UNIVERSIDAD DE EXTREMADURA

ESCUELA POLITÉCNICA

Sistema de Gestión de Parking con Arduino

Francisco Javier González Rosco

10 de noviembre de 2024

ÍNDICE:

MOTIVACIÓN

El objetivo de este proyecto es crear un sistema automatizado de parking que controle el acceso y la salida de vehículos de manera eficiente, mostrando la disponibilidad en una pantalla LCD y gestionando una barrera automática, todo mediante Arduino y sensores. La motivación principal detrás de este proyecto fue el interés en entender y construir un sistema de control de acceso automatizado, como los utilizados en aparcamientos comerciales y públicos. Este proyecto permitió explorar el funcionamiento de sistemas de detección, control de acceso y contadores de ocupación en un entorno real, aplicando conocimientos básicos de electrónica y programación.

ESTADO DEL ARTE

Se consultaron varias fuentes para la implementación y diseño de este proyecto, tales como el manual oficial de Arduino y documentación de bibliotecas específicas, además de ejemplos de proyectos similares disponibles en plataformas de aprendizaje en línea como Instructables y Arduino Project Hub. Estos recursos ayudaron a comprender el uso de la librería **LiquidCrystal_I2C** para el manejo de pantallas LCD con interfaz I2C y el control del **Servo** para gestionar la barrera del parking.

Fuentes Consultadas

- Documentación oficial de Arduino (www.arduino.cc)
- Para la pantalla con módulo I2c (Conectar Pantalla LCD a Arduino por conexión I2C - InputMakers)
- Para el detector de obstáculos con sensor de infrarrojos (Detector de obstáculos con sensor infrarrojo y Arduino)
- Para el servo motor (Arduino - Control Servo with Visual Basic | Random Nerd Tutorials)
- Tutoriales de YouTube sobre detección y control de vehículos en Arduino

---

TÉCNICAS Y MATERIALES UTILIZADOS

Componentes de Hardware:

- Arduino Uno: El Arduino Uno es el controlador principal del sistema. Este microcontrolador lee los datos de los sensores y activa los LEDs y la barrera de forma sincronizada, permitiendo la automatización del estacionamiento.

- Sensores de infrarrojos (2 unidades); Los sensores de infrarrojos se colocan en la entrada y salida del estacionamiento para detectar la presencia de un vehículo. Cuando un vehículo cruza el sensor de entrada, el Arduino registra un ingreso y disminuye el conteo de espacios disponibles.Si el vehículo pasa por el sensor de salida, el conteo aumenta.

- Pantalla LCD con módulo I2C: La pantalla LCD, controlada por el módulo I2C, se usa para mostrar el número de espacios disponibles en el estacionamiento en tiempo real. El Arduino envía constantemente actualizaciones de la disponibilidad de espacios a la pantalla, proporcionando una referencia clara para los conductores antes de entrar.

- Servomotor (para la barrera): El servomotor controla la apertura y cierre de la barrera. Cuando el sensor de entrada detecta un vehículo, el Arduino envía una señal al servomotor para levantar la barrera, permitiendo el acceso. Tras el paso del vehículo, el Arduino envía otra señal para bajar la barrera.

- LEDs (rojo y verde): Los LEDs rojo y verde sirven como indicadores visuales para los conductores.

