Práctica No.2 Ensamblador, Interrupciones Temporizadores. Principios de Mecatrónica

Arcadio Alexis Calvillo Madrid 159702
Arlet Díaz Méndez 154840

Resumen—En esta práctica del laboratorio de Principios de mecatrónica se llevaron a cabo tres procesos importantes en el desarrollo de sistemas embebidos: la programación en lenguaje ensamblador, el uso de interrupciones y la programación de temporizadores en dos modos distintos. Estas soluciones permiten un mejor acercamiento al funcionamiento del Arduino, en particular del microcontrolador ATmega2560.

1. INTRODUCCIÓN

Los objetivos planteados para la realización de esta práctica son: programar un microcontrolador utilizando ensamblador, comparar el lenguaje de bajo nivel contra el de alto nivel, aprender a utilizar las interrupciones y finalmente aprender a configurar y utilizar los temporizadores del ATmega2560. Esto contribuye al objetivo general del laboratorio de Principios de Mecatrónica de realizar un sistema embebido. Para el desarrollo de esta primera práctica se tomaron tres sesiones, en la primera se desarrollaron soluciones en lenguaje ensamblador y en C, para poder compararlos en distintos aspectos. En la segunda sesión se desarrollaron programas en que ejemplifican la utilización de las interrupciones. Por último, en la última sesión se configuraron los temporizadores para mostrar sus distintos usos. Para el desarrollo de esta práctica se utilizó un Arduino Mega 2560 y el IDE de Arduino. La programación en lenguaje ensamblador es posible en este IDE dado que el compilador para procesadores AVR permite la opción de introducir código ensamblador directamente en código C. La finalidad de esta posibilidad es poder introducir comandos que no se encuentran disponibles en C u optimizar partes críticas del código.

Este documento tiene la siguiente organización: un Marco Teórico en el que se explican las tecnologías utilizadas, así como los conocimientos teóricos que las respaldan, un Desarrollo en el que se explica cómo contribuyen los componentes a la solución, una sección de Resultados en la que se habla de los valores obtenidos durante la práctica, luego se presentan las Conclusiones de la práctica, los Roles de cada integrante del equipo y las Referencias utilizadas durante el proceso.

2. MARCO TEÓRICO

Arduino Mega Es una placa de Arduino (compañía de desarrollo abierto de hardware)que dispone de un microcontrolador AVR Atmel de 8 bits. Se considera la placa más potente de Arduino debido a que es el que más pines i/o tiene, es apto para trabajos más complejos que el que soportan otras placas. El Arduino Mega usa el microcontrolador Atmega2560.

IDE Arduino El entorno de desarrollo integrado de Arduino es una aplicación disponible para varios sistemas operativos desarrollado en Java que se utiliza para escribir y cargar programas en las placas Arduino. Este IDE permite el uso de los lenguajes C y C++ pero se utilizan reglas especiales de estructura de códigos.

Lenguaje C Es un lenguaje de programación imperativo, de propósito general, que admite programación estructurada, alcance de variable léxica y recursión, lo que se define como lenguaje de alto nivel. Sin embargo, cuenta con un sistema de tipo estático que evita muchas operaciones no intencionadas en bajo nivel. Lo anterior se debe a que es un lenguaje orientado a la implementación del sistema operativo Unix.

Lenguaje ensambladorEs un lenguaje de programación de bajo nivel y es la representación más próxima al código máquina. Consta de un conjunto de mnemónicos que efectúan las operaciones básicas en microprocesadores y microcontroladores operando directamente sobre las direcciones de memoria. Es definido por los proveedores de hardware, por lo que no es portable como los lenguajes de alto nivel, sino que depende de la arquitectura de cada dispositivo.

3. DESARROLLO

La práctica constó de tres sesiones. Durante la primera sesión, se atacaron tres problemas: el cálculo del discriminante de una ecuación cuadrática, el cálculo del promedio y el uso de entradas y salidas del Arduino. Estos tres problemas se resolvieron primero escribiendo un programa en C y después en lenguaje ensamblador. Luego, se comparó el lenguaje ensamblador escrito por el equipo con el código ensamblador compilador, que es el código ensamblador

generado por el compilador a partir del programa en C. En la siguiente imágene se muestra el programa que ejemplifica el uso de entradas y salidas utilizando como actuador un led.

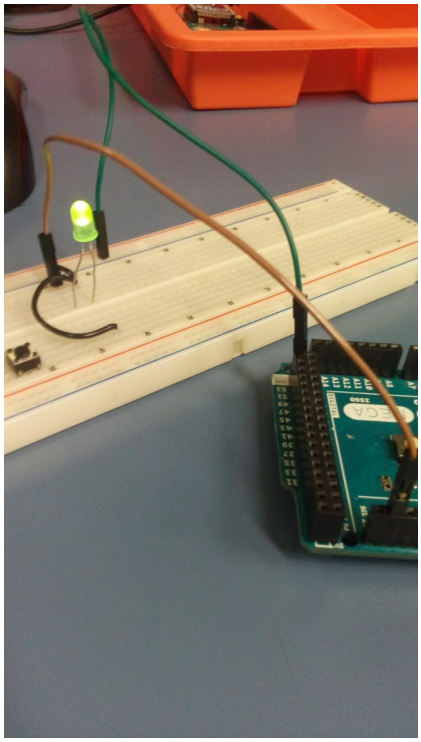

Fig 1: Cambio de estado del led con un botón

En la segunda sesión, se realizaron interrupciones, primero con la función delay y después mediante la recepción (input) de una señal externa, esto a través de un botón que incrementa el contador cada vez que se presionaba. Después se cambió el botón por un led infrarrojo y un fotodiodo que fungieron como un sensor infrarrojo, esta vez el incremento del contador se realizaba ante un cambio en el estado del fotodiodo. A continuación, se asoció la solución anterior a una interrupción. A continuación se muestra la implementación de ambos sistemas.

