UNIVERSIDAD PROVINCIAL DEL SUDOESTE

FACULTAD DE LA MICRO, PEQUEÑA

Y MEDIANA EMPRESA

CARRERA DE TECNICATURA UNIVERSITARIA EN

TECNOLOGÍAS DE PROGRAMACIÓN SEDE PUNTA ALTA

TRABAJO FINAL

Autores:

Dolores Ponce

Gaston Ponce

Lorena Quipildor

Gisela Yede

Rodrigo Zamora

Docente:

Ing. Ronny Stalin Guevara Cruz

Punta Alta – Argentina

2023

CONTENIDO

RESUMEN

El proyecto se centra en el desarrollo de un reloj digital basado en un módulo GPS (Ublox NEO-6M) y un display de 4 dígitos de 7 segmentos (TM1637) utilizando una placa Arduino Uno. El sistema tiene como objetivo principal mejorar la precisión temporal al eliminar las desviaciones inherentes a los osciladores internos de los relojes convencionales. La implementación se lleva a cabo mediante la configuración física de los componentes y la programación adecuada en Arduino. La simulación en Proteus valida la eficacia del diseño, ofreciendo una representación visual de la sincronización y visualización de la hora.

INTRODUCCIÓN

La medición precisa del tiempo ha sido una necesidad constante a lo largo de la historia, impulsando la evolución de relojes y dispositivos cada vez más precisos. En la actualidad digital, los relojes electrónicos comunes han alcanzado notables niveles de precisión, pero persisten desafíos asociados a la estabilidad de sus osciladores internos y las variaciones ambientales. Estas pequeñas desviaciones, aparentemente insignificantes a corto plazo, pueden acumularse y afectar la exactitud del tiempo con el paso del tiempo.

En este contexto, la implementación de un reloj digital basado en un módulo GPS surge como una solución innovadora y altamente precisa. A diferencia de los relojes convencionales que dependen de osciladores internos propensos a derivas y fluctuaciones, un reloj alimentado por un módulo GPS utiliza señales satelitales para obtener la hora exacta, eliminando las imprecisiones asociadas y permitiendo una sincronización temporal sin precedentes.

Ventajas del Reloj Basado en Módulo GPS:

1. Precisión Atómica: Los módulos GPS reciben señales de relojes atómicos en el espacio, ofreciendo una referencia de tiempo extremadamente precisa. Esta precisión garantiza una sincronización sin igual, minimizando cualquier desviación temporal.

2. Corrección Automática de Zona Horaria: Los relojes basados en GPS ajustan automáticamente la zona horaria según la ubicación geográfica, evitando errores manuales y asegurando una representación exacta de la hora local.

3. Resistencia a Desviaciones Ambientales: Mientras los osciladores internos de los relojes convencionales pueden ser afectados por factores ambientales, los módulos GPS, al obtener la hora de señales externas, son menos propensos a desviaciones causadas por condiciones adversas.

4. Sincronización Continua: La constante comunicación con satélites permite que el reloj GPS se sincronice continuamente, corrigiendo cualquier desviación que pueda surgir con el tiempo y garantizando precisión a largo plazo.

5. Utilización Global: Al recibir señales de satélites en el espacio, los relojes basados en GPS ofrecen precisión global, independientemente de la ubicación geográfica.

En conclusión, la adopción de un reloj digital basado en un módulo GPS representa una mejora significativa en la precisión temporal. Esta tecnología no solo aborda los desafíos asociados con la estabilidad de los osciladores internos, sino que también ofrece funcionalidades avanzadas, como la corrección automática de zona horaria y la resistencia a desviaciones ambientales, estableciéndose como una solución integral para las necesidades de medición del tiempo exacto en diversos entornos.

CAPÍTULO I

Objetivos

Objetivo general

Desarrollar un reloj digital basado en un módulo GPS y un display de 4 dígitos de 7 segmentos utilizando una placa Arduino Uno.

Objetivos específicos

1. Configurar la comunicación entre el módulo GPS y la placa Arduino Uno.
2. Programar la lectura y procesamiento de la hora desde el módulo GPS.
3. Implementar la visualización precisa de la hora en el display de 4 dígitos.

CAPÍTULO II

Marco teórico

Arduino Uno

Principio de Funcionamiento:

La Figura 1 muestra el diseño de la Placa Arduino Uno y sus componentes principales. Como se puede observar, el Arduino Uno es una plataforma de desarrollo de código abierto basada en microcontroladores ATmega328. Diseñada para la programación y prototipado de proyectos electrónicos, el Arduino Uno facilita la interacción entre hardware y software. Sus pines de entrada/salida, capacidad para conectar periféricos y su entorno de desarrollo integrado (IDE) la convierten en una elección popular para una variedad de aplicaciones.

