware y una aplicación de software; por lo que al presentarse problemas, es difícil identificar cuál es la causa y se crea la necesidad de técnicas de depuración efectivas.

Una de las técnicas más utilizadas para realizar esta depuración consiste en insertar código extra que **encienda uno de los LEDs** del microcontrolador, con el fin de determinar si una función o interrupción se está ejecutando correctamente. Asimismo, se puede encender y apagar el led en ciertas partes importantes del programa para poder identificar el progreso del código; sin embargo, esta técnica se vuelve menos efectiva al utilizarla en múltiples ocasiones ya que se puede confundir a que sección de código corresponde cada actividad.

Otra simple técnica de depuración es utilizar un **Pin del microcontrolador**, el cual se pone en set y reset en alguna seccion de codigo que se necesite monitorear, por lo que también se necesita d e un osciloscopio, contador de frecuencia p analizador para medir los intervalos de tiempo y entender lo que está sucediendo. Esta técnica tiene poco impacto en la velocidad del código y provee una tasa de repetición.

También es común utilizar un **LCD (*Liquid Crystal Display*)** el cual proporciona una forma de desplegar información así como salida de texto. Este módulo suele tener 2 filas de 16 caracteres y suelen usar el HD44780 controlador, lo que permite tener actualizaciones rápidas del estado del programa y proporciona una interfaz simple. Sin embargo, una desventaja es que este es un hardware externo y, por tanto, es más caro y ocupa 8 bits de los puertos del microcontrolador.

Asimismo, es posible utilizar un **ICE (*In Circuit Emulator*)** para depurar el hardware, el cual es la forma más costosa de realizar esta operación, ya que consisten en un procesador externo que toma el lugar del procesador preestablecido; este procesador especial permite el acceso del software al funcionamiento interno del procesador (puede establecer puntos de interrupción en los módulos de hardware).

Otra herramienta similar a un ICE es un **ICD (*In Circuit Debug*)** también llamado BDM (*Background Debug Mode*) el cual permite ejecutar el programa paso a paso en el procesador de destino. Para ICD, el procesador tiene una pequeña cantidad de hardware incorporado que puede detener el programa llega a una dirección específica. Luego, el software puede leer todos los registros y el estado del procesador.

Por otro lado, existen otros métodos para depurar microcontroladores que dependen del uso y el programa a utilizar; por ejemplo, la técnica más común es el uso de la función print/printf para imprimir ciertos datos durante la ejecución del programa, sin embargo, el código detrás de esta función requiere de muchos recursos del procesador, especialmente, si los datos de salida con strings. También es recomendable el uso de simuladores, el cual permite observar el efecto de los cambios y ver las operaciones internas en relación al programa; igualmente, se puede correr el código paso por paso o correr ciertas secciones del programa.

Desarrollo:

Instalación del entorno de desarrollo

Para poder desarrollar código y programas para microcontroladores, se hará uso del editor de texto VS Code, el cual cuenta con una extensión útil llamada PlataformIO. Esta extensión es una alternativa al IDE arduino, ya que permite trabajar el proyectos avanzados con más de 200 líneas de código, con múltiples archivos así como funciones avanzadas como la corrección de errores [7].