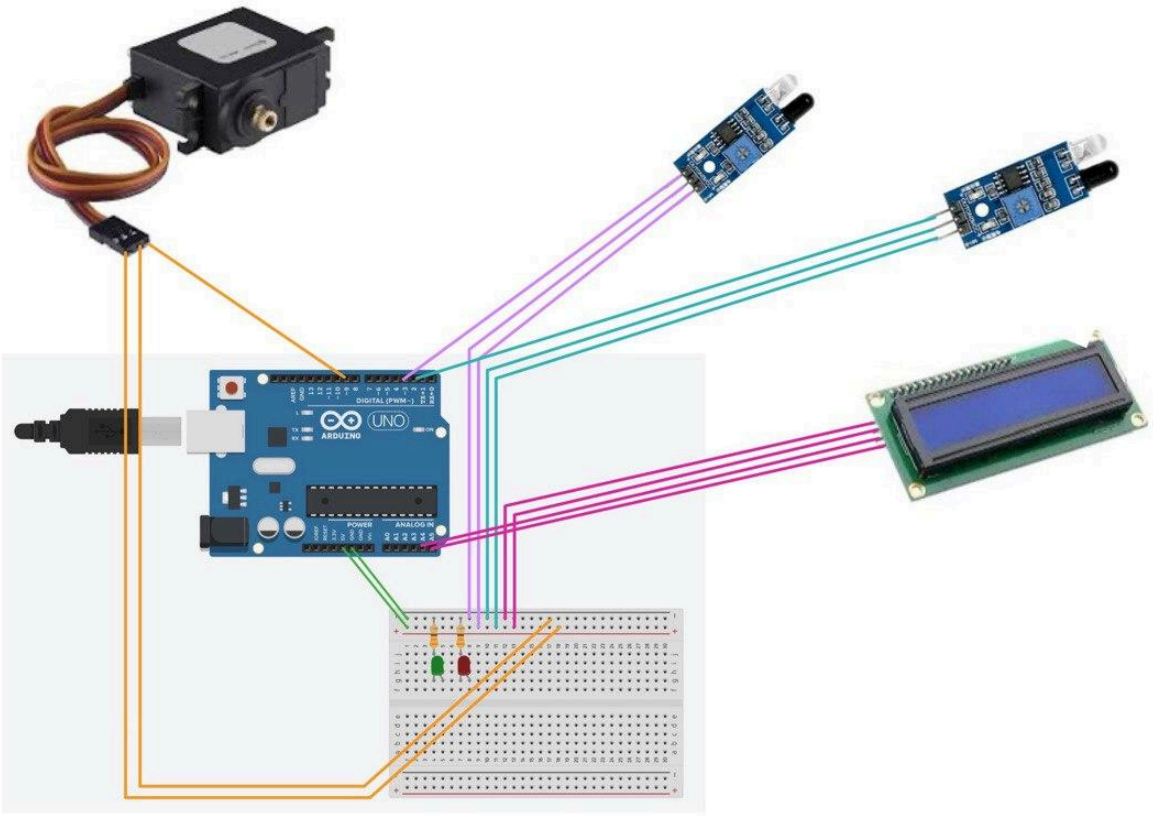
- Resistencias para LEDs (220Ω): Las resistencias de 220Ω se conectan en serie con los LEDs para limitar la corriente que pasa a través de ellos y evitar daños. Estas resistencias protegen los LEDs de sobrecargas, asegurando su funcionamiento seguro y prolongado en el sistema.

Software:

- IDE de Arduino (Eclipse para Arduino en Windows 11)
- Librerías: LiquidCrystal_I2C, Servo

ESQUEMAS, GRÁFICOS, FICHEROS O CÓDIGOS

Esquema de Conexiones



Código Principal (Enlace github)
Incluye fragmentos relevantes comentados:

1. **Configuración del sistema y librerías:** Se configuran los pines, se inicia la comunicación con la pantalla LCD y se establece el ángulo inicial del servomotor.
2. **Funciones Principales:**
    - inicializarParking(): Inicializa componentes del sistema y presenta un mensaje en pantalla.
    - verificarEntrada() y verificarSalida(): Gestionan la entrada y salida de vehículos mediante los sensores de infrarrojos.
    - actualizarPantalla(): Actualiza la pantalla LCD con el estado de ocupación actual.
    - subirBarrera() y bajarBarrera(): Controlan el servomotor para abrir y cerrar la barrera.

Explicación de Archivos

El proyecto está dividido en tres archivos principales, organizados para facilitar la estructura y el mantenimiento del código. Estos archivos son main.ino, parking.cpp y parking.h, y su propósito es dividir las funciones del sistema de forma clara.

1. main.ino

- **Propósito**: Este archivo contiene el código base de Arduino y gestiona el ciclo de ejecución principal del sistema de parking.
- **Estructura**:
  - **setup()**: Aquí se llama a la función inicializarParking() para configurar y preparar todos los componentes al inicio, como el servomotor, los sensores de infrarrojos, los LEDs y la pantalla LCD.
  - **loop()**: En el bucle principal, se ejecutan continuamente las funciones verificarEntrada() y verificarSalida() para detectar vehículos en los puntos de entrada y salida. Además, se incluye un breve retraso de 500 ms para asegurar que las lecturas de los sensores no sean demasiado rápidas.

2. parking.cpp

- **Propósito**: Este archivo implementa las funciones principales que gestionan el funcionamiento del sistema. Aquí se encuentran todas las operaciones de control del acceso de vehículos, actualización de la pantalla y manejo de la barrera.
- **Funciones Clave**:
  - **inicializarParking()**: Configura los pines de los componentes y muestra un mensaje inicial en la pantalla LCD para indicar que el sistema está listo.
  - **verificarEntrada()** y **verificarSalida()**: Estas funciones verifican si hay un vehículo en la entrada o salida respectivamente. Al detectar un vehículo, se actualiza el contador de vehículos, se sube la barrera, y se actualiza la pantalla.
  - **actualizarPantalla()**: Muestra la cantidad de espacios ocupados y libres en la pantalla LCD. También activa el LED rojo si el parking está lleno, y el LED verde si hay espacios disponibles.
  - **subirBarrera()** y **bajarBarrera()**: Controlan el movimiento del servomotor para abrir y cerrar la barrera.
  - **estadoBarrera()**: Retorna el estado actual de la barrera (abierta o cerrada).
  - **obtenerVehiculosOcupados()** y **obtenerEspaciosLibres()**: Devuelven la cantidad de vehículos en el parking y los espacios disponibles, respectivamente.

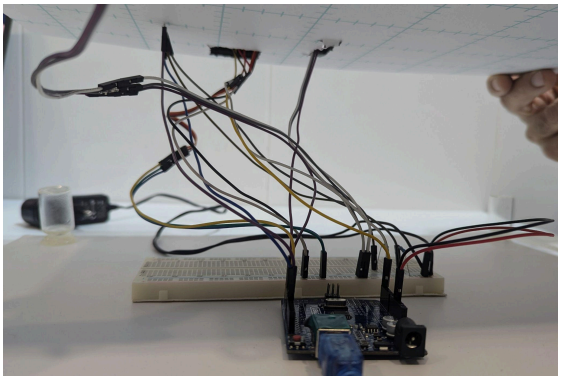
3. parking.h

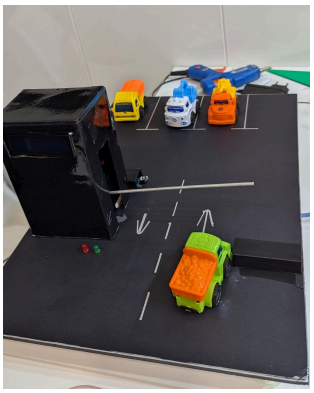
- **Propósito**: Este archivo de cabecera declara todas las funciones y variables globales utilizadas en parking.cpp. Su objetivo es proporcionar una interfaz clara y evitar problemas de duplicación de código.

- **Contenido**:
  - **Declaraciones de funciones**: Aquí se declaran todas las funciones que se utilizan en el archivo parking.cpp, como inicializarParking(), verificarEntrada(), verificarSalida(), y otras.
  - **Variables globales**: Se declaran las variables necesarias para el funcionamiento del sistema, como los pines de los sensores y LEDs, el contador de vehículos y el límite máximo de ocupación.

Esta estructura modular permite separar la lógica del programa en distintos archivos, facilitando así su mantenimiento, comprensión y futura expansión.

RESULTADO OBTENIDO (VÍDEO)





Este sistema automatizado de parking permite gestionar la ocupación mediante sensores de infrarrojos y una pantalla LCD que indica el estado actual. Durante las pruebas, el sistema demostró funcionar de acuerdo con las expectativas, detectando la entrada y salida de vehículos, actualizando el contador de ocupación y gestionando la barrera de manera automática. Los LEDs reflejan el estado del parking: verde para disponibilidad y rojo cuando está lleno. En este caso la pantalla estaba defectuosa por lo que no se ve con el brillo que debería.

Pruebas de Validación
Se realizaron pruebas para verificar la funcionalidad de cada módulo:

- **LCD**: Muestra el estado de ocupación en tiempo real.
- **LEDs**: Indican el estado del parking (libre o lleno).
- **Servomotor**: Abre y cierra la barrera tras detectar un vehículo.
- **Sensores de Infrarrojos**: Detectan la entrada y salida de vehículos con precisión.

FUTUROS DESARROLLOS POSIBLES
Existen varias mejoras potenciales para este sistema:

1. **Conexión a una Aplicación Móvil o Web**: Para visualizar en tiempo real la disponibilidad del parking desde cualquier lugar.
2. **Control de Acceso con Tarjeta RFID**: Permitir la entrada solo a vehículos autorizados, aumentando la seguridad.
3. **Integración de Cámara**: Capturar la matrícula del vehículo para registrar el acceso.

Este sistema podría mejorarse con un módulo Wi-Fi para enviar datos de ocupación a una aplicación móvil, permitiendo la consulta en tiempo real. Además, integrar una cámara podría habilitar funciones avanzadas como la captura de matrícula para mayor seguridad.

CONCLUSIÓN
Este proyecto permitió comprender y aplicar conceptos básicos de Arduino, como el uso de servomotores, pantallas LCD con protocolo I2C y sensores infrarrojos para la detección de vehículos. El sistema desarrollado ofrece una solución simple pero efectiva para la gestión de un parking automatizado, con posibilidades de adaptarse a aplicaciones a mayor escala.