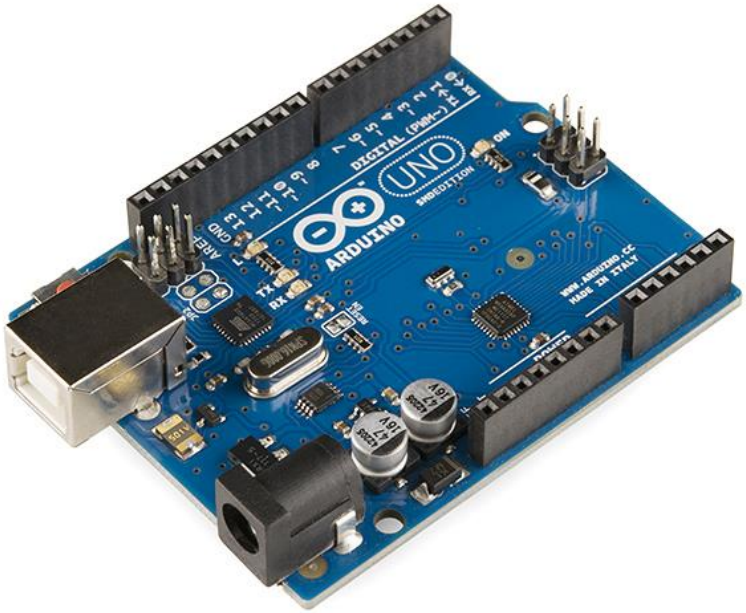

Figura 1. Placa Arduino Uno [1]

[1] SparkFun Electronics. (21 de enero de 2013). Arduino Uno - R3. Fuente: SparkFun Electronics. Recuperado de https://www.flickr.com/photos/41898857@N04/8406865680

Display de 7 Segmentos y 4 Dígitos (TM1637)

Principio de Funcionamiento:

El Display de 7 Segmentos y 4 Dígitos basado en el controlador TM1637 (Figura 2) es un componente que simplifica la visualización de información numérica. Utiliza la técnica de multiplexación para activar secuencialmente cada dígito, permitiendo representar números y letras mediante segmentos individuales. Su interfaz de comunicación serial facilita la conexión y el control eficiente desde una placa Arduino.

Figura 3. Módulo Display 7 Segmentos 4 Dígitos basado en TM1637 [2]

[2] Yoyojacky. (27 de marzo de 2020). Página del producto D-0007. Fuente: Wiki 52pi. Recuperado de https://wiki.52pi.com/index.php?title=D-0007

Módulo GPS Ublox NEO-6M

Principio de Funcionamiento:

El Módulo GPS Ublox NEO-6M (Figura 3) utiliza tecnología GPS para recibir señales de satélites y determinar la ubicación y hora con gran precisión. Al utilizar el sistema de posicionamiento global, este módulo proporciona datos de geolocalización y tiempo que son esenciales para proyectos que requieren sincronización temporal precisa.

Figura 3. Módulo Ublox NEO-6M [3]

En conclusión, la combinación de la Placa Arduino Uno, el Display de 7 Segmentos y 4 Dígitos basado en TM1637, y el Módulo GPS Ublox NEO-6M establece una base sólida para nuestro proyecto de reloj digital. La Placa Arduino Uno proporciona la capacidad de programación y control central, el Display de 7 Segmentos y 4 Dígitos ofrece una interfaz clara y numérica, mientras que el Módulo GPS Ublox NEO-6M asegura una referencia de tiempo precisa. Esta sinergia entre componentes esenciales nos permite abordar los desafíos asociados con la precisión temporal y la representación visual en el contexto de un reloj digital. A medida que avanzamos en el desarrollo del proyecto, exploraremos cómo estos elementos trabajan en conjunto para ofrecer un sistema integral y versátil.

