

**Modelación de sistemas mínimos y arquitecturas
computacionales (Gpo 200)**

**Reporte Final: Desarrollo de una Solución con
Sensores y Microcontroladores**

Prof. Ricardo Valera Velázquez

Integrantes

Ángel Alexander Báez Flores - A01425613

Osmar Enrique Sánchez Martínez - A01425521

Alejandra Galván Bojórquez - A01424388

Marco Armando Islas Brito - A01425495

Fecha de entrega:

20 de octubre de 2024



Índice

Resumen ejecutivo	2
Introducción	2
Identificación y especificación del problema	3
1.1 Identificación del problema	3
1.2 Especificación de la variable a medir	4
1.3 Selección y descripción del sensor	7
1.4 Diagrama de la propuesta de solución	9
1.5 Desarrollo del programa	14
Resultados	18
Logros obtenidos	18
Desafíos Enfrentados	19
Lecciones Aprendidas	19
Conclusión	20
Referencias	21
Anexos	21




Resumen ejecutivo

El uso de sensores y sistemas electrónicos en la industria automotriz se ha vuelto cada vez más frecuente, debido a su capacidad para optimizar el funcionamiento de los vehículos. No obstante, al igual que otras industrias, la industria automotriz no está exenta de desafíos tecnológicos que buscan poner a prueba el desempeño y la eficiencia de los vehículos que tenemos hoy en día. Por lo tanto, el presente trabajo busca plantear una solución a 4 sistemas que consideramos esenciales para la industria automotriz: frenado ABS (Anti-Lock Braking System), inyección de combustible, sistema de enfriamiento y control de estabilidad (ESC), con el objetivo de mitigar estos problemas que la industria automotriz enfrenta actualmente. Además, el principal objetivo alcanzado en este trabajo es la simulación de dicha solución en 16 bit programada en ensamblador, la cual ha sido analizada bajo 4 situaciones distintas para verificar que los mecanismos de activación de los sensores sea el adecuado dado los parámetros iniciales de los mismos.

Entre los principales logros obtenidos se encuentra la comprensión del funcionamiento de sistemas embebidos a bajo nivel y la capacidad de programar en ensamblador para gestionar datos sin comprometer el tiempo de respuesta. Asimismo, este trabajo demuestra la importancia de la eficiencia, optimización del código y la evaluación de escenarios distintos para adaptar el sistema a condiciones variables del vehículo y lograr una toma de decisiones precisa.

Introducción

La industria automotriz enfrenta diversos desafíos tecnológicos que demandan soluciones innovadoras y eficaces. En un mundo donde la seguridad, la eficiencia y la sostenibilidad son prioridades, la programación avanzada de vehículos se ha convertido en un pilar fundamental para abordar estos retos. La modernización de sistemas críticos, como el frenado, la inyección de combustible y el sistema de enfriamiento, no solo es esencial para el rendimiento del vehículo, sino que también juega un papel clave en la reducción de su impacto ambiental.



El propósito de esta situación problema es identificar un problema actual que afecte a los vehículos y que pueda ser mitigado o solucionado por medio de la incorporación de uno o varios sensores. Nos enfocaremos en 4 sistemas que consideramos esenciales para la industria automotriz los cuales son: el frenado ABS (Anti-Lock Braking System), inyección de combustible, sistema de enfriamiento y control de estabilidad (ESC).

Entre los principales objetivos del proyecto está seleccionar un sensor adecuado para cada uno de los sistemas planteados anteriormente y diseñar un diagrama de flujo donde se visualice la propuesta de solución al sistema. Finalmente, crear un programa funcional usando ensamblador que lea, almacene y procese la información del sensor, simulando la solución completa al problema planteado.


Identificación y especificación del problema

1.1 Identificación del problema

En las últimas décadas, los países que han tenido un repunte económico importante, tienen (entre otras características) el desarrollo de tecnología propia como eje principal de su economía. Tal es el caso de Corea del Sur, China, India, entre otros.

A partir de la década de 1970, el desarrollo tecnológico ha estado basado en las computadoras y sus diversos componentes, uno de estos componentes, incluso considerado el cerebro de la misma, es la Unidad Central de Procesamiento (CPU por sus siglas en inglés).

El CPU no solo es utilizado en las computadoras, sino en todo dispositivo digital que existe hoy en día, por ejemplo, en los teléfonos celulares, tabletas electrónicas y automóviles. Es interesante el caso de los automóviles, donde, para que te des una idea, se pueden encontrar hasta 70 u 80 microcontroladores, donde cada uno de ellos es considerado por sí mismo, una pequeña computadora con su propio CPU.



En la actualidad, los automóviles dependen de los sensores para realizar un sinnúmero de funciones. Entre los sensores más comunes se encuentran los acelerómetros de las bolsas de aire, los sensores de presión absoluta, los sensores de velocidad de rotación y los monitores de presión de los neumáticos.

Como sabemos que la tecnología va cambiando constantemente y que los avances tecnológicos ocurren día con día. El propósito de esta situación problema es que identifiques un problema actual que afecte a los vehículos y que este pueda ser mitigado o solucionado por medio de la incorporación de uno o varios sensores.

1.2 Especificación de la variable a medir

Frenado ABS (Anti-Lock Braking System)

- **Desafíos:** El ABS evita que las ruedas se bloqueen durante el frenado, manteniendo la tracción y el control del vehículo. La precisión en la programación del sistema es crucial para adaptarse a diferentes condiciones de carretera (lluvia, nieve, grava, etc.).
- **Soluciones Programáticas:** Los algoritmos de control de ABS deben procesar datos de los sensores de velocidad de las ruedas, ajustando automáticamente la presión de frenado. El reto es crear sistemas de control robustos y de baja latencia que funcionen en diversas condiciones.
- **Velocidad de las ruedas:** Se mide en revoluciones por minuto (RPM) o en metros por segundo (m/s). Esta variable es crucial para determinar si una rueda está bloqueando o patinando durante el frenado.
- **Sensor de velocidad de rueda:** Utilizando un sensor de efecto Hall, se puede medir la velocidad de cada rueda. Estos datos se procesan mediante un convertidor analógico-digital para obtener una señal digital que pueda ser utilizada por el ECU del vehículo.



