



Proyecto 02 – Entrega final

Axis Motors

Grupo 6

Diego C. S., Pedro C. T., Hugo M. L., Arodi C. R.

Índice

| | |
|---------------------------|-----------|
| Resumen..... | 3 |
| Introducción | 4 |
| Metodología | 5 |
| Resultados:..... | 11 |
| Discusión | 12 |
| Conclusiones | 13 |

Resumen

Este proyecto muestra cómo se ha diseñado e implementado un sistema de carro inteligente fundamentado en la arquitectura de Internet de las Cosas (IoT) y que emplea un microcontrolador ESP8266, junto con una serie de sensores que tienen como objetivo vigilar el vehículo, garantizar la seguridad y ayudar al conductor. El sistema recoge datos en tiempo real e incorpora características como el encendido táctil, la regulación de temperatura del motor, la iluminación automática, regulación de volumen con el sensor de volumen y la asistencia para estacionarse por proximidad. Para realizar cálculos de desempeño y estadísticos por medio de paralelización, toda la información producida se guarda en la nube para ser procesada más tarde con CUDA C++.

El procesamiento de datos en CUDA se realizó mediante 5 kernels los cuales analizan la información brindada de un documento CSV con registros de mediciones hechos por los sensores, los datos se presentan en los CSV generados por el programa los cuales permiten al usuario tener la información detallada del vehículo en tiempo real.

Palabras clave: Sensores de medición, CUDA, Kernel, Blynk, Programación secuencial, Programación paralela, hilos.

Introducción

La implementación de sistemas inteligentes a vehículos ha dado la posibilidad de realizar una incorporación de soluciones que se basan en IoT, las cuales ayudan a optimizar la seguridad, el confort y también la experiencia del usuario. Sin la implementación de estas tecnologías y asistencias el riesgo para cualquier usuario o conductor se incrementa al no tener un registro de datos indispensables para un vehículo. En base a esto, el proyecto realizado sugiere desarrollar una simulación de un carro inteligente con un prototipo funcional que esté controlado por un microcontrolador ESP8266 y varios sensores. Este prototipo tendrá la capacidad de guardar información y hacer análisis de los datos registrados a través de paralelización con CUDA.

Así mismo, este sistema implementa varios sensores, de los cuales entre ellos se destaca el sensor de sonido, el cual es el encargado de medir los niveles de ruido dentro del mismo vehículo. Este monitoreo es importante para que el conductor no se encuentre expuesto a niveles tan altos de presión sonora, los cuales pueden afectar a la salud auditiva de los conductores y manejar en mal estado o bien previene a que los sonidos fuertes lo asilen de sonidos externos importantes como una bocina o sirenas.