[3] Simple Projects. (16 de junio de 2018). Interfacing Arduino with NEO-6M GPS module. Fuente: Simple Circuit. Recuperado de https://simple-circuit.com/arduino-neo-6m-gps-module/

CAPÍTULO III

Procedimiento

1. Investigación Inicial: Se comenzó explorando las especificaciones técnicas y los requisitos de los componentes necesarios para implementar nuestro reloj digital. Este dispositivo, basado en un módulo GPS y un display de 4 dígitos de 7 segmentos, utiliza una placa Arduino Uno como plataforma principal.
2. Programación Específica: Desarrollamos el código esencial para obtener y visualizar con precisión la hora. Esta fase incluyó la configuración de la comunicación con el módulo GPS, el procesamiento de los datos temporales y la actualización del display. Se priorizó la eficiencia y la exactitud en esta etapa.
3. Simulación en Proteus: Antes de pasar a la implementación física, realizamos simulaciones exhaustivas en Proteus. Este enfoque nos permitió validar el diseño y detectar posibles problemas antes de pasar a la fase de construcción. Verificamos el funcionamiento del circuito (Figura 4) en un entorno virtual.
4. Configuración Física: Con el diseño validado, procedimos a realizar las conexiones precisas entre el módulo GPS, Arduino Uno y el display. Esta fase involucró un montaje físico del circuito, donde cuidamos la disposición ordenada de los cables y aseguramos la alimentación adecuada de cada componente.

Este enfoque secuencial, desde la investigación inicial hasta la configuración física, garantizó una implementación metódica y eficiente de nuestro proyecto de reloj GPS.

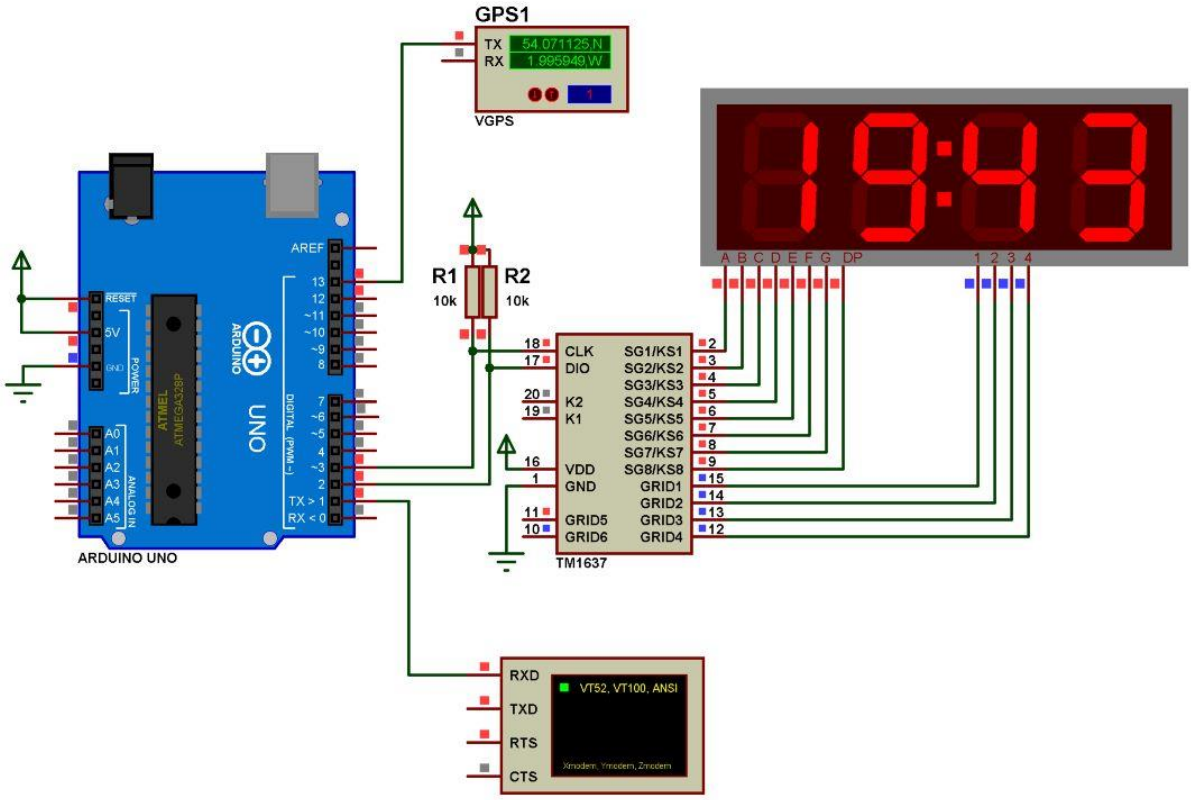

Figura 4. Conexión entre Arduino Uno, módulo GPS NEO-6M y Display TM1637 [4]

Presentación del código

El código desarrollado para este proyecto se encuentra a continuación. Este código permite la comunicación con el módulo GPS, la lectura de la hora y la visualización precisa en el display de 4 dígitos.