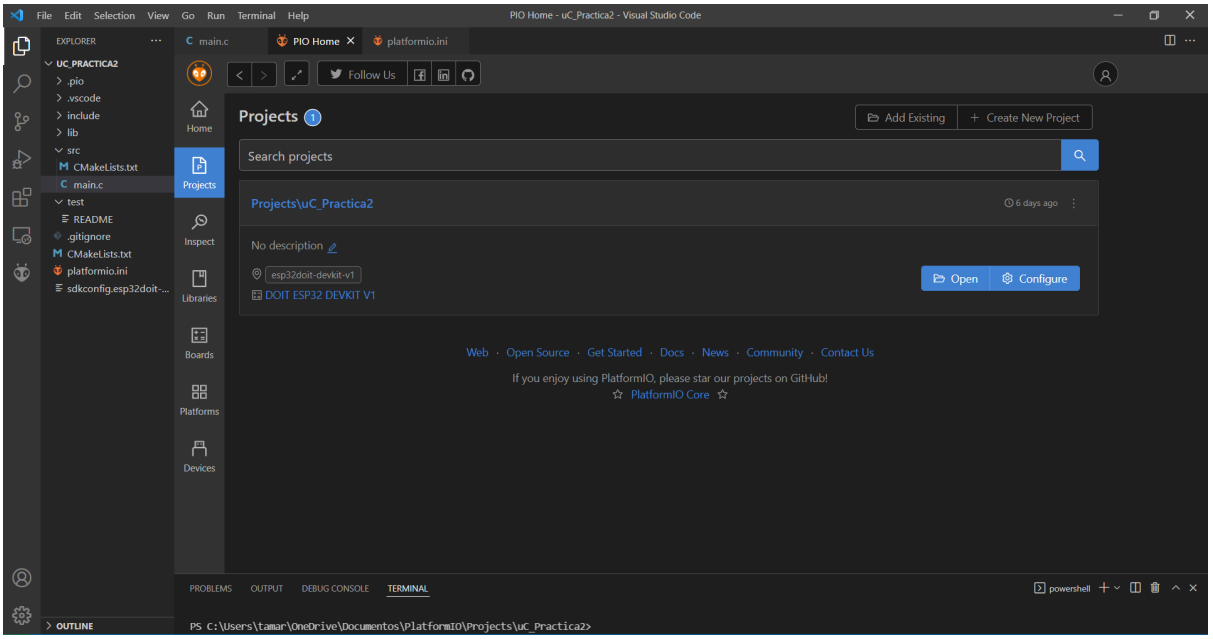

***Figura 1**. Captura de pantalla de PlataformIO.*

El desarrollo dentro de esta plataforma se realizará utilizando el microcontrolador ESP32 así como el Framework de Espressif IoT Development Framework. Este software nos permite compilar el código fuente. En la figura 1 se muestra un ejemplo de la estructura del IDE así como la creación del primer proyecto de trabajo.

Sin embargo, debido a que todavía no se contaba con el microcontrolador de forma física, se utilizó el simulador Wokwi; el cual permite simular proyectos enfocados a IoT (*Internet of Things*) en el navegador.

Desarrollo de la práctica

El desarrollo de la práctica consistió en hacer que uno de los leds del microcontrolador produjera mi nombre es código morse. El código morse es un sistema de representación de números y letras mediante señales emitidas de forma intermitente; que se dividen en combinaciones de puntos y guiones. En esta ocasión, los puntos son considerados un periodo de tiempo que dura el led encendido, mientras que los guiones representan los leds encendidos por un periodo de tiempo tres veces mayor que los puntos; por otro lado, las letras deben tener una separación equivalente a un guión.

Para la realización del código, se tomó en cuenta el ejemplo visto en clase para el manejo de los leds de un ESP32, con funciones como retrasos, parpadeo de un led, y configuración de un led. Asimismo, se tomó como referencia un programa que manejaba el código morse en arduino, donde se usa un arreglo de cadenas de caracteres para representar las letras en su equivalente en morse [1].

