UNIVERSIDAD SERGIO ARBOLEDA

SISTEMAS EMBEBIDOS

INGENIERÍA ELECTRÓNICA

WENDY JOHANNA CHAPARRO – CRISTOPHER RAMÍREZ RAMÍREZ

INFORME DE LABORATORIO #6

En este laboratorio se diseñó e implementó un protocolo de transmisión con el objetivo de enviar los datos de las RPM (revoluciones por minuto) para su visualización en una interfaz gráfica a través de QT Creator. Para esta tarea, se utilizó la placa de desarrollo STM32F411 y el entorno de desarrollo STM32CubeIDE, herramientas que permitieron la configuración, envío y recepción de los datos a través de un protocolo personalizado.

La placa STM32F411 se configuró para enviar y recibir los datos de RPM a través del puerto serial. En la aplicación QT Creator, se desarrolló un programa en C++ que permitió la visualización y el control de los datos recibidos para el monitoreo en tiempo real de las RPM.

**SENSORES Y MODULOS IMPLEMENTADOS:**

* **Sensor infrarrojo de herradura KLH512:** El sensor de herradura KLH512 es un sensor fotoeléctrico que responde al cambio en la intensidad de la luz. Estos sensores requieren de un componente emisor que genera la luz, y un componente receptor que percibe la luz generada por el emisor. Están diseñados especialmente para la detección, clasificación y posicionado de objetos; la detección de formas, colores y diferencias de superficie, incluso bajo condiciones ambientales extremas. Los sensores de luz se usan para detectar el nivel de luz y producir una señal de salida representativa respecto a la cantidad de luz detectada. [1]
* **Modulo puente H TB6612FNG:** Es una pequeña placa integrada por un circuito integrado TB6612FNG que permite tener la manipulación de motores de DC. El controlador TB6612FNG es un driver de motores que puede controlar hasta dos motores DC a una corriente constante de 1.2 Amp .  Las dos salidas de motor (A y B) pueden controlarse por separado, la velocidad de cada motor se controla con una señal de entrada PWM de hasta 100 kHz. Este controlador integra en su placa dos señales de entradas IN1 y IN2, que se utilizan para controlar el motor en uno de los cuatro modos, como CW, CCW, corto freno y modo de parada. [2]

**ANÁLISIS Y DESARROLLO:**

Configuración de la placa STM32F411 mediante el software STM32CubeIDE: se configuró el sistema para enviar los datos de RPM con un protocolo diseñado para enviar los valores de RPM en formato hexadecimal.

Desarrollo de la interfaz en QT Creator: Se creó una interfaz gráfica simple que permite al usuario visualizar las RPM y controlar el motor con botones para encender y detener el motor. El código en QT Creator se encargó de la recepción y procesamiento de los datos seriales enviados por la placa STM32. A continuación, se describe el funcionamiento de los elementos clave que se utilizaron:

* **Interfaz de usuario (UI):** La aplicación utiliza un diseño sencillo donde se muestra la información de las RPM. Se emplea una etiqueta (ui->label\_RPM) para mostrar el valor actual de las RPM, y botones que permiten enviar comandos para el control del motor.
* **Control del Motor:** El sistema incluye funciones para el control del motor a través de comandos específicos enviados por el puerto serial. Las funciones on\_set\_RPM\_clicked(), on\_add\_clicked(), y on\_off\_clicked() envían comandos codificados en bytes (como 0x31 para girar el motor a la izquierda, 0x30 para girar el motor a la derecha y 0x1B para apagar el motor).
* **Inicialización del Puerto Serial:** En el constructor del widget principal, se inicializa el puerto serial y se identifican todos los puertos disponibles en el sistema, mostrándolos en un QComboBox para permitir la selección por parte del usuario. Además, se configura el puerto serial con las propiedades necesarias para la transmisión de datos a 115200 baudios y con 8 bits de datos sin paridad ni control de flujo.
* **Lectura de Datos:** La función readyRead() se activa cuando hay datos disponibles en el buffer del puerto serial. Los bytes recibidos se procesan para reconstruir el valor de RPM. Los datos son empaquetados y transmitidos desde la STM32 en un formato específico que el programa en QT Creator decodifica. Los bytes se procesan en las posiciones específicas para generar el valor de RPM, que luego se muestra en la interfaz gráfica.
* **Protocolo de comunicación:** El protocolo implementado en el sistema embebido envía los datos en bloques de bytes que son decodificados en la aplicación QT Creator, lo que permite visualizar el valor de las RPM y proporcionar retroalimentación visual.

**RESULTADOS:**

La implementación fue exitosa, permitiendo la transmisión eficiente de datos entre el sistema embebido y la interfaz gráfica. Las RPM del motor fueron visualizadas en tiempo real, y el control del motor se logró mediante comandos enviados desde la interfaz gráfica.

Calendario

Descripción generada automáticamente

**CONCLUSIONES:**

1. La interfaz gráfica desarrollada en QT Creator resultó ser sencilla y eficaz, permitiendo no solo la visualización clara de los datos de RPM, sino también el control del motor mediante botones que envían comandos específicos al sistema embebido.
2. Es importante conocer el funcionamiento y aplicación de clases (programación orientada a objetos) para este tipo de proyectos ya que son una herramienta demasiado útil que permite hacer el trabajo mucho más eficiente.
3. El manejo de los tipos de datos es fundamental para el desarrollo exitoso de estas prácticas porque, al no tener claro cómo funcionan, se vuelve una tarea demasiado difícil acciones como controlar un motor DC A mediante comandos seriales enviados desde la interfaz gráfica o recibir un dato.

**BIBLIOGRAFÍA:**

[1] «Sensor de Herradura KLH512 - Moviltronics», *Moviltronics*, 3 de mayo de 2024. Disponible en: <https://moviltronics.com/tienda/sensor-klh512/>

[2] «Doble Puente H TB6612FNG», *UNIT ELECTRONICS*. Disponible en: <https://uelectronics.com/producto/doble-puente-h-tb6612fng/?srsltid=AfmBOor2_Fn69ubd_3kAXtR1Zh-oOXWBrBFq8dmfOs97bHHpi7R7ogZn>

[3] alldatasheet.com, «TB6612FNG PDF». Disponible en: <https://www.alldatasheet.com/datasheet-pdf/view/807693/TOSHIBA/TB6612FNG.html>