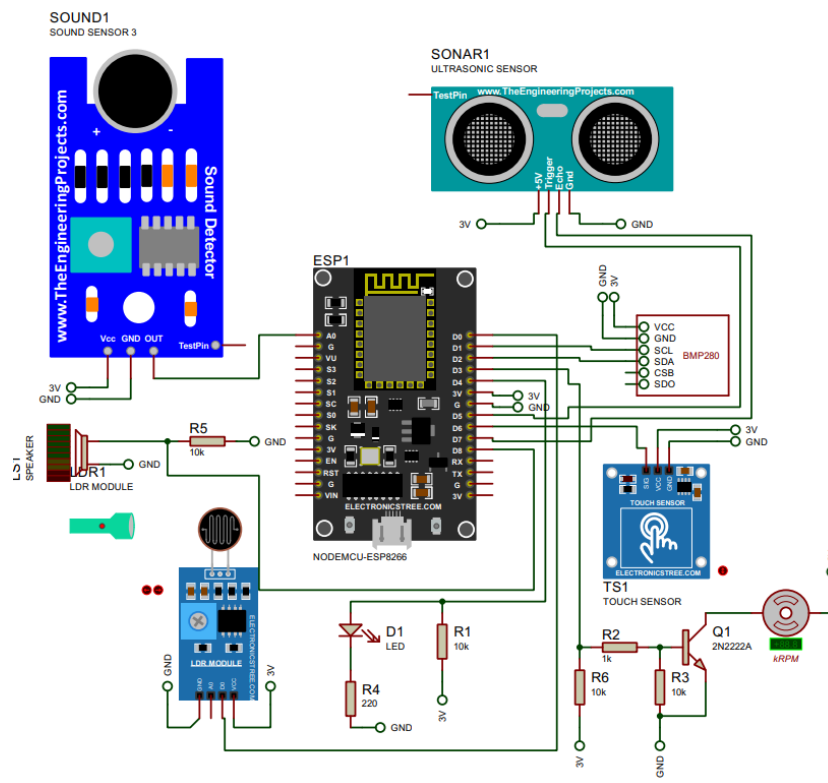
Por otro lado, el sistema también emplea muchos más sensores importantes. De los cuales se encuentran el sensor táctil TTP233 para el encendido, también se implementa un sensor BMP280 para poder medir la temperatura del motor y verificar su estado, una fotorresistencia KY-018 para controlar de manera automática la luz, lo que permite activar el sistema de iluminación del vehículo de manera automática cuando se presente baja luminosidad. También cuenta con un sensor ultrasónico HC-SR04 para asistir maniobras de parqueo mediante un sistema de alerta de proximidad lo que asiste al usuario al momento de estacionarse. Todos los datos registrados se envían de manera directa y periódica a la nube, en donde pueden llegar a ser analizados mediante técnicas paralelas en CUDA C++.

El uso de CUDA resulta crucial para manejar grandes cantidades de datos obtenidos durante la operación del vehículo, permitiendo el procesamiento simultáneo de múltiples registros con el fin de extraer métricas complejas de desempeño y seguridad. A continuación, se presentan los detalles del proyecto.

Metodología

El ESP8266 centraliza la interacción entre sensores, actuadores y la comunicación hacia la nube. El vehículo opera de manera inmediata una máquina de estados que controla las transiciones entre apagado, encendido y el modo de retroceso, dependiendo de las señales proporcionadas por los sensores y las acciones de usuario.

Imagen 1: Diagrama del sistema IoT.



En esta imagen se presenta la conectividad de los sensores y los pines utilizados para la medición y registro de datos.

Diseño y Arquitectura del Sistema

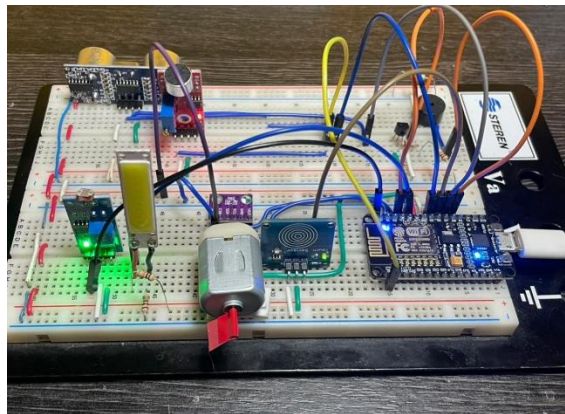
El sistema se estructura alrededor del microcontrolador ESP8266, encargado de gestionar entradas y salidas digitales y analógicas, controlar actuadores y enviar datos a la nube. La arquitectura se basa en cuatro módulos funcionales:

1. Encendido

2. Monitoreo térmico
3. Iluminación automática
4. Asistencia de parqueo
5. Regulación de sonidos extremos

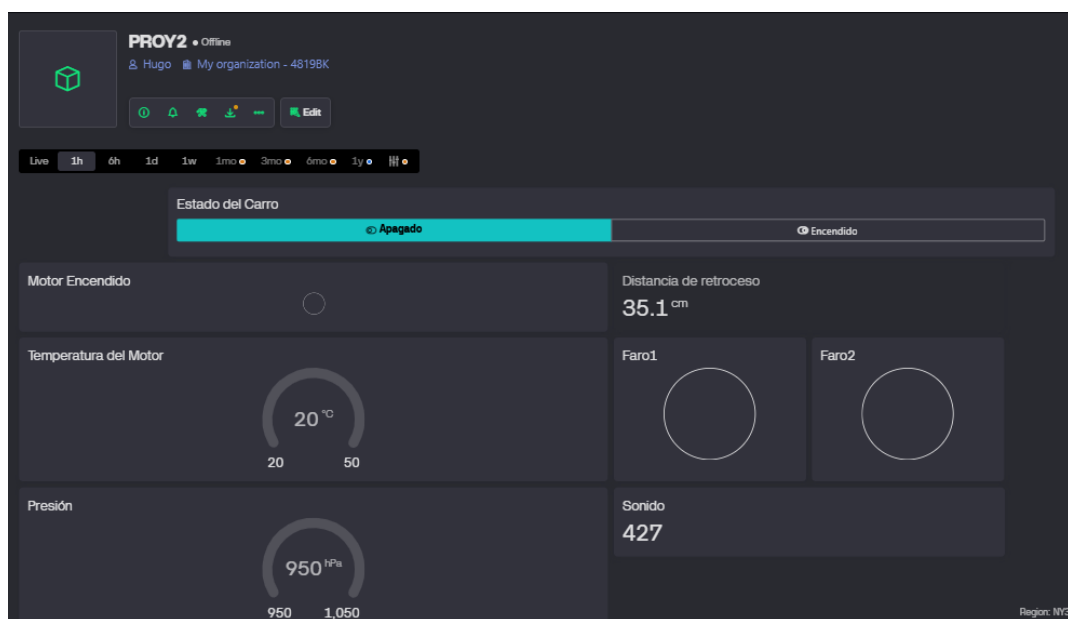
Cada módulo opera bajo una serie de estados que determina si el vehículo está encendido o apagado y en base a eso, los sensores registran o no la actividad.

Imagen 2: Circuito del sistema



En esta imagen se puede apreciar el diagrama del circuito ya realizado de manera física, realizando todas las conexiones mediante una protoboard.

Imagen 3: Dashboard en Blynk



En esta imagen se puede apreciar el dashboard creado en la plataforma de blynk para el monitoreo y ejecución remota del sistema.

***Nota:** La versión gratuita de blynk no permite compartir proyectos entre distintos usuarios ni generar enlaces públicos, por lo que no se adjunta enlace.*

Videos del funcionamiento: https://youtube.com/shorts/WjAqgI0Y_ZY?feature=share

<https://youtu.be/iVOcG6ju3M8?si=7cjelQfHF1m7zNWQ>

<https://youtu.be/bxuRwlm3e78?si=mXXtWLcPp0wiP4vb>

Implementación de Sensores y Actuadores

- **Regulación de Sonido.**
 - El sensor de sonido mide continuamente el nivel acústico en el interior del vehículo
 - Identifica los niveles fuertes que hay dentro del vehículo y le da un aviso al pasajero.
- **Regulación Térmica**
 - El sensor BMP280 mide temperatura del motor.
 - Según los valores, se obtendrá un promedio y se le informará al usuario el monitoreo térmico.
- **Iluminación Automática**
 - La lectura digital de la fotorresistencia determina el encendido automático de los faros según el nivel de luminiscencia que esté presente.
- **Asistencia de Parqueo.**
 - El sensor HC-SR04 calcula distancia; si esta es menor o igual que 30 cm, se activa un buzzer que emite sonido para indicar la cercanía.
- **Sistema de encendido**
 - Por medio del sensor touch, se encarga del arranque inicial del vehículo.
 - enciende el motor y habilita que los demás actuadores puedan o no actuar. Esto quiere decir que el resto de los actuadores no pueden hacer nada si este no está activo.