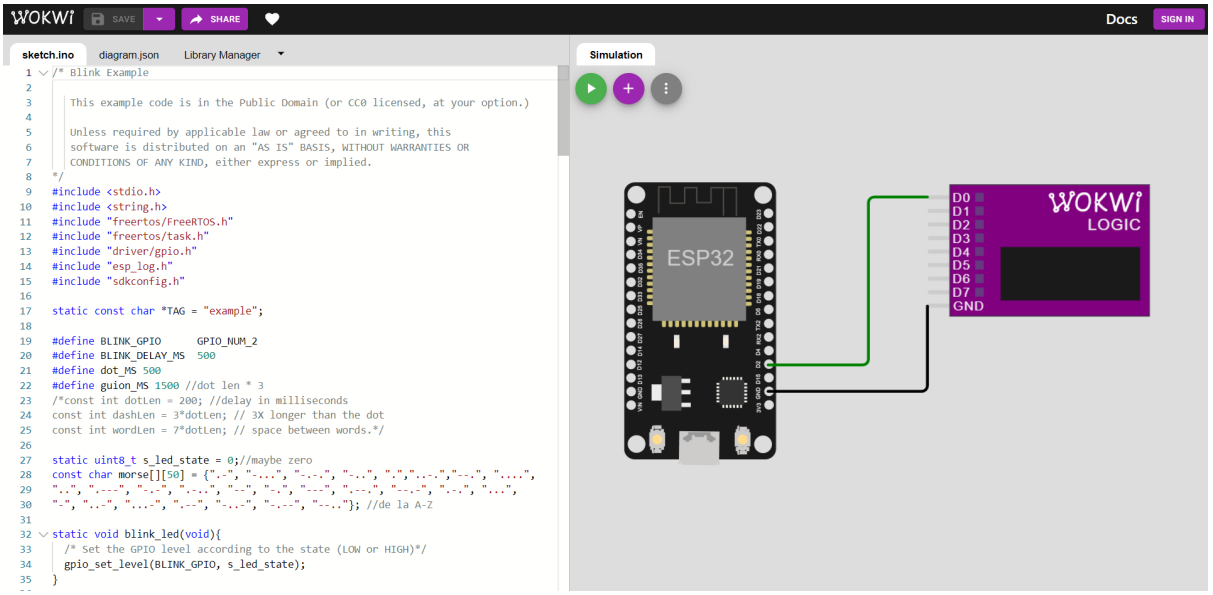

Figura 2. Captura de pantalla del simulador Wokwi.

A falta del microcontrolador físico, se trabajó con el simulador Wokwi Fig. 2, donde se encuentra con un Analizador lógico, el cual registraba los pulsos producidos por el dispositivo en relación al programa. El código utilizado para producir el resultado esperado fue realizado el lenguaje de programación C, así como la inclusión múltiples librerías utilizadas en el desarrollo de programas para ESP32.

Resultados

Como se mencionó anteriormente, el simulador Wokwi cuenta con un analizador lógico, el cual registra los pulsos, causados por los leds, y genera un archivo .vcd. Este archivo .vcd es usado por la herramienta de software Pulse View para mostrar la gráfica resultante; asimismo, este software cuenta con un decodificador, el cual incluye el código morse, por lo que se utiliza para decodificar el resultado de la simulación. Como se muestra en la figura 4, el resultado del analizador corresponde al nombre especificado.

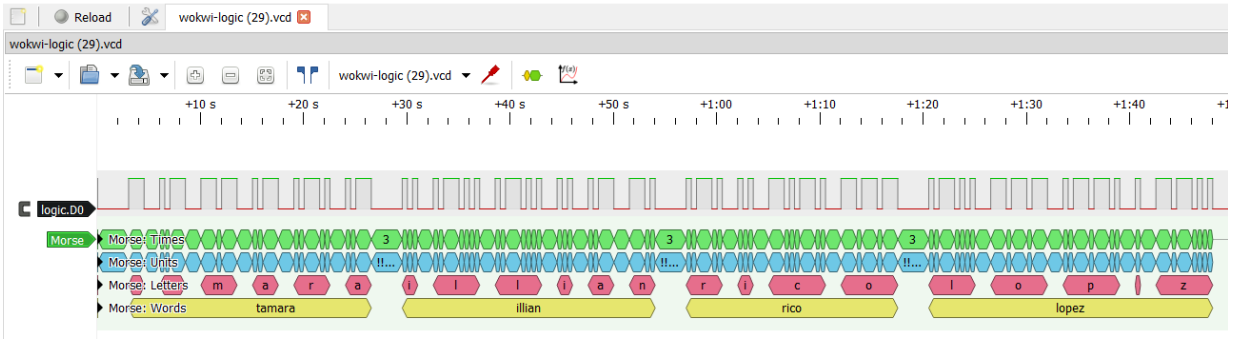

Figura 3. Decodificación de la gráfica en Pulse View.

