Universidad de Costa Rica

Facultad de Ingeniería

Escuela de Computación e Informática

Proyecto Integrador de Arquitectura y Ensamblador

CI-0119

Grupo D

Avance 4 del Panel De Instrumentos de un Vehículo

Profesor:

Francisco Arroyo Mora

Elaborado por:

Pérez Morera Daniel C15906

Sánchez Zeledón Ignacio C17326

Rojas Sancho Josué C16882

Sancho Vásquez Randy Yasiel C17344

30 de noviembre del 2022

Tema:

Este proyecto consiste en simular un panel de instrumentos de un vehículo. Este panel de instrumentos requerirá de una arquitectura en base a C++ con Qt como interfaz y lenguaje ensamblador y de una serie de instrucciones que ofrezcan información en tiempo real para controlar u observar datos con distintos sensores, como la velocidad del vehículo (velocímetro), presión de neumáticos, temperatura del vehículo, combustible disponible, estado de encendido/apagado, luces direccionales(Izquierda, Derecha), luces frontales (altas/bajas) y un sensor anti colisiones.

Panel de Instrumentos de un Vehículo:

Este panel de instrumentos se utilizará para controlar y alertar sobre el estado de un vehículo en tiempo real. Dicho panel recibe lecturas recolectadas por varios sensores del vehículo, entre los cuales se incluyen: velocímetro, presión de neumáticos, temperatura de vehículo, cantidad de combustible, direccionales, luces frontales, luces de emergencia, encendido/apagado y sensor anticolisiones. El monitor toma los valores recibidos por estos sensores, los muestra y adicionalmente puede activar una alarma en caso de que se detecte que los valores reportados excedan o no lleguen a un umbral específico.

Para poder simular todos los sensores hemos decidido implementar unas funciones las cuales toman decisiones mediante un random y semi random. En el caso del semi random se utiliza un random para obtener un número aleatorio, después a partir de la velocidad se asignan las posibilidades de las posibles rutas de simulación para determinar con el anterior número random cual cambio se realizará en una o varias simulaciones, un ejemplo para esto es la temperatura del vehículo, ya que entre más rápido vaya el vehículo es más probable que la temperatura incremente. Ahora para la implementación de funciones randoms lo que se hace es solicitar un número random, despues de ahi ese número se compara por alguna condición para a partir de ahí definir un comportamiento

de la simulación, un ejemplo de esto es el sensor anticolisión el cual solicita números randoms y activa su sensor hasta que uno de esos números sean igual a 1.

Requerimientos Funcionales:

1. El panel de instrumentos muestra en tiempo real los valores recibidos por cada sensor conectado (temperatura, combustible, luces delanteras, velocímetro, luces direccionales, sensor anti colisiones, presión de las llantas, sensor de encendido).
2. El panel de instrucciones permite la definición de umbrales superiores e inferiores para la activación de alarmas para los sensores de temperatura, combustible, sensor anti colisiones y presión de las llantas. Y estos umbrales están definidos por el desarrollador, que están tomados con base a datos reales.
3. Cuando se activa una alarma porque se excedieron los umbrales, se muestra en la interfaz gráfica una luz de color rojo para cada sensor correspondiente.
4. Se cuenta con un botón de encendido/apagado general para el estado del vehículo a excepción de las luces frontales, las cuales podrán utilizarse sin importar si está encendido o apagado. Y además las luces se podrán utilizar mediante un botón en pantalla.
5. Para el sensor de temperatura se utiliza la unidad de medida de grados centígrados, se mostrará la temperatura actual del vehículo en tiempo real en base a la velocidad del vehículo. Además contará con una alarma que se activará en el momento en que la temperatura sobrepase el umbral máximo o descienda del umbral mínimo establecido en el Anexo 1.
6. El sensor de combustible muestra la cantidad de litros con los que cuenta el vehículo. Por otra parte, cuenta con alertas que se activarán para alertar al usuario cuando el vehículo tenga una cantidad demasiado baja de combustible (umbral mínimo).
7. El sensor de velocidad (velocímetro) utiliza la unidad de medidas de Kilómetros por hora, este mostrará al usuario la velocidad del vehículo en tiempo real. Y la velocidad podrá ser cambiada por el usuario mediante un slider que posee el mismo programa.

8. La presión de las llantas será medido mediante PSI (libras por pulgada cuadrada) , además de poseer un mínimo y un máximo, y una alerta que saltará si la presión excede el umbral o desciende del umbral.

9. Los sensores de luces delanteras cuentan con luces altas y bajas, luces direccionales y sensor de encendido y apagado, y luces de emergencia, cuenta con luces en el panel, dependiendo de su estado(encendido o apagado). El usuario será quien decida su estado.

10. El sensor de anti colisiones funciona de manera que el programa genera un número random, y el sensor activará una alerta con una luz roja si este es menor a uno.

Requerimientos De Arquitectura:

1. Para el diseño inicial del panel de instrumentos a nivel de hardware se usó la herramienta LOGISIM y la implementación se simuló utilizando programación híbrida: un lenguaje de alto nivel C++ y Ensamblador(x86).

2. El despliegue de los gráficos del panel de instrumentos, así como la selección y configuración de los sensores a desplegar se programaron en un lenguaje de alto nivel en Qt que está basado en el lenguaje C++.

3. La recolección de lecturas de los sensores, implicaciones de umbral mínimo, umbral máximo hechos por el panel de instrumentos, se realizó en lenguaje ensamblador.

4. El código para crear la simulación de los valores obtenidos para los sensores se creó en lenguaje ensamblador, específicamente las funciones para temperatura, gasolina, las alertas y los umbrales.

5. Para poder intercambiar información entre los sensores de bajo nivel y el panel de instrumentos se implementaron funciones pensadas para que puedan recibir valores como la velocidad y gasolina, y retornar un resultado al panel implementado en alto nivel.

Problemas del proyecto:

Hubo muchas dificultades al momento de unir Qt con las funciones en ensamblador, ya que las funciones funcionaban correctamente al momento de probarlas, pero al momento de enlazarlas con la interfaz gráfica de Qt daban muchos errores.

El orden de los archivos en con qt es muy complicado manipularlo y por ende no se logró obtener un directorio relativamente organizado.

También existieron complicaciones al intentar hacer parpadear las luces de las alertas ya que no se podía mantener en un solo proceso para las luces porque eso detendría a todos los demás sensores.

Soluciones de algunos problemas:

Tuvimos que crear las variables estáticas, para así evitar el cambio de valor que estaba ocurriendo , que hacía que las variables retornarán a su valor definido como base. Y que además algunas funcionaban perfectamente si eran estáticas y otras no.

Las funciones de presión de las llantas y del sistema de anti colisiones tuvimos que crearlas en C++ debido a que por alguna razón daba problemas en la interfaz.

Para el tema de que se realizará todo en tiempo real utilizamos un Timer proveniente de la librería de Qt, para que así ejecutara todo cada segundo.

Mejoras diseño pasado :

- Se personalizo el color de las letras y fondo de la interfaz
- Se agregó sonido a las alertas producidas por sobrepasar los umbrales.

Anexo:

1. Umbrales del dispositivo

	Temperatura (°C)(°F)	**Presion de Neumaticos (PSI)(Lb/cm^2)**	**Combustible (L)**	**Sensor de anticolisiones (m)**
Umbral Mínimo	(70°)(158°)	(27)(384)	5	1
Umbral Máximo	(150°)(302°)	(33)(469)	63	