Inyección de Combustible

- **Desafíos:** La eficiencia del motor depende de la cantidad exacta de combustible que se inyecta en los cilindros. La programación del sistema de inyección de combustible debe optimizar la mezcla aire-combustible en función de las condiciones del motor, como la velocidad, la carga y la temperatura.
- **Soluciones Programáticas:** La optimización de algoritmos de control para mejorar la eficiencia del motor y reducir las emisiones es un área clave. Estos algoritmos deben manejar con precisión la sincronización de la inyección y ajustar la mezcla de aire y combustible en función de los datos obtenidos por los sensores.
- **Presión de inyección de combustible:** Se mide en bar (bar) o psi (libras por pulgada cuadrada). Esta variable afecta directamente la cantidad de combustible que se inyecta en los cilindros.
- **Sensor de presión de combustible:** Utilizando un sensor piezorresistivo, se puede medir la presión del combustible en la línea de inyección. Los datos se envían al ECU para ajustar la cantidad de combustible inyectado según las condiciones del motor.

Sistema de Enfriamiento

- **Desafíos:** El sobrecalentamiento del motor puede causar daños severos. La programación del sistema de enfriamiento debe regular la temperatura del motor de manera efectiva, activando ventiladores y ajustando el flujo de refrigerante.
- **Soluciones Programáticas:** Los algoritmos de control del sistema de enfriamiento deben ser capaces de anticipar cambios en la temperatura basados en la carga del motor y las condiciones externas, optimizando el uso de energía para mantener el motor dentro de un rango seguro de temperatura.

- **Temperatura del refrigerante:** Se mide en grados Celsius ($^{\circ}\text{C}$). Esta variable es fundamental para asegurar que el motor opere en un rango seguro de temperatura.
- **Sensor de temperatura del refrigerante:** Un sensor de temperatura tipo termistor puede ser utilizado para medir la temperatura del refrigerante. Los datos se envían al ECU, que toma decisiones sobre la activación de ventiladores o el ajuste del flujo de refrigerante.

Control de Estabilidad (ESC)

- **Desafíos:** El sistema de control de estabilidad ayuda a mantener el control del vehículo en situaciones de manejo difíciles. La programación debe detectar el deslizamiento y aplicar automáticamente los frenos a las ruedas específicas para corregir la trayectoria del vehículo.
- **Soluciones Programáticas:** Se requieren algoritmos que integren datos de múltiples sensores (como giroscopios y acelerómetros) para predecir y prevenir la pérdida de control. Estos sistemas deben tener alta capacidad de procesamiento para reaccionar en milisegundos.
- **Ángulo de deslizamiento:** Se mide en grados ($^{\circ}$). Esta variable indica la desviación entre la dirección en la que el vehículo se mueve y la dirección en la que está apuntando.
- **Sensor de ángulo de dirección:** Utilizando un giroscopio, se puede medir el ángulo de deslizamiento. Estos datos se procesan para determinar cuándo se debe activar el ESC y aplicar los frenos a las ruedas necesarias para corregir la trayectoria del vehículo.

1.3 Selección y descripción del sensor

Sensor de Velocidad de Rueda (para Frenado ABS)

- **Sensor de Efecto Hall:** Este sensor es elegido debido a su alta precisión y capacidad para medir la velocidad de las ruedas. Es robusto y se adapta bien a las condiciones adversas del entorno automotriz (como humedad y vibraciones). Además, su bajo consumo de energía lo convierte en una opción ideal para aplicaciones en vehículos.
- **Función:** El sensor de velocidad de rueda envía datos al ECU sobre la velocidad de cada rueda. Si detecta que una o más ruedas están a punto de bloquearse durante el frenado, se activa un aviso luminoso (LED) o un mensaje corto (ej. "Frenado ABS Activo").
- **Mecanismo de activación:** Cuando la velocidad de la rueda se reduce abruptamente, el ECU interpreta esto como un posible bloqueo. Entonces, el ECU envía una señal a un LED para encenderlo, indicando al conductor que el sistema ABS está en funcionamiento.

Sensor de Presión de Combustible (para Inyección de Combustible)

- **Sensor Piezorresistivo:** Este sensor es ideal para medir la presión de combustible debido a su alta sensibilidad y precisión. Es capaz de operar en un amplio rango de presiones y temperaturas, lo que lo hace adecuado para el entorno del motor.
- **Función:** El sensor monitorea constantemente la presión de inyección de combustible y envía los datos al ECU. Si la presión está por debajo de un umbral crítico, se activa un aviso luminoso (LED) o un mensaje corto (ej. "Baja Presión Combustible").
- **Mecanismo de activación:** Si la presión de combustible cae por debajo del nivel seguro, el ECU activa el LED y envía un mensaje al tablero de instrumentos, alertando al conductor sobre el problema.



Sensor de Temperatura del Refrigerante (para Sistema de Enfriamiento)

- **Termistor:** Este sensor es seleccionado por su capacidad de medir temperaturas con alta precisión en un rango adecuado para los sistemas de refrigeración de vehículos. Su respuesta rápida a los cambios de temperatura lo hace perfecto para esta aplicación.
- **Función:** El sensor monitorea la temperatura del refrigerante y envía esta información al ECU. Si la temperatura excede un límite crítico, se activa un aviso luminoso (LED) o un mensaje corto (ej. "Motor Sobrecalentado").
- **Mecanismo de activación:** Si la temperatura del refrigerante supera un valor predefinido, el ECU enciende un LED y muestra un mensaje en el tablero, advirtiendo al conductor que el motor puede estar sobrecalentándose.

Sensor de Ángulo de Dirección (para Control de Estabilidad)

- **Giroscopio:** Este sensor es elegido debido a su capacidad para medir ángulos con alta precisión y su rapidez de respuesta. Su uso en aplicaciones de control de estabilidad es fundamental para una respuesta instantánea ante situaciones de deslizamiento.
- **Función:** El sensor detecta el ángulo de deslizamiento y envía datos al ECU. Si se detecta un deslizamiento significativo que podría comprometer la estabilidad del vehículo, se activa un aviso luminoso (LED) o un mensaje corto (ej. "Estabilidad Comprometida").
- **Mecanismo de activación:** Cuando el giroscopio indica que el vehículo está perdiendo tracción, el ECU enciende un LED y muestra un mensaje en el tablero, alertando al conductor sobre la necesidad de una intervención inmediata.