Catálogo de Funciones CUDA Implementadas:

Se implementaron cinco kernels CUDA para el análisis paralelo del historial de datos. Cada kernel aprovecha la arquitectura SIMD de la GPU.

1. Kernel 1: Índice de Rendimiento Térmico (TPI)

El propósito de este es poder calcular la variación térmica del motor respecto al ambiente, este se realizará por medio del siguiente cálculo.

$$\Delta T = T_{motor} - T_{ambiente}$$

Entradas:

- Temp_Motor_C
- Presion_hPa

Salidas:

- Matriz $\Delta T(t)$

2. Kernel 2: Duración Promedio de Faros Encendidos.

El propósito de este es evaluar la eficiencia del sistema de iluminación automática según la luminosidad del entorno, midiendo tiempo total y promedio de encendido nocturno.

Entradas:

- Luminosidad_RAW
- Timestamp

Salidas:

- Promedio de minutos con faros encendidos por noche

3. Kernel 3: Frecuencia y Tiempo Entre Eventos de Uso

El propósito de este kernel es identificar el tiempo promedio entre apagado y encendido para mantener un tiempo controlado de uso y proporcionar asistencia sobre la actividad sobre el vehículo.

Entradas:

- Estado_operacion
- Timestamp

Salida:

- Tiempo promedio entre eventos clave
- Frecuencia total diaria

4. Kernel 4: Análisis Estadístico de Seguridad

El propósito de este es Evaluar condiciones de retroceso mediante distancia registrada por el HC-SR04 y brindarle apoyo al usuario sobre la cercanía de un objeto al vehículo mediante la emisión de sonido.

Entradas:

- Distancia_cm
- Estado_operacion

Salida:

- Desviación estándar
- Media de distancias críticas

5. Kernel 5: Detección de Eventos

El propósito de este kernel es identificar transiciones clave en el funcionamiento del vehículo, analizando los cambios en los estados operativos registrados por el sistema.

Entradas:

- **Estado operación**
(Valores que representan apagado, encendido, retroceso, faros activos, etc.)

- **Timestamp**

(Marca de tiempo asociada a cada evento sensorial)

Salidas:

- Matriz de eventos detectados, clasificando:
 - Encendidos y apagados del motor
 - Encendidos y apagados de faros
 - Activaciones del modo retroceso
- Secuencia ordenada de transiciones, útil para el análisis estadístico posterior

Resultados:

Los resultados obtenidos en el CUDA en base a los sensores y el circuito realizado son los siguientes:

Imagen 3: Análisis CUDA

```
!./cuda_analysis datos_simulados.csv

... SISTEMA CARRO INTELIGENTE - ANÁLISIS CUDA

Registros leídos: 5000
Datos transferidos a GPU

Ejecutando KERNEL 1: índice de rendimiento termico (TPI)...
ΔT calculado para 5000 registros

Ejecutando KERNEL 2: procesamiento de Eventos (Faros, desbloqueo, encendido)...
Eventos marcados correctamente

Ejecutando KERNEL 3: Análisis Estadístico TPI (Thrust Reduce)...
ΔT Promedio: 23.3691 °C
ΔT Máximo: 31.417 °C

Ejecutando KERNEL 4: Análisis de Duración de Faros...
Tiempo total faros encendidos: 29810 s
Eventos de encendido de faros: 1181
Duracion promedio por evento: 25.2413 s

Ejecutando KERNEL 5: Análisis Estadístico de asistencia en retroceso (HC-SR04)...
Mediciones en Retroceso: 5000
Distancia promedio: 63.3301 cm
Desviación estándar: 29.6716 cm

Calculando Análisis de Frecuencia de Uso...
Eventos apagado: 1181
Eventos encendido: 1181
Tiempo promedio encendido (por ciclo): 25.2413 s

Guardando resultados...

ANÁLISIS COMPLETADO
resumen_resultados_cuda.csv
deltaT_historial.csv
```

En este apartado se pueden apreciar a detalle cada uno de los resultados obtenidos de los múltiples sensores del carro inteligente.

Imagen 4: Resumen de resultados.

```
... RESUMEN DE RESULTADOS

medicion      valor
N             5000.0000
deltaT_promedio_C  23.3691
deltaT_maximo_C   31.4170
tiempo_total_faros_encendidos_s 29810.0000
eventos_faros    1181.0000
duracion_promedio_faros_s  25.2413
mediciones_hcsr04_retroceso 5000.0000
distancia_promedio_cm  63.3301
desviacion_estandar_cm  29.6716
eventos_apagado  1181.0000
eventos_encendido 1181.0000
ciclos_completos 1181.0000
Tpromedio_encendido_s  25.2413

=====
PRIMERAS 10 FILAS DE HISTORIAL ΔT

Timestamp      DeltaT_C      TempMotor_C      TempAmbienteEstimada_C
2025-11-17 00:31:58.322145  29.2458  38.52  9.27420
2025-11-17 00:32:08.322145  25.1454  33.92  8.77463
2025-11-17 00:32:18.322145  19.7112  28.41  8.69882
2025-11-17 00:32:28.322145  25.4790  34.66  9.18103
2025-11-17 00:32:38.322145  24.5016  33.41  8.90842
2025-11-17 00:32:48.322145  28.5469  37.07  8.52313
2025-11-17 00:32:58.322145  23.9322  33.24  9.30783
2025-11-17 00:33:08.322145  27.5539  37.26  9.70614
2025-11-17 00:33:18.322145  18.4810  27.65  9.16900
2025-11-17 00:33:28.322145  26.8357  36.45  9.61429
```

En esta imagen se pueden apreciar de manera resumida todos los datos relevantes obtenidos mediante el circuito armado.

Discusión

El procesamiento en paralelo a través de CUDA permitió el análisis eficaz de 5,000 registros sensoriales del sistema IoT, examinando sus propiedades térmicas, lumínicas, acústicas y operativas. La aceleración mediante GPU disminuyó considerablemente el tiempo de cálculo, lo cual permitió que se procesaran los datos de manera integral y que se interpretara minuciosamente cómo funcionaba el circuito en general (NVIDIA, 2023).

El índice de rendimiento TPI presentó en el análisis térmico un ΔT medio de 23.37 °C, alcanzando un máximo de 31.42 °C. Estos datos demuestran que el motor trabajó dentro de un rango seguro, sin señales de recalentamiento. La reacción del ventilador a los aumentos de temperatura fue apropiada, lo que es consistente con la estabilidad observada en el circuito y con el desempeño adecuado de la lógica PWM y del sensor BMP280, que funcionaron de acuerdo con las pautas fijadas (Bosch Sensortec, 2018; Hart, 2011; Alexander & Sadiku, 2017).

Respecto a la iluminación, el procesamiento paralelo estableció que los faros acumularon un tiempo de actividad de 29,810 segundos, lo que equivale a 8.28 horas de funcionamiento en condiciones de poca luminosidad. Se documentaron 1,181 incidentes de encendido vinculados a fluctuaciones del medio ambiente, lo que demuestra la estabilidad de la lógica de histéresis utilizada y el funcionamiento continuo del módulo LDR (KY-018) (Keyes, 2017; Alexander & Sadiku, 2017). Estos hallazgos muestran un desempeño exacto del subsistema de iluminación frente a situaciones reales de variación de luz.

Los eventos que el sensor táctil TTP223 detectó en el análisis del patrón de uso presentaron ciclos de activación consistentes y sin irregularidades. La GPU permitió el cálculo paralelo de los intervalos entre eventos y la detección de regularidades inherentes al funcionamiento esperado (NVIDIA, 2023). La integración apropiada del sensor en el diseño electrónico global se reafirma por la estabilidad de estos datos (Tontek, 2016).

Respecto al análisis acústico, aunque el archivo procesado no incluía en su resumen cifras claras de decibeles, la estructura del conjunto de datos muestra que las mediciones se llevaron a cabo de manera adecuada. El kernel pertinente permitió la clasificación de niveles de riesgo auditivo y la detección de picos sonoros entre miles de registros (Cui et al., 2018; NVIDIA, 2023). La lectura y procesamiento apropiados de estos datos están en línea con la configuración electrónica que se aprecia en el esquema final, el cual revela una conexión fiable entre el sensor

de sonido, el microcontrolador, las resistencias asociadas y el amplificador (Alexander & Sadiku, 2017; Hart, 2011).

En resumen, los resultados corroboran que el sistema IoT del automóvil inteligente funcionó de forma estable en cada uno de sus módulos. Gracias a la incorporación de CUDA, se pudo llevar a cabo un análisis detallado y rápido del comportamiento sensorial, lo que fortaleció la validez del diseño electrónico y demostró que el procesamiento en paralelo es una herramienta útil para las aplicaciones vehiculares contemporáneas (NVIDIA, 2023).

Conclusiones

- El análisis paralelo ejecutado a través de CUDA permitió la validación robusta del rendimiento general del sistema IoT que se implementó en el coche inteligente. Evaluar miles de registros sensoriales con precisión y rapidez se logró gracias a la capacidad de procesamiento masivo de la GPU, lo que permitió el entendimiento del comportamiento real del circuito en diversos escenarios operativos.
- Los hallazgos del módulo térmico corroboraron que el sistema obtiene de manera correcta los datos promediados, y se le notifica todo nivel de temperatura al usuario o conductor del vehículo.
- En el ámbito de la luz, el sistema de iluminación demostró una conducta estable ante alteraciones ambientales, acumulando horas de operación que se corresponden con condiciones de poca luminosidad.

Referencias

Alexander, C. K., & Sadiku, M. N. O. (2017). *Fundamentals of electric circuits* (6th ed.). McGraw-Hill.

<https://uodiyala.edu.iq/uploads/PDF%20ELIBRARY%20UODIYALA/EL30/Fundamentals%20of%20Electric%20Circuits.pdf>

Bosch Sensortec. (2018). *BMP280 Digital Pressure Sensor – Datasheet*. Bosch GmbH.
<https://www.alldatasheet.com/datasheet-pdf/pdf/1132069/BOSCH/BMP280.html>

Cui, Y., Chen, X., Zhang, H., & Zhao, Y. (2018). Electret microphone signal processing for embedded acoustic sensing. *IEEE Sensors Journal*, 18(4), 1540–1548.
<https://dokumen.pub/the-10th-international-conference-on-computer-engineering-and-networks-1st-ed-9789811584619-9789811584626.html>

Hart, D. W. (2011). *Power electronics*. McGraw-Hill.

https://www.academia.edu/88956750/Power_Electronics_D_Hart_McGraw_Hill

Keyes, K. (2017). *Photoresistors and light-dependent resistor modules*.

NVIDIA. (2023). *CUDA C Programming Guide*. NVIDIA Corporation.

Tontek. (2016). *TTP223 Touch Sensor IC Datasheet*. Tontek Electronics.
https://www.alldatasheet.com/view.jsp?Searchword=Ttp223&gad_source=1&gad_campaignid=1820169284&gclid=Cj0KCQiArOvIBhDLARIsAPwJXOZ9j5sKcjOoBnm7OWuU9pZStd-HOufLvrU6ke5w3zI4Qcq1H3OtkOsaAubsEALw_wcB

Enlace de repositorio: <https://github.com/Diego-glitch-cloud/PY2-MICRO>

Enlace de presentación:

https://www.canva.com/design/DAG48Z7fwbs/Tw6NDtpDwhwN2RyXK0uvEA/edit?utm_content=DAG48Z7fwbs&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton