Reporte de Proyecto: Sistema de Control de Nivel de Agua con Microcontrolador AVR

Autor: Eduardo Gonzalez Gonzalez

Resumen Ejecutivo

Este reporte describe el diseño e implementación de un sistema automatizado para el control del nivel de agua en un tinaco, desarrollado íntegramente por Eduardo Gonzalez Gonzalez. El sistema utiliza un microcontrolador AVR para monitorear el estado de un sensor de nivel y activar o desactivar una bomba de agua de manera eficiente. La funcionalidad principal radica en su capacidad para encender la bomba cuando el tinaco está vacío y apagarla una vez que ha alcanzado su capacidad máxima, garantizando un llenado automático y sin supervisión. El diseño del código se enfoca en la reactividad y la eficiencia energética, empleando interrupciones de hardware para una respuesta instantánea a los cambios en el nivel del agua. Este proyecto demuestra una sólida comprensión de la programación de sistemas embebidos y la interacción directa con el hardware para resolver problemas de automatización en el mundo real.

1. Introducción

La gestión eficiente de recursos hídricos es un aspecto crítico en diversas aplicaciones, desde sistemas domésticos hasta procesos industriales. La automatización del llenado y vaciado de tanques de almacenamiento de agua no solo optimiza el consumo, sino que también previene desbordamientos o la operación en seco de las bombas, lo que podría llevar a daños costosos. En este contexto, el presente proyecto se centra en el desarrollo de un sistema de control de nivel de agua utilizando un microcontrolador AVR, diseñado para operar de manera autónoma y fiable. Como único diseñador de este código, mi objetivo fue crear una solución que fuera simple en su concepción, pero robusta en su ejecución, aprovechando las capacidades de bajo nivel de los microcontroladores para una respuesta en tiempo real.

Este documento detalla la arquitectura del código, los componentes de hardware

involucrados, la lógica de funcionamiento basada en interrupciones y las decisiones de
diseño que contribuyeron a la funcionalidad y eficiencia del sistema. Se destacará cómo la implementación de un control basado en eventos permite al sistema operar con un consumo mínimo de recursos, reaccionando únicamente cuando un cambio en el nivel del agua lo requiere. Este proyecto es una clara demostración de mi habilidad para traducir un requisito funcional en una solución de hardware y software embebido efectiva y optimizada.

2. Arquitectura y Componentes del Sistema

El sistema de control de nivel de agua se basa en una arquitectura sencilla pero efectiva, que integra un microcontrolador AVR con un sensor de nivel y un actuador (bomba de agua). La interacción entre estos componentes es fundamental para la automatización del proceso de llenado.

2.1. Componentes de Hardware Controlados

El código está específicamente diseñado para interactuar con los siguientes elementos de hardware:

- **Microcontrolador AVR:** El cerebro del sistema, encargado de ejecutar el código y gestionar la lógica de control. Se asume un microcontrolador de la familia AVR (como un ATmega328P, común en plataformas como Arduino, aunque el código es genérico para AVR) debido a su eficiencia y capacidad para manejar interrupciones de hardware.
- **Sensor de Nivel:** Conectado al pin `PD2` (definido como `INT_PIN`). Este sensor es el encargado de detectar el estado del nivel de agua en el tinaco. Su funcionamiento se basa en proporcionar un cambio de estado lógico (por ejemplo, de alto a bajo o viceversa) cuando el nivel del agua cruza un umbral predefinido. Comúnmente, se utilizan sensores de flotador o de tipo capacitivo para esta aplicación.
- **Bomba de Agua:** Controlada a través del pin `PB0` (definido como `MOTOR_PIN`). Este pin actúa como una señal de control para un circuito de potencia (como un relé o un módulo de driver de motor) que, a su vez, activa o desactiva la bomba de agua. Cuando el pin `PB0` está en estado alto, la bomba se enciende; cuando está en estado bajo, la bomba se apaga.

2.2. Estructura del Código

El código está organizado en varias secciones lógicas, cada una con una responsabilidad específica:

- **Directivas de Preprocesador:** Incluyen librerías estándar de AVR (`avr/io.h` , `avr/interrupt.h`) y la definición de la frecuencia del CPU (`F_CPU`). También se definen los pines de control (`MOTOR_PIN` , `INT_PIN`) y una variable volátil (`motor_estado`) para gestionar el estado de la bomba.
- **Rutina de Servicio de Interrupción (ISR):** `ISR(INT0_vect)` es la función que se ejecuta automáticamente cuando se detecta un cambio en el pin del sensor de nivel. Contiene la lógica principal para activar o desactivar la bomba.
- **Configuración:**
Funciones de
  - `configurar_interrupcion()` : Configura la interrupción externa `INT0` para que se active con cualquier cambio lógico en el pin del sensor.
  - `configurar_puertos()` : Inicializa los pines del microcontrolador, configurando `MOTOR_PIN` como salida y `INT_PIN` como entrada con pull up interno.
- **Función de Inicialización (`inicio()`):** Llama a las funciones de configuración de puertos e interrupciones.
- **Función Principal (`main()`):** Contiene el bucle infinito que mantiene el microcontrolador en funcionamiento, esperando las interrupciones.

3. Lógica de Funcionamiento y Diseño Funcional

El corazón de este sistema reside en su lógica de control basada en interrupciones, lo que garantiza una respuesta inmediata y un uso eficiente de los recursos del microcontrolador. Como diseñador de este código, mi enfoque principal fue crear un sistema reactivo y de bajo consumo.

3.1. Proceso de Inicialización

Al encender el microcontrolador, la función `inicio()` se ejecuta para preparar el sistema:

1. **Configuración de Puertos:** El pin `PB0` (conectado a la bomba) se configura como una salida digital y se establece inicialmente en un estado bajo, asegurando que la bomba permanezca apagada al inicio. El pin `PD2` (conectado al sensor de nivel) se configura como una entrada digital y se habilita su resistencia pull-up interna. Esta configuración es crucial, ya que muchos sensores de nivel de tipo interruptor funcionan cerrando un circuito a tierra, lo que provoca un cambio de estado de alto a bajo en el pin de entrada.
2. **Configuración de Interrupción Externa:** La interrupción externa `INT0` se configura para que se dispare con *cualquier cambio lógico* en el pin `PD2` . Esto significa que la interrupción se activará tanto cuando el sensor pase de un estado bajo a alto como de alto a bajo. Esta sensibilidad a ambos flancos de la señal es vital para detectar tanto el vaciado como el llenado del tinaco. Finalmente, las interrupciones globales se habilitan, permitiendo que el microcontrolador responda a los eventos externos.

3.2. Lógica de Control Basada en Interrupciones

La funcionalidad central del sistema se encuentra en la Rutina de Servicio de Interrupción (ISR) asociada a `INT0` . Esta ISR se ejecuta automáticamente y de forma prioritaria cada vez que el sensor de nivel cambia de estado:

- **Detección de Nivel Lleno (o Suficiente):** Si el pin `INT_PIN` se detecta en estado ALTO (`PIND & (1 << INT_PIN)`), esto indica que el tinaco ha alcanzado el nivel deseado (es decir, está lleno). En respuesta, la variable `motor_estado` se actualiza a `0` (indicando que la bomba debe estar apagada) y el pin `MOTOR_PIN` se pone en estado bajo, desactivando la bomba de agua.
- **Detección de Nivel Vacío (o Bajo):** Si el pin `INT_PIN` se detecta en estado BAJO, esto significa que el nivel del agua ha descendido por debajo del umbral crítico (el tinaco está vacío o necesita ser llenado). En este caso, `motor_estado` se establece en `1` (bomba encendida) y el pin `MOTOR_PIN` se pone en estado alto, activando la bomba.

3.3. Bucle Principal y Eficiencia

Una vez que el sistema ha sido inicializado, la función `main()` entra en un bucle infinito (`while (1)`). Es importante destacar que este bucle permanece vacío. Esto no es un error,

sino una característica de diseño fundamental en sistemas embebidos controlados por interrupciones. El microcontrolador no necesita estar constantemente verificando el estado del sensor (polling); en su lugar, permanece en un estado de baja actividad, esperando pasivamente a que la interrupción del sensor de nivel lo "despierte" para ejecutar la lógica de control. Este enfoque minimiza el consumo de energía y asegura que el microcontrolador esté siempre disponible para responder instantáneamente a los cambios en el nivel del agua.

4. Aportación de Eduardo Gonzalez Gonzalez: Diseño Funcional

Como el único diseñador de este código, mi principal contribución radica en la concepción y la implementación de un sistema de control de nivel de agua que es altamente funcional, eficiente y robusto. Cada decisión de diseño fue tomada con el objetivo de crear una solución óptima para la automatización del llenado de tinacos. Los aspectos clave de mi diseño funcional incluyen:

- **Diseño Orientado a Interrupciones:** La elección de utilizar interrupciones externas para detectar los cambios en el sensor de nivel es una muestra de mi comprensión de la programación de microcontroladores de bajo nivel. Este enfoque garantiza una respuesta en tiempo real a los eventos, a diferencia de un sistema basado en sondeo que podría introducir latencia y consumir más recursos del CPU.
- **Gestión de Estados Simplificada y Efectiva:** La variable `motor_estado` es un ejemplo de cómo se puede gestionar el estado de un sistema de manera clara y concisa. Esta simplicidad contribuye a la legibilidad del código y facilita su mantenimiento.
- **Configuración Precisa del Hardware:** La configuración directa de los registros del microcontrolador para los pines y las interrupciones demuestra un control granular sobre el hardware. Esto es crucial para exprimir el máximo rendimiento y asegurar la fiabilidad en entornos embebidos, donde los recursos son limitados.
- **Robustez y Fiabilidad:** El código está diseñado para ser inherentemente robusto. Al reaccionar directamente a los cambios de estado del sensor, se minimiza la posibilidad de errores lógicos o de sincronización. La bomba se activa o desactiva de manera determinista en función del nivel del agua, lo que garantiza un funcionamiento fiable a largo plazo.

- **Eficiencia Energética:** Al mantener el microcontrolador en un bucle inactivo y solo activarse mediante interrupciones, el sistema consume una cantidad mínima de energía. Esto es especialmente importante para aplicaciones que pueden depender de fuentes de energía limitadas o que requieren una operación continua durante largos períodos.
- **Solución Práctica y Aplicable:** El diseño resuelve un problema común de automatización de manera elegante y efectiva. La aplicabilidad de este sistema se extiende a diversas situaciones, desde el control de tinacos domésticos hasta sistemas de llenado automático en pequeñas instalaciones, lo que demuestra mi capacidad para desarrollar soluciones prácticas con impacto real.

En síntesis, este proyecto es una clara evidencia de mi habilidad para diseñar y desarrollar código funcional para sistemas embebidos, con un enfoque en la eficiencia, la reactividad y la fiabilidad. Mi trabajo en este código refleja una sólida base en la electrónica digital y la programación de microcontroladores.

5. Conclusiones y Trabajo Futuro

5.1. Conclusiones

El sistema de control de nivel de agua basado en microcontroladores AVR ha sido diseñado y desarrollado con éxito, demostrando una solución eficiente y fiable para la automatización del llenado de tinacos. La implementación de una lógica de control basada en interrupciones es un pilar fundamental que garantiza una respuesta inmediata a los cambios en el nivel del agua, optimizando el consumo de energía del microcontrolador. Este proyecto valida mi capacidad para concebir, diseñar y programar sistemas embebidos que interactúan directamente con el hardware para resolver problemas de automatización específicos. La simplicidad y robustez del código son testimonio de un diseño funcional bien pensado, lo que lo hace ideal para aplicaciones prácticas donde la fiabilidad es primordial.

5.2. Trabajo Futuro

Para seguir mejorando y expandiendo las capacidades de este sistema, se proponen las siguientes líneas de trabajo futuro:

- **Implementación de Múltiples Sensores de Nivel:** Integrar varios sensores de nivel (por ejemplo, uno para nivel bajo, uno para nivel medio y uno para nivel

alto) para proporcionar un control más granular y permitir diferentes modos de operación (ej. llenado parcial, alerta de nivel bajo).

- **Interfaz de Usuario (HMI):** Desarrollar una interfaz de usuario simple, como una pantalla LCD o LEDs adicionales, para mostrar el estado actual del tinaco (vacío, llenando, lleno) y el estado de la bomba.
- **Conectividad Inalámbrica:** Añadir un módulo de comunicación inalámbrica (Wi Fi, Bluetooth o LoRa) para permitir el monitoreo remoto del nivel del agua y el control de la bomba a través de una aplicación móvil o una plataforma en la nube.
- **Detección de Fallos y Alarmas:** Implementar lógica para detectar posibles fallos (ej. bomba funcionando por demasiado tiempo sin que el nivel suba, sensor
defectuoso) y activar alarmas visuales o sonoras.
- **Optimización del Consumo de Energía:** Explorar modos de bajo consumo del microcontrolador (sleep modes) para aplicaciones alimentadas por batería, maximizando la autonomía del sistema.
- **Calibración y Ajuste:** Desarrollar un mecanismo de calibración para el sensor de nivel que permita ajustar los umbrales de activación/desactivación de la bomba de manera más precisa y adaptable a diferentes tamaños de tinacos.

Estas mejoras futuras no solo ampliarán la funcionalidad del sistema, sino que también consolidarán mi experiencia en el diseño de sistemas embebidos más complejos y conectados.