[4] Ponce, G. (2023). Circuito de Reloj GPS en Proteus. Fuente: Captura de pantalla propia.

1. Configuración inicial:

```
// Incluir las bibliotecas necesarias
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include <TM1637Display.h>

// Definir los pines para la comunicación serial por software y el módulo TM1637
#define S_RX 13
#define S_TX 12
#define CLK 3
#define DIO 2

// Inicializar instancias de las clases
TinyGPSPlus gps;  // Objeto para manejar datos del GPS
SoftwareSerial gpsSerial(S_RX, S_TX);   // Configurar la comunicación serial por software
TM1637Display display(CLK, DIO);  // Objeto para controlar el display de 7 segmentos

// Constante para ajuste de zona horaria local
const int TIME_ZONE_OFFSET = -3;
// Variable para controlar el estado de parpadeo
bool blinkState = false;

void setup() {
  // Inicializar la comunicación serial
  Serial.begin(9600);
  gpsSerial.begin(9600);
  // Configurar el brillo del display (0 a 7)
  display.setBrightness(7);
}
```

- Incluimos las bibliotecas necesarias para el GPS, la comunicación serial por software y el display TM1637.
- Definimos los pines para la comunicación serial por software, los pines de conexión del display y la zona horaria.
- Inicializamos instancias de las clases TinyGPSPlus, SoftwareSerial y TM1637Display.
- Configuramos la comunicación serial para la lectura del GPS y ajustamos el brillo del display.

2. Bucle principal (loop):

```
void loop() {
  // Verificar si hay datos disponibles en el módulo GPS
  if (gpsSerial.available() > 0) {
    // Leer y procesar los datos del GPS
    while (gpsSerial.available() > 0) {
      // Decodificar el mensaje del GPS
      if (gps.encode(gpsSerial.read())) {
        // Verificar si los datos del GPS son válidos
        if (gps.time.isValid()) {
          // Obtener la hora UTC del módulo GPS
          int utcHour = gps.time.hour();
          int utcMinute = gps.time.minute();

          // Calcular la hora local
          int localHour = (utcHour + TIME_ZONE_OFFSET + 24) % 24;
          // Concatenar horas y minutos en una variable entera
          int timeValue = localHour * 100 + gps.time.minute();

          // Imprimir la hora local en el puerto serie (se envían los segundos)
          Serial.print(localHour < 10 ? "0" : "");
          Serial.print(localHour);
          Serial.print(':');
          Serial.print(gps.time.minute() < 10 ? "0" : "");
          Serial.print(gps.time.minute());
          Serial.print(':');
          Serial.print(gps.time.second() < 10 ? "0" : "");
          Serial.println(gps.time.second());

          // Mostrar la hora completa en el display y controlar el parpadeo
          display.showNumberDecEx(timeValue, blinkState ? 0b11110000 : 0, true);
          blinkState = !blinkState;  // Cambiar el estado de parpadeo cada segundo
        } else {
          // Mostrar "Err" en el display si los datos del GPS no son válidos
          display.showNumberDecEx(0b01111001, 0b1000, true);   // E
          display.showNumberDecEx(0b01001111, 0b1000, false);  // r
          display.showNumberDecEx(0b01110011, 0b1000, false);  // r
        }
      }
    }
    gpsSerial.flush(); // Limpiar el buffer de recepción
  }
}
```

- Inicia el bucle principal, que se ejecuta continuamente.
- Verifica si hay datos disponibles en el módulo GPS.
- Procesa los datos del GPS utilizando la biblioteca TinyGPS++.
- Ajusta la hora local según la zona horaria.
- Muestra la hora en el puerto serie y en el display de 4 dígitos de 7 segmentos.
- Controla el parpadeo del display cada segundo para hacer más visible la actualización.
- En caso de datos inválidos del GPS, muestra "Err" en el display.

Este código integra la lectura del módulo GPS, ajusta la hora según la zona horaria y muestra la hora tanto en el puerto serie como en el display del reloj digital. Además, utiliza una técnica de parpadeo para hacer más visible la actualización de la hora en el display.

Simulación en Proteus y montaje físico

Se realizaron simulaciones en Proteus para depurar y verificar el funcionamiento del circuito en un entorno virtual antes de proceder con la implementación física, como se ilustra en la Figura 5.