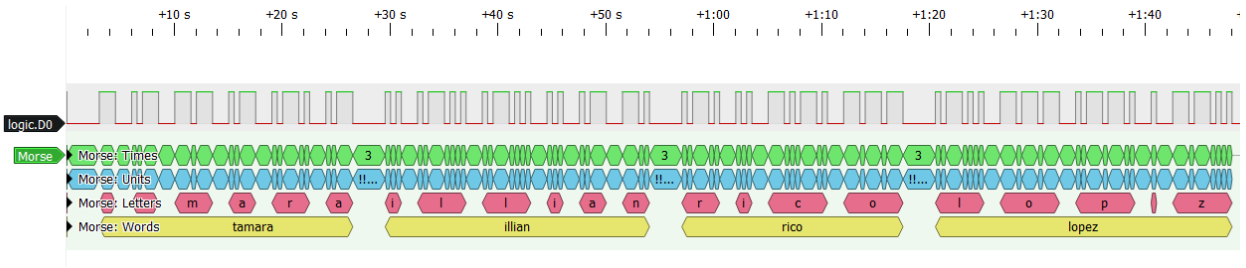

Figura 4. Código morse correspondiente a "Tamara Illian Rico Lopez".

Conclusiones y comentarios:

El desarrollo de la práctica me permitió aprender a trabajar con la programación en C de forma distinta, al tener que ajustar el código a los requerimientos del microcontrolador. Asimismo, la práctica presentó una oportunidad para poder hacer un código, probar el programa con un simulador a falta del dispositivo físico, así como el analizar la salida de los leds mediante el uso de un analizador de señales. Por otro lado, la práctica fue un buen ejercicio para practicar la depuración del código de los microcontroladores y el uso del hardware como los leds y los puertos; así como el análisis del código morse.

En conclusión, la práctica representó una buena oportunidad para practicar los conocimientos adquiridos en clase, así como para superar algunas dificultades al tener que trabajar con el lenguaje C de forma diferente y no poder imprimir datos, así como el aprender la estructura del código morse.

Referencias:

[1] Morse, "Let's Start Coding," *Let's Start Coding*, 2019. https://www.letsstartcoding.com/morse-code-with-sound-and-light (accessed Feb. 21, 2022).

[2] "Top 10 Tools for Embedded Development [Ultimate Guide] | SaM Solutions," SaM Solutions, May 22, 2019. https://www.sam-solutions.com/blog/top-ten-embedded-software-development-tools/ (accessed Feb. 21, 2022).

[3] "Embedded Software Development Tools: Research Of The Market And Trends," Sirin Software, Jun. 29, 2021. https://sirinsoftware.com/blog/embedded-software-development-tools-research-of-the-market-and-trends/ (accessed Feb. 21, 2022).

[4] https://www.facebook.com/theengineeringprojects, "Embedded Systems Software Development Tools - The Engineering Projects," The Engineering Projects, Nov. 17, 2016. https://www.theengineeringprojects.com/2016/11/top-10-embedded-systems-software-development-tools.html (accessed Feb. 21, 2022).

[5] "Simple techniques for debugging microcontroller hardware.," Best Microcontroller Projects, 2022. https://www.best-microcontroller-projects.com/article-debugging.html (accessed Feb. 21, 2022).

[6] Elektormagazine, “The Full Gamut of Microcontroller Debugging Techniques,” Elektor, Jun. 30, 2020. https://www.elektormagazine.com/articles/microcontroller-debugging-techniques (accessed Feb. 21, 2022).

[7] “Getting Started with VS Code and PlatformIO IDE for ESP32 and ESP8266 | Random Nerd Tutorials,” Random Nerd Tutorials, Sep. 24, 2020. https://randomnerdtutorials.com/vs-code-platformio-ide-esp32-esp8266-arduino/ (accessed Feb. 23, 2022).