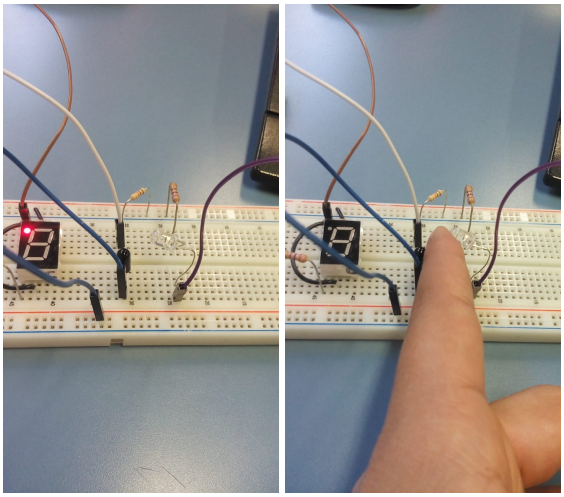

Fig 2: Interrupciones mediante sensores

En la tercera sesión, se realizó la configuración de los temporizadores, primero en modo normal a 2 Hz y luego en CTC a 4Hz. En ambos experimentos el objetivo fue hacer parpadear un led a la frecuencia antes mencionada para cada caso. Por último, se implementó un semáforo mediante un timer CTC en el cual el led rojo encendía durante 10s, el verde durante 15s y el amarillo durante los últimos 3s del verde. Las siguientes imágenes ilustran el funcionamiento del semáforo.

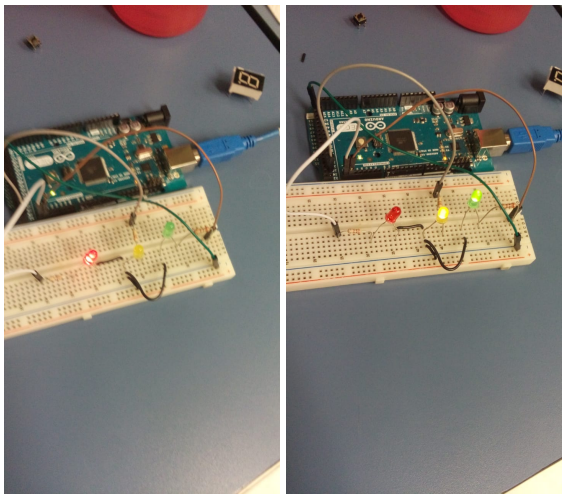

Fig 3: Semáforo implementado en Arduino

4. RESULTADOS

Los resultados obtenidos durante la práctica fueron satisfactorios. Sin embargo, fue necesario invertir más tiempo que el de la duración de las tres sesiones para lograr el funcionamiento de todos los ejercicios.

5. CONCLUSIONES

Arlet: Aunque nunca había utilizado el lenguaje C, me fue muy fácil utilizarlo pues se parece mucho a otros lenguajes de alto nivel que ya había utilizado con anterioridad. Contrariamente, la programación del lenguaje ensamblador presentó muchas dificultades para mí.

Alexis: Existe una gran diferencia entre el código generado por C y el código que se necesita para hacer la misma tarea, pero programada en ensamblador y aunque esta vez no importe mucho en programas que necesiten bastante memoria esto es un factor importante. Además, es un poco más difícil programar en ensamblador.

Otro de los aspectos importantes es el uso de interrupciones y timers. Por un lado las interrupciones nos pueden ayudar cuando ya tenemos todos los puertos ocupados y así cambiarlos de entrada a salida; o de analógico a digital. Además podríamos necesitar interrumpir una acción para hacer otra antes. Por otro lado los timers nos permiten sincronizar tareas para un correcto funcionamiento, cabe recalcar que el uso de timers puede manejarse con interrupciones.

6. ROL O PAPEL

Arlet: Programar algunas partes del código tanto en C como en ensamblador. Actualizar las versiones de la práctica en GitHub. Redacción y corrección del reporte escrito.

Alexis: Programar algunas partes del código tanto en C como en ensamblador. armar el circuito electrónico. Redacción y corrección del reporte escrito.

7. FUENTES CONSULTADAS

[1] Wordpress. Aprendiendo Arduino. [Online]. Available: https://aprendiendoarduino.wordpress.com/tag/arduino-mega/

[2] ATMEL. Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V Datasheet. [En línea]. Disponible en: http://ww1.microchip.com/downloads/en/devicedoc/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561$_datasheet.pdf$

[3] Wikipedia. C (Lenguaje de programación.) [En línea]. Available: https://es.wikipedia.org/wiki/C$_(lenguaje_de_programación)$

[4] EcuRed. Lenguaje Ensamblador [En línea]. Disponible en: https://www.ecured.cu/Lenguaje$_ensamblador$