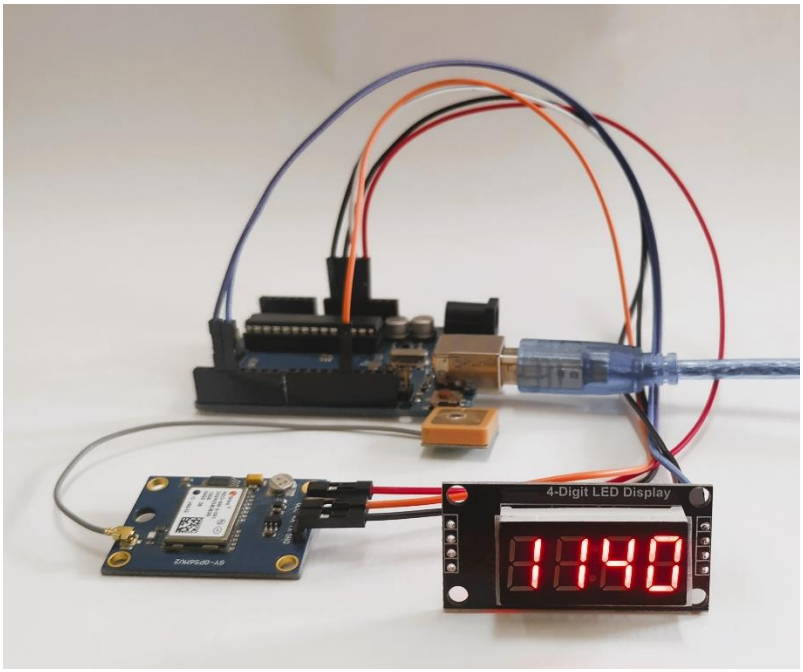

Figura 5. Montaje físico del reloj GPS en funcionamiento[5]

[5] Ponce, G. (2023). Montaje del Reloj GPS. Fotografía tomada por el autor.

Análisis de los resultados

El análisis de los resultados se centró en evaluar la eficacia y precisión del sistema implementado. Se revisaron aspectos como la estabilidad de las conexiones físicas, la sincronización entre el módulo GPS y el display, la actualización constante del reloj digital, así como los tiempos requeridos para que el módulo GPS sincronizara con los satélites, evaluando la eficiencia del proceso.

CAPÍTULO IV

Conclusiones y recomendaciones

Conclusiones

La implementación del reloj digital basado en GPS ha demostrado una mejora significativa en la precisión temporal en comparación con los relojes convencionales. La constante sincronización con satélites GPS elimina desviaciones y ofrece una representación exacta de la hora local, que, en la presente versión, debe ser configurada en el código.

Recomendaciones

Se recomienda el diseño de una placa de circuito impreso (PCB) personalizada que integre de manera eficiente los componentes del reloj GPS, optimizando su tamaño y organización. La inclusión de conexiones etiquetadas facilitará el montaje, mientras que la selección de materiales robustos garantizará resistencia a condiciones adversas. Es esencial documentar detalladamente el montaje y llevar a cabo pruebas de rendimiento en entornos específicos para asegurar su funcionamiento óptimo. Además, se sugiere mejorar la eficiencia energética, por ejemplo, mediante el control del brillo del display. Asimismo, se propone considerar la posibilidad de añadir ajustes de zona horaria directamente en el PCB o a través de procesamiento adicional en el código, aprovechando la información proporcionada por el módulo GPS.

Bibliografía

- Arduino. (2023). TM1637Display Library. Arduino Reference. Recuperado de https://bit.ly/3Ncozbi el 09 de noviembre de 2023.

- Components101. (16 de octubre de 2018). NEO-6M GPS Module. Components101. Recuperado de https://bit.ly/4182tMU el 09 de noviembre de 2023.

- Wikipedia. (2023, 11 de noviembre). Arduino Uno. Wikipedia, La enciclopedia libre. Recuperado de https://w.wiki/8Q2B el 11 de noviembre de 2023, a las 16:48.

- Wikipedia. (2023, 5 de enero). Reloj atómico. Wikipedia, La enciclopedia libre. Recuperado de https://w.wiki/8Q2A el 5 de enero de 2023, a las 06:37.

- GNSS.gov. (s. f.). Recepción de señales GPS para aplicaciones de tiempo. Recuperado de https://bit.ly/3R9b4KS

ANEXOS

Video explicando el proyecto: https://bit.ly/3RdAGGn

Repositorio de GitHub: https://bit.ly/3uRMxCJ