1.4 Diagrama de la propuesta de solución

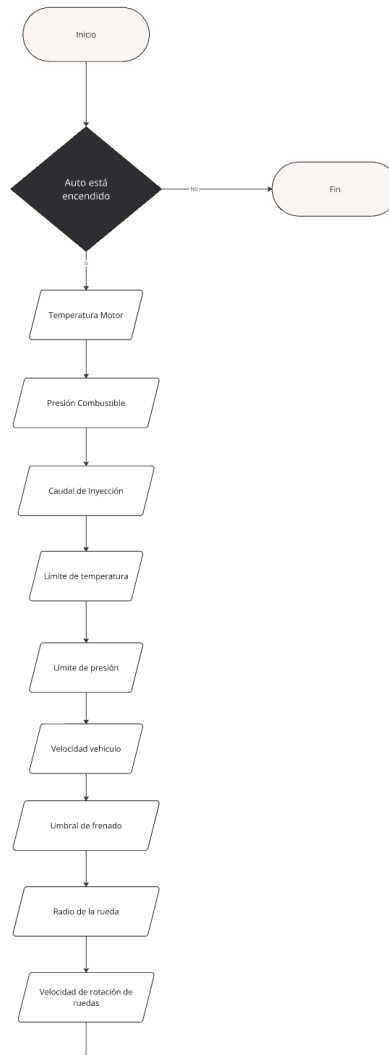


Figura 1. Parte inicial del diagrama de flujo para la simulación de sensores automotrices.

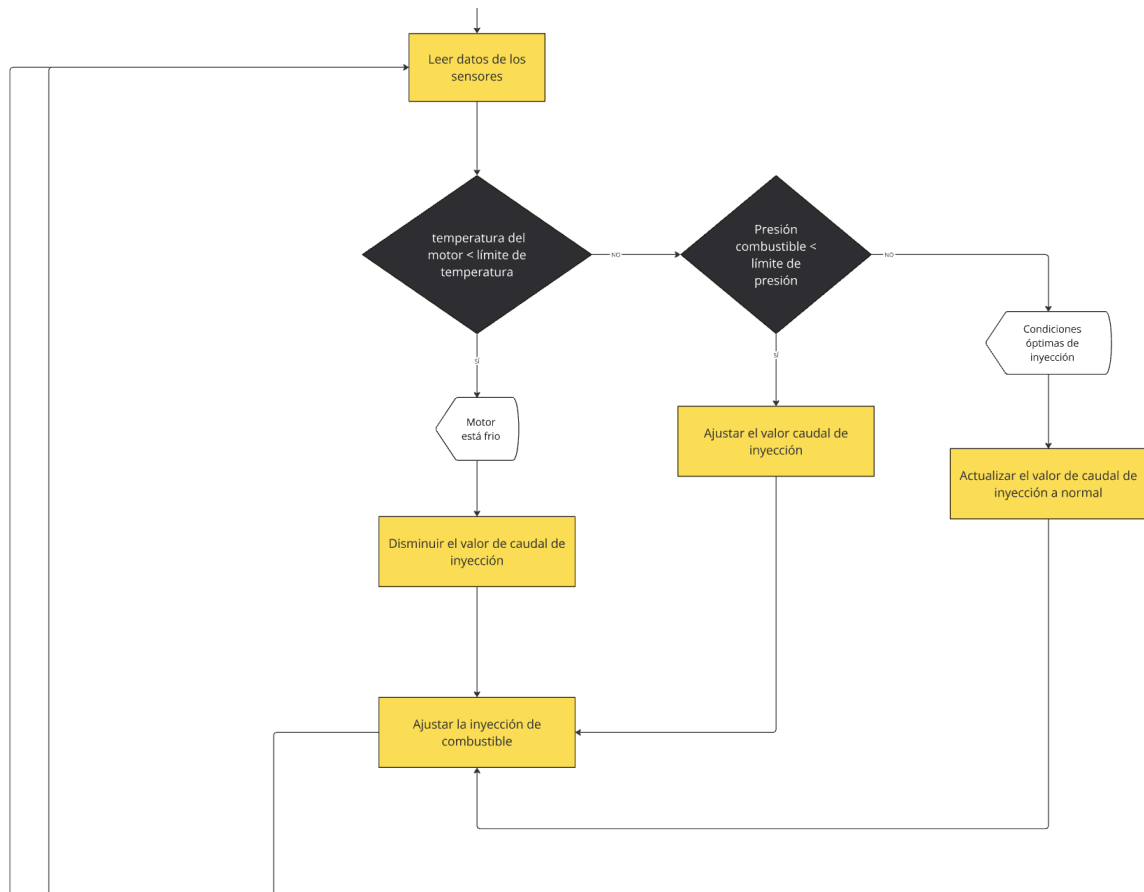


Figura 2. Parte 2 del diagrama de flujo para la simulación de sensores automotrices.

