Axel Giovanni Ojeda Hernandez 6CM1
Ibarra Perez Teodoro
"Microcontrolador y sistemas embebidos"
Sistemas en chip
Sistemas computacionales

Introdoccion

El MSP430 es una familia de microcontroladores producida por texas instrument, conocida por su bajo consumo de energía y eficiencia; Ideales para aplicaciones embebidas

Cuenta con registros de propósito general, de control y estado y específicos de periféricos. En los periféricos encontramos lo que son los temporizadores, módulos de comunicacion, ADC, etc. Los puertos de entrada salida son escenciales para la interacción con el entorno

Registros basