Laboratorio 05-CONTROL DE ENERGÍA ENTREGADA A MOTOR DC CON PWM

Martinez Benavides, Camilo
Escuela de Ciencias Exactas e Ingeniería
Universidad Sergio Arboleda - Bogotá, Colombia
camilo.martinez01@usa.edu.co

Rodriguez Rios, Oscar
Escuela de Ciencias Exactas e Ingeniería
Universidad Sergio Arboleda - Bogotá, Colombia
oscar.rodriguez02@usa.edu.co

Resumen

En este laboratorio se debe controlar la energía suministrada por el módulo PWM del STM32 para alimentar un motor DC, dependiendo de los valores de frecuencia que se configuren tanto en el "prescaler", "counter periodz "pulse", con esto se buscará una frecuencia óptima donde el motor trabaje adecuadamente, para que así no se sobrecaliente, y trabaje a una buena velocidad para que funcione óptimamente. Gracias a una interfaz gráfica desarrollada en el software QT Creator donde se podrá ingresar el valor deseado para el PWM que se transmitirá por USB a la STM32. la trasmisión de datos se realizará bajo un protocolo de bajo nivel, el cual será explicado a detalle en el informe.

Palabras cláve:

PWM, QT creator, Protocolo de comunicaciones, USB

1. Marco teórico

PWM: PWM significa Pulse Width Modulation (Modulación por ancho de pulso). Para transmitir una señal, ya sea analógica o digital, se debe modular para que sea transmitida sin perder potencia o sufrir distorsión por interferencias. Esta modulación es muy usada para controlar la cantidad de energía que se envía a una carga, es una técnica utilizada para regular la velocidad de giro de los motores, regulación de intensidad luminosa, controles de elementos termoeléctricos o controlar fuentes conmutadas entre otros usos. [1]
QT creator: Qt Creator es un entorno de desarrollo integrado multiplataforma que puede usar para crear aplicaciones o modificar aplicaciones existentes. [2]
Protocolo de comunicaciones: Los protocolos de red son estándares y políticas formales, conformados por restricciones, procedimientos y formatos que definen el intercambio de paquetes de información para lograr la comunicación entre dos servidores o más dispositivos a través de una red. [3]

2. Procedimiento

En primer lugar se aclara que la tarea principal de esta práctica de laboratorio es el control de un motor DC mediante una señal PWM que emite a una determinada frecuencia un pulso, al cual se le puede variar su energía entregada según el tiempo aplicado, que será predeterminado y enviada por una interfaz de usuario en el PC que permita cambiar este valor en cuestión de porcentaje y enviarlo al microcontrolador para que este realice el ajuste de manera continua al trabajo de emisión de la señal PWM ejecutándose, por lo que primero se hará la configuración general del programa que debe tener el STM32F401CCU6 para realizar las tareas que se requieren, las cuales son activación del TIM, configuración de su frecuencia, CP, activación del sistema PWM con ajuste del valor de comparación respecto al CP del TIM denominado (CCR1), configuración de comunicación USB Dual y procesamiento de información.

Así que al crear el proyecto de STM32CubeIDE se configura el High Speed Clock (HSE) con el Crystal/Ceramic Resonator, Luego se predetermina el Debug en Serial Wire y el Timebase Source con la opción Serial Wire, para con todo lo anterior colocar la frecuencia de funcionamiento del reloj interno (HCLK) en 84 MHz, con esto se finaliza la configuración general del sistema de la STM32, pasando ahora al desarrollo relacionado con la USB, aun en el archivo .ioc, en la sección de Connectivity-USB_OTG_FS, se configura el modo como dispositivo únicamente ya que no es necesaria otra configuración mas avanzada, y allí mismo se habilita la interrupción global "USB ON The Go FS", para luego en la sección de middleware, que actúa como una capa de traducción para comunicaciones entre dos dispositivos, allí se configura el USB_DEVICE como "Communication Device Class (Virtual Port Com)". Con este aspecto se finaliza el segundo de tres aspectos generales a configurar antes de generar el código de compilación del proyecto. Por ultimo en la sección de Timers, se selecciona el TIM2, donde primero para la fuente de reloj se selecciona el internal clock que ya fue configurado

anteriormente, después en el Channel 1 se selecciona la opción de PWM generation CH1, que asocia un pin físico a esta configuración que para este caso puntual fue el Pin A0, después de esto se tienen algunas características del TIM como el pre-escaler y el Counter Period, las cuales se modificaron en varias oportunidades para evidenciar la influencia en la aplicación, por lo que por ahora se dejan los valores predeterminados. Luego de esto se genera el código y en el main, se agregan llamados a funciones que están bibliotecas anexas para iniciar la configuración de GPIO, TIM2 Y USB_DEVICE, continuamente a esto se encuentra dos funciones para iniciar el PWM y permitir variar el valor del CCR1 para la generación del pulso, estas funciones respectivamente son:

- HAL_TIM_PWM_Start(&htim2,TIM_CHANNEL_1);
- __HAL_TIM_SET_COMPARE(&htim2,TIM_CHANNEL_1,0);

En donde la segunda función, tiene el tercer parámetro de entrada en 0, el cual representa el CCR1, que en este caso indica que no se genera ningún pulso. Luego de esto dentro del while(1) se coloca la siguiente fracción de código, en donde se tienen dos whiles independientes que aumentan y disminuyen respectivamente el valor del CCR1 según unos parámetros de evaluación de este mismo registro.

```
while(TIM2->CCR1<CP-1)
      {
          TIM2->CCR1++;
          HAL_Delay(50);
      }
      while(TIM2->CCR1>1)
      {
          TIM2->CCR1--;
          HAL_Delay(50);
      }
```

Antes de continuar con la explicación del eje central de este laboratorio que es la configuración PWM a partir del TIM, se explica a profundidad la sintaxis y procedimiento del sistema USB de comunicación integrado iniciando con el llamado de manera global de la funcion para transmisión de datos que es:

```
uint8_t CDC_Transmit_FS(uint8_t* Buf, uint16_t Len);
```

Luego de esto se realiza un proceso de intervención para la recepción de datos, primero creando una función de tipo weak en la librería de la USB para llamarla dentro de la función predeterminada de la librería que recibe la información, y poder acceder a la información desde el main, allí se procesan y asignan los datos recibidos según corresponda que para este caso, se basa en la asignación de un porcentaje de funcionamiento para el PWM, es decir desde que valor se quiere como base. Volviendo al PWM del TIM se utilizaron distintos preescalers, CP y delay's en el while, para obtener distintas frecuencias de aplicación y momentos de percepción para el motor, algunos ejemplos para estos son valores fueron un preescaler de 60000 para obtener la mayor frecuencia entera posible junto con un CP de 1400 para tener una resolución muy alta, pero con un delay pequeño para que no sea tan extensa la percepción, que luego se cambio a 100, pero no tenia mucho rango de aplicación, así que luego se cambio a 300 y por conveniencia se termino dejando en 200, otros valores utilizados fueron un preescaler de 21000, para una frecuencia de 4 kHz, acompañada de un delay de 100 ms y un CP variante entre 100, 500, 300 y finalmente puesto en 200, pero se observo que esta frecuencia era muy alta para el motor e implicaba un delay mas alto que no es una buena costumbre confiable.

3. Conclusiones

- Se puede concluir en este laboratorio que gracias a la STM32 la energía que se le entregue a un motor se controla fácilmente con el módulo PWM donde se comprendió y entendió su funcionamiento para en la práctica acertar en una frecuencia óptima para un motor DC de 12V para generar una velocidad deseada
- En conclusión al experimentar con varias valores de CP y preescaler se determino que la mejor opción de configuración es un preescaler de 42000 para que el TIM tenga una frecuencia resultante de 2 kHz y completado a esto un CP de 200, para obtener una resolución de 0,5
- En conclusión, se puede afirmar que gracias a la practica de prueba y error se aproximó de la mejor manera, al método mas óptimo de aplicación para este problema planteado, además que el PWM es una herramienta útil con un amplia gama de aplicación en el ámbito de la electrónica enfocado en el control de motores específicamente para su precisión, en arranque o funcionamiento a largo plazo.

4. Referencias

[1] ¿Qué son y para qué sirven los protocolos de comunicación de redes? (s. f.). Recuperado 29 de septiembre de 2022, de https://www.kionetworks.com/blog/data-center/protocolos-de-comunicaci %C3 %B3n-de redes#: %7E:text=Un %20protocolo %20es %20un %20conjunto,a %20trav %C3 %A9s %20de %20una %20red.

[2] Qt Company. (s. f.). Embedded Software Development Tools — Cross Platform IDE — Qt Creator. Recuperado 29 de septiembre de 2022, de https://www.qt.io/product/development-tools

[3] ¿Qué es PWM y cómo usarlo? (s. f.). Recuperado 29 de septiembre de 2022, de https://solectroshop.com/es/blog/que-es-pwm-y-como-usarlo–n38