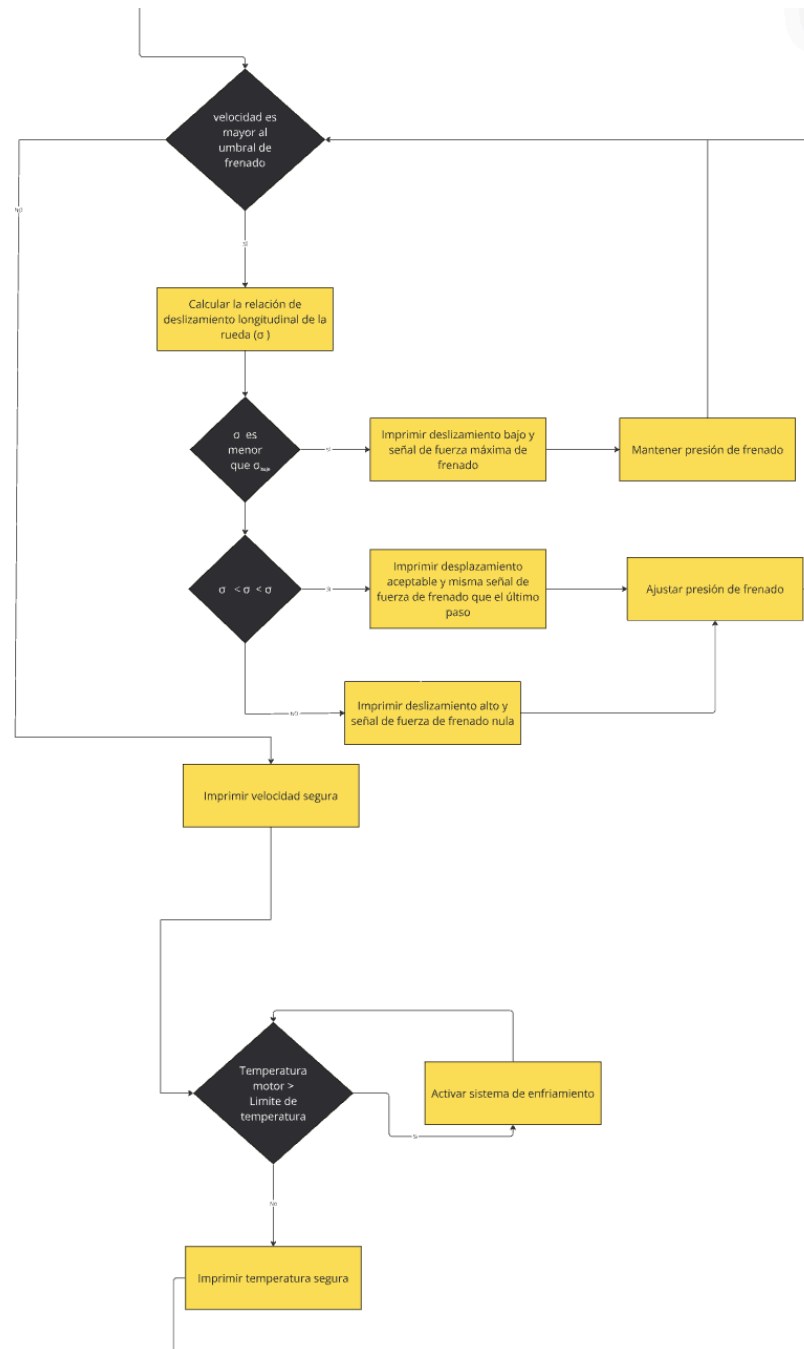


Figura 3. Parte 3 del diagrama de flujo para la simulación de sensores automotrices.

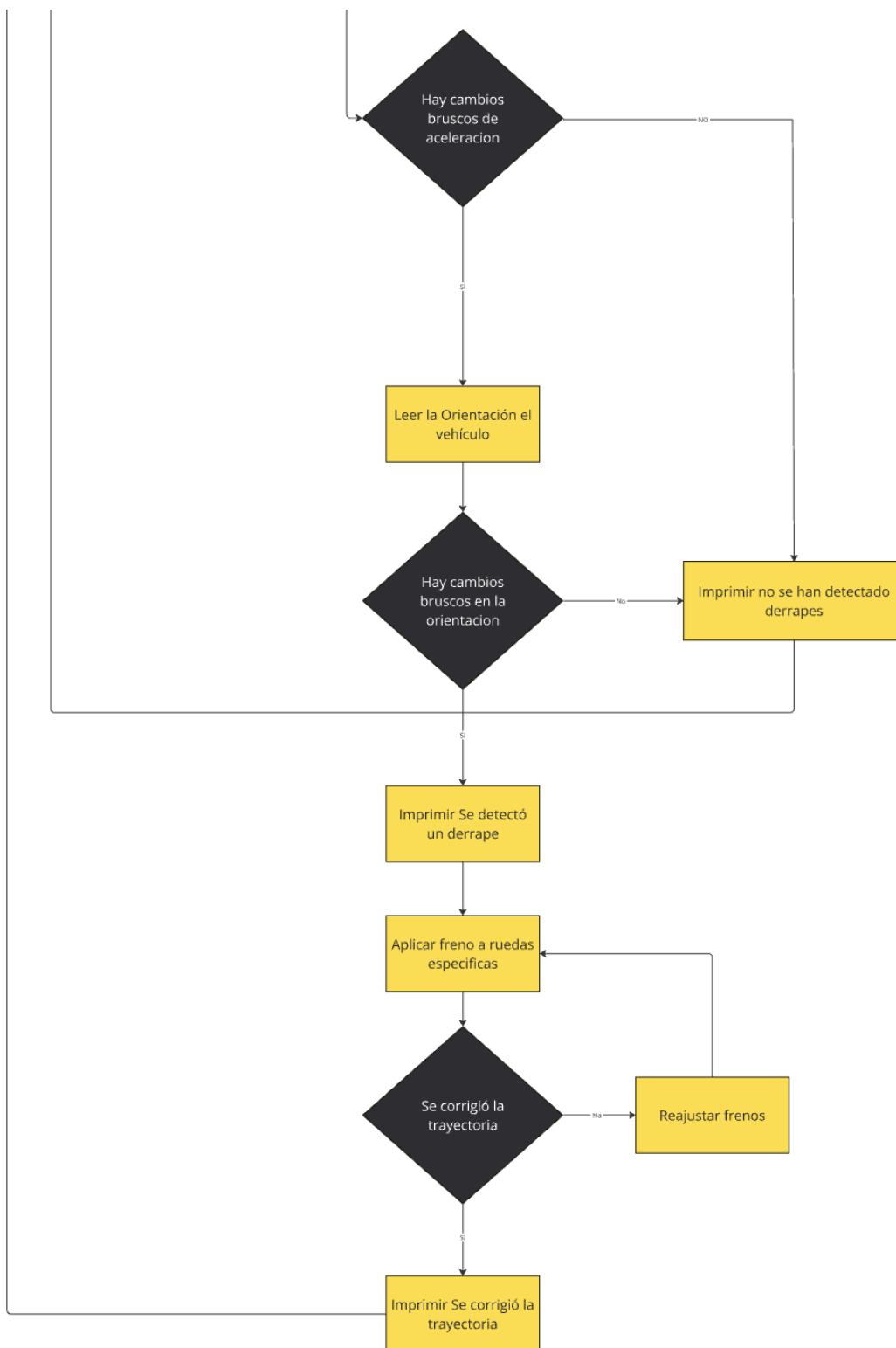


Figura 4. Parte final del diagrama de flujo para la simulación de sensores automotrices.

Relación del diagrama con el programa desarrollado

Inicio del programa:

El flujo del programa comienza con la inicialización, representada en el código por la instrucción MOV SP, stackTop, que configura el puntero de pila para asegurar que el sistema está preparado para comenzar a procesar los valores de los sensores. Esta etapa corresponde al punto inicial del diagrama, donde se prepara el sistema para la toma de decisiones.

Lectura de los sensores:

En el programa, se emplean subrutinas como comparar_sensores para realizar comparaciones de los valores actuales de los sensores con los límites predefinidos. Cada comparación se refleja en el diagrama como un bloque de decisión (diamantes), en el que se evalúan condiciones como si la dirección excede el límite o si la temperatura es mayor a lo permitido. El programa usa las instrucciones CMPB para estas comparaciones, y JBE y JA para definir los saltos condicionales en función del resultado, lo que se traduce en las bifurcaciones del flujo que se ven en el diagrama.

Activación de sistemas:

Si el valor de un sensor supera el límite predefinido, se activa el sistema correspondiente. Por ejemplo, en el programa, cuando la dirección excede el límite, se activa el ESC con la instrucción MOV C, ON_ESC y se muestra el estado con CALL print. Esto se refleja en el diagrama mediante los bloques de acción (amarillos) que indican que un sistema se pone en marcha al excederse los valores límite.

Flujo condicional:

El diagrama presenta rutas de decisión separadas para cada sistema basado en los valores de los sensores. Esto coincide con las diferentes comparaciones del programa, donde cada subsistema como el ESC, ABS, SEN e IYC se evalúa de manera independiente. Por ejemplo, si la temperatura del motor supera el límite, el sistema de inyección de combustible se activa, siguiendo una ruta de decisiones específicas.

Salida del programa:

Después de procesar todos los sensores, el programa imprime el estado de los sistemas utilizando la subrutina print. En el diagrama, esta etapa podría estar representada por un bloque de salida o visualización de resultados, donde se muestra el estado final de cada sistema, ya sea activo o inactivo.

1.5 Desarrollo del programa

Descripción del programa desarrollado

Este programa en ensamblador está diseñado para monitorear y gestionar los sistemas críticos de un vehículo, como parte de un sistema automotriz de control. Los principales objetivos del programa incluyen:

- *Leer y procesar datos de sensores:* El programa detecta si los valores actuales de los sensores exceden los límites predefinidos para cada sistema.
- *Almacenar resultados:* Las variables de salida como OFF_ESC, ON_SEN, etc., se utilizan para representar el estado de los sistemas, ya sea como "ON" (activo) o "OFF" (inactivo).
- *Generar alertas:* El programa compara los valores actuales de los sensores con los umbrales definidos para cada subsistema y, si se superan, activa los sistemas como ESC (Control Electrónico de Estabilidad), SEN (Sistema de Enfriamiento), ABS (Sistema de Frenos Antibloqueo) e IYC (Sistema de Inyección de Combustible).

El flujo principal:

- *Inicialización:* El programa comienza configurando el puntero de pila con la instrucción MOV SP, stackTop. A continuación, llama a la subrutina comparar_sensores para evaluar los valores de los sensores.
- *Comparación:* En la subrutina comparar_sensores, se compara cada sensor con su límite utilizando instrucciones CMPB. Si los valores exceden los límites, se actualizan las variables de salida correspondientes para indicar que un sistema está activado.
- *Visualización:* Finalmente, el programa usa la subrutina print para mostrar en la memoria los resultados de las comparaciones, indicando si los sistemas están activos o inactivos mediante cadenas como ABS-ON o ESC-OFF.

Ejemplos de funcionamiento del programa en situaciones simuladas

A continuación, se presentan cuatro combinaciones de valores de sensores y cómo el programa responde:

Situación 1. Sensor de dirección y velocidad lateral exceden límites (ESC)

Dirección: 35° (límite 30°).

Velocidad lateral: 55 km/h (límite 50 km/h).

Resultado: El sistema ESC se activa para mejorar la estabilidad del vehículo. La salida correspondiente en el display es |ESC-ON|.

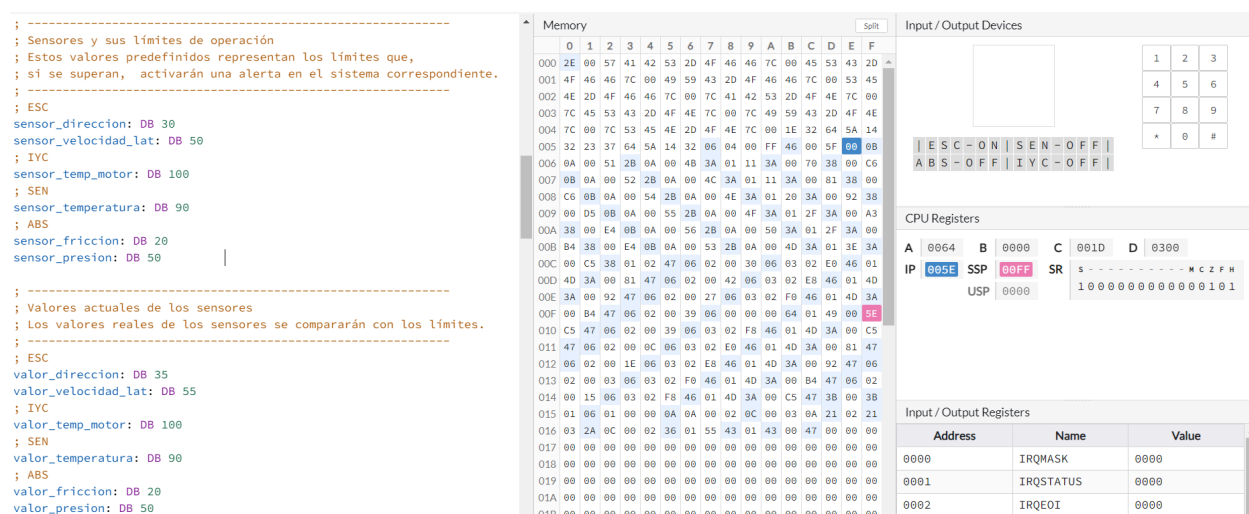


Figura 5. Fragmento del programa que simula la situación 1.

Situación 2: Temperatura del motor alta (IYC activo)

Temperatura del motor: 105°C (límite 100°C)

Resultado: Se activa el sistema de inyección de combustible para evitar sobrecalentamiento. La salida correspondiente es |IYC-ON|.

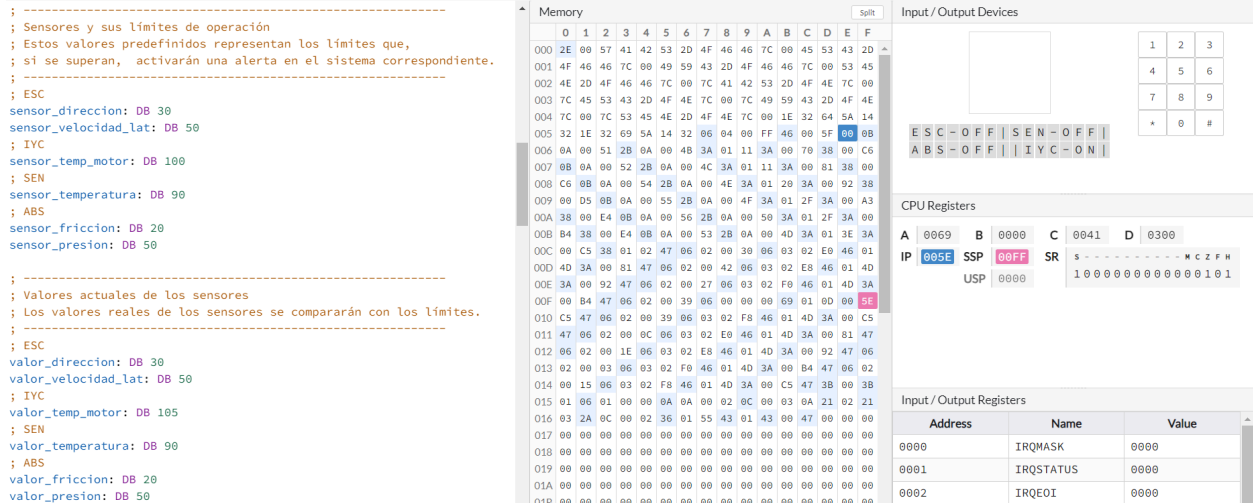


Figura 6. Fragmento del programa que simula la situación 2.

Situación 3: Baja fricción y presión alta (ABS activo)

Fricción: 15% (límite 20%)

Presión: 55 kPa (límite 50 kPa)

Resultado: El sistema ABS se activa para mejorar la respuesta de frenado del vehículo. La salida es |ABS-ON|.

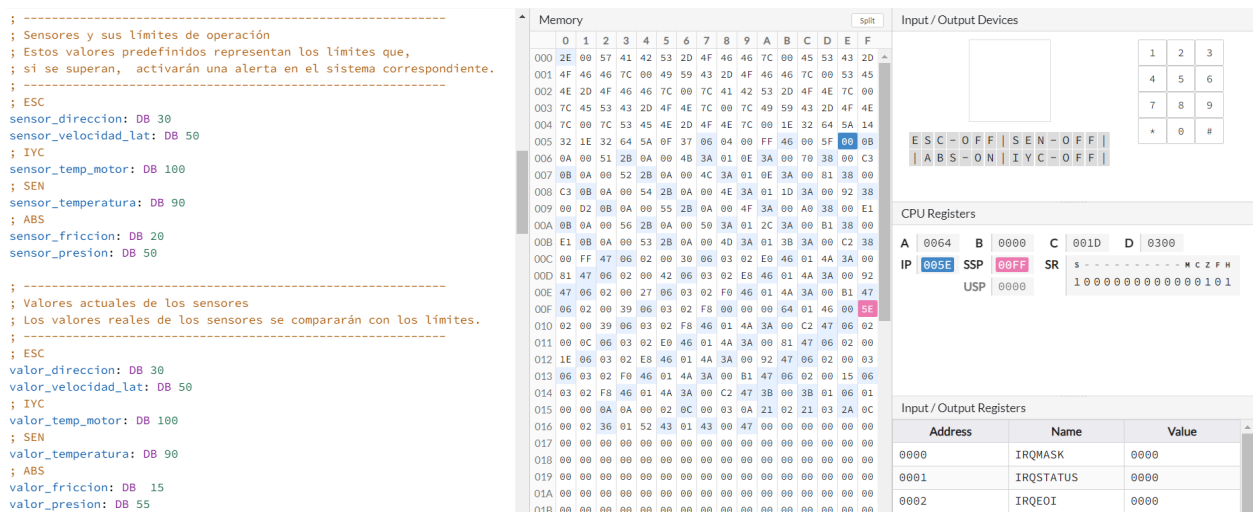


Figura 7. Fragmento del programa que simula la situación 3.

Situación 4: Temperatura ambiente alta (SEN activo)

Temperatura ambiente: 95°C (límite 90°C)

Resultado: El sistema de enfriamiento se activa debido a la temperatura alta. La salida correspondiente es |SEN-ON|.

The screenshot displays a microcontroller simulation environment with the following components:

- Memory:** A table showing memory addresses from 000 to 01B. Address 000 contains the instruction `ESC-0FF`, and address 001 contains `SEN-ON`. Address 002 contains `ABS-0FF` and `IYC-0FF`.
- Input / Output Devices:** A panel showing a 3x3 grid of buttons labeled 1 through 9, with additional buttons for *, 0, and #.
- CPU Registers:** A panel showing the status of various registers: A (0064), B (0000), C (001D), D (0300), IP (005E), SSP (00FF), SR (S), and USP (0000). The SR register is highlighted with a red border.
- Input / Output Registers:** A table with three columns: Address, Name, and Value. It lists registers 0000 (IRQMASK), 0001 (IRQSTATUS), and 0002 (IRQEOI), all with a value of 0000.

Figura 8. Fragmento del programa que simula la situación 4.

Estas situaciones simulan distintos escenarios que un sistema automotriz puede enfrentar, y cómo los sensores detectan condiciones fuera de los límites predefinidos. En cada caso, el programa actualiza la salida de los sistemas correspondientes de "OFF" a "ON", reflejando el estado activo de los sistemas críticos del vehículo.

Resultados

Logros obtenidos

A lo largo del desarrollo del proyecto, logramos una comprensión profunda de cómo programar y estructurar sistemas mínimos utilizando ensamblador. Aunque trabajamos exclusivamente en la parte de software, nos permitió entender a fondo cómo interactúan los programas a bajo nivel con el hardware, particularmente en un contexto automotriz.

Uno de los logros más destacados fue la capacidad de abstraer el funcionamiento de los sistemas de sensores de un vehículo, comprendiendo cómo se recopilan y procesan los datos provenientes de distintos sensores, como velocidad, presión de combustible, y temperatura del refrigerante. Aunque no trabajamos directamente con hardware real, logramos simular el procesamiento de esta información en el microcontrolador, creando un programa en ensamblador que emula la lectura, comparación y respuesta ante diferentes condiciones simuladas, como cambios en la temperatura o variaciones en la velocidad del vehículo.

Desarrollamos subrutinas que gestionan eficientemente la comparación de los valores de los sensores con los umbrales predefinidos, activando alertas cuando las condiciones superan ciertos límites. También, diseñamos e implementamos un sistema para mostrar el estado de los sensores, superando las limitaciones del simulador, que solo permitía la impresión de caracteres individuales. Tuvimos que construir una lógica personalizada para la impresión de cadenas que reportara de forma precisa el estado de los sistemas como el ESC, ABS y SEN.

Este proyecto nos permitió afianzar conocimientos clave sobre programación en ensamblador y sistemas embebidos, entendiendo la importancia de la optimización del código para reducir la latencia y mejorar la eficiencia del sistema, incluso en un entorno simulado. A través de este desafío, obtuvimos una visión clara sobre cómo los sistemas críticos en automóviles dependen de la precisión y la rapidez en el procesamiento de la información, elementos esenciales en el diseño de software para sistemas embebidos.

Desafíos Enfrentados


Durante el desarrollo del proyecto, uno de los mayores desafíos fue la integración de múltiples sensores en un sistema funcional utilizando un simulador con un conjunto limitado de instrucciones. Las instrucciones disponibles, como CMPB y los saltos condicionales, restringían las operaciones que podíamos realizar, lo que complicó la construcción de un flujo eficiente para la comparación de los sensores (como velocidad de las ruedas, presión de combustible, temperatura del refrigerante y ángulo de deslizamiento) sin generar retrasos.

Otro reto significativo fue estructurar los ciclos de comparación y de impresión debido a las limitaciones mencionadas. El simulador solo permitía procesar un carácter a la vez en la salida, lo que complicó la creación de subrutinas eficientes para imprimir cadenas completas. Esto requirió la implementación de un manejo manual de cada carácter, lo cual incrementó la complejidad del código.

Adicionalmente, surgieron conflictos al nombrar las subrutinas. Algunos nombres de subrutinas iniciales generaban errores o conflictos con las palabras reservadas del simulador, lo que nos obligó a renombrarlas y ajustar su funcionalidad sin afectar la lógica del programa. Adaptar el programa para responder de manera precisa a diferentes condiciones del vehículo, como variaciones en la velocidad y temperatura, también implicó un ajuste constante de los algoritmos y la optimización de la lógica de control, asegurando que se mantuviera una baja latencia en el procesamiento de los datos.

Lecciones Aprendidas

A lo largo del desarrollo del proyecto, adquirimos un entendimiento profundo sobre la programación en ensamblador y su aplicación en sistemas automotrices. Aprendimos la importancia de estructurar y optimizar el código para que opere de manera eficiente dentro de las limitaciones de hardware. En sistemas críticos como los de un vehículo, donde la rapidez en el procesamiento de datos es vital, cada ciclo de reloj y cada instrucción optimizada cuentan. La



gestión eficiente de los recursos, tanto de memoria como de tiempo de ejecución, se volvió un aspecto clave al escribir un programa que debe interactuar con múltiples sensores en tiempo real.

Una de las lecciones más valiosas fue la importancia de diseñar con precisión los ciclos de comparación y decisión en los sistemas críticos, garantizando que el programa funcione dentro de los tiempos establecidos para evitar retrasos en la activación de sistemas como el ESC o el ABS. Nos enfrentamos a las limitaciones inherentes al simulador, lo que nos obligó a ser creativos en la estructuración del código, maximizando la eficiencia y asegurando que cada instrucción fuera utilizada de manera óptima.

Asimismo, el proyecto subrayó el valor de las pruebas exhaustivas y la simulación de entornos antes de cualquier implementación física. Simular escenarios diversos nos permitió identificar posibles fallos, ajustar algoritmos y mejorar la respuesta del sistema de manera segura, sin necesidad de realizar pruebas directas en un vehículo real. Esto refuerza la lección de que la simulación no solo es una herramienta indispensable en la fase de desarrollo, sino que también contribuye a la reducción de riesgos en sistemas críticos.

Conclusión

Este proyecto nos permitió implementar una solución viable para mejorar la funcionalidad de sistemas automotrices críticos mediante la integración de sensores. Los resultados demostraron que es posible optimizar el rendimiento del vehículo y mejorar la seguridad del conductor a través de la incorporación de algoritmos de control avanzados y sensores de alta precisión. La simulación del comportamiento de los sensores ofreció una visión profunda del papel que juegan estos dispositivos en la toma de decisiones en tiempo real, y cómo su correcta programación puede prevenir situaciones de riesgo.

En resumen, el proyecto destaca la relevancia de la tecnología de sensores en la industria automotriz moderna, y subraya la necesidad de seguir innovando en el desarrollo de soluciones que mejoren tanto la seguridad como la eficiencia energética de los vehículos.

Referencias

- Universidad Europea. (2024, 1 febrero). Últimos avances en la industria automotriz. <https://universidadeuropea.com/blog/industria-automotriz/>
- Den. (2023, 22 diciembre). Sistema de frenos: dispositivo, explicación, componentes. <https://club.autodoc.es/magazin/sistema-de-frenos-dispositivo-explicacion-componentes>
- renault. (2022, 23 septiembre). ¿Qué es la inyección de combustible y cuáles son los tipos? Renault. <https://www.renault.com.mx/blog/tips/inyeccion-de-combustible.html>
- ¿Qué es un sistema de enfriamiento automotriz y cómo funciona? | Grupo Herres. (2019, 3 julio). Grupo Herres. <https://www.grupoherres.com.mx/sistema-de-enfriamiento/>
- 16-bit Assembler Simulator — asm-simulator 1.3.0 documentation. (2017.). <https://asm-simulator.readthedocs.io/en/latest/index.html>
- Frenos ABS: Todo lo que Debes Saber | Jeep® México. (s. f.). <https://www.jeep.com.mx/blog/experiencia-jeep/frenos-abs-todo-lo-que-debes-saber-de-su-funcionamiento.html>
- Motors, M. (2023, 4 julio). 6 sistemas de freno disponibles en una camioneta | Mitsubishi Motors. Mitsubishi Motors Blog | Venta de Camionetas SUV & MPV. <https://www.mitsubishi-motors.com.pe/blog/sistemas-freno-camioneta/>

Anexos

Programa de ensamblador:

```
; -----  
; El programa compara los valores actuales de varios sensores con límites predefinidos para determinar el  
; estado de cada sistema (ENCENDIDO o APAGADO). Luego, actualiza las pantallas de salida con los  
; estados de los sistemas.  
; Sistemas controlados:  
; - ABS (Sistema de Frenos Antibloqueo)  
; - ESC (Control Electrónico de Estabilidad)  
; - IYC (Sistema de Inyección de Combustible)  
; - SEN (Sistema de Enfriamiento)  
; -----
```

```

; -----
; Puntos de inicio de variables y direcciones clave
; -----

        JMP start

stackTop EQU 0xFF
txtDisplay1 EQU 0x2E0
txtDisplay2 EQU 0x2E8
txtDisplay3 EQU 0x2F0
txtDisplay4 EQU 0x2F8

; -----
; Variables de salida para los sistemas (Indicadores de estado)
; Estas cadenas representan el estado actual de cada sistema.
; Incluyen indicadores "ON" y "OFF" para mostrar si el sistema
; está activo o inactivo.
; -----
OFF_ABS:    DB "ABS-OFF|"
             DB 0

OFF_ESC:    DB "ESC-OFF|"
             DB 0

OFF_IYC:    DB "IYC-OFF|"
             DB 0

OFF_SEN:    DB "SEN-OFF|"
             DB 0

ON_ABS:     DB "|ABS-ON|"
             DB 0

```

ON_ESC: DB "|ESC-ON|"
DB 0

ON_IYC: DB "|IYC-ON|"
DB 0

ON_SEN: DB "|SEN-ON|"
DB 0

; -----
; Sensores y sus límites de operación
; Estos valores predefinidos representan los límites que,
; si se superan, activarán una alerta en el sistema correspondiente.

; -----
; ESC

sensor_direccion: DB 30
sensor_velocidad_lat: DB 50

; IYC
sensor_temp_motor: DB 100

; SEN
sensor_temperatura: DB 90

; ABS
sensor_friccion: DB 20
sensor_presion: DB 50

; -----
; Valores actuales de los sensores
; Los valores reales de los sensores se compararán con los límites.

; -----
; ESC

valor_direccion: DB 35
valor_velocidad_lat: DB 55

; IYC


```

valor_temp_motor: DB 105
; SEN
valor_temperatura: DB 95
; ABS
valor_friccion: DB 25
valor_presion: DB 55

; -----
; Punto de inicio del programa
; Configura el stack y realiza las comparaciones.
; -----
start:
    MOV SP, stackTop
    CALL comparar_sensores
    ;JMP boot (esto es para los "LEDs")
    HLT

; -----
; Subrutina: comparar_sensores
; Compara los valores actuales de los sensores con los límites
; predefinidos.
; Si un valor excede su límite, se activa una alerta (estado "ON"),
; de lo contrario, se muestra el estado "OFF".
; -----
comparar_sensores:
    MOVB AL, [valor_direccion]
    CMPB AL, [sensor_direccion]
    JBE .ESC_alertOFF
    JBE .skip_ESC
    JA .ESC_alertON

.skip_ESC:
    MOVB AL, [valor_velocidad_lat]

```

```
CMPB AL, [sensor_velocidad_lat]
JBE .ESC_alertOFF
    JBE .skip_ESC2
    JA .ESC_alertON

.skip_ESC2:
    MOVB AL, [valor_temperatura]
    CMPB AL, [sensor_temperatura]
    JBE .SEN_alertOFF
    JBE .skip_SEN
    JA .SEN_alertON

.skip_SEN:
    MOVB AL, [valor_friccion]
    CMPB AL, [sensor_friccion]
    JBE .ABS_alertOFF
    JBE .skip_ABS
    JA .ABS_alertON

.skip_ABS:
    MOVB AL, [valor_presion]
    CMPB AL, [sensor_presion]
    JBE .ABS_alertOFF
    JBE .skip_ABS2
    JA .ABS_alertON

.skip_ABS2:
    MOVB AL, [valor_temp_motor]
    CMPB AL, [sensor_temp_motor]
    JBE .IYC_alertOFF
    JBE .skip_IYC
    JA .IYC_alertOn
```



.skip_IYC:

RET

.ESC_alertON:

MOV C, ON_ESC

MOV D, txtDisplay1

CALL print

JBE .skip_ESC2

RET

.SEN_alertON:

MOV C, ON_SEN

MOV D, txtDisplay2

CALL print

JBE .skip_SEN

RET

.ABS_alertON:

MOV C, ON_ABS

MOV D, txtDisplay3

CALL print

JBE .skip_ABS2

RET

.IYC_alertON:

MOV C, ON_IYC

MOV D, txtDisplay4

CALL print

JBE .skip_IYC

RET

.IYC_alertOn:

MOV C, ON_IYC

```
MOV D, txtDisplay4
CALL print
JBE .skip_IYC
RET
```

.ESC_alertOFF:

```
MOV C, OFF_ESC
MOV D, txtDisplay1
CALL print
JBE .skip_ESC2
RET
```

.SEN_alertOFF:

```
MOV C, OFF_SEN
MOV D, txtDisplay2
CALL print
JBE .skip_SEN
RET
```


.ABS_alertOFF:

```
MOV C, OFF_ABS
MOV D, txtDisplay3
CALL print
JBE .skip_ABS2
RET
```

.IYC_alertOFF:

```
MOV C, OFF_IYC
MOV D, txtDisplay4
CALL print
JBE .skip_IYC
RET
```

; -----



```
; Subrutina 'print'
; Imprime el contenido de las variables de salida en la dirección
; de memoria especificada en D. Termina cuando encuentra un 0.
; -----
```

```
print:
```

```
    PUSH A
    PUSH B
    MOV B, 0
```

```
.loop:
```

```
    MOVB AL, [C]
    MOVB [D], AL
    INC C
    INC D
    CMPB BL, [C]
    JNZ .loop
```

```
    POP B
    POP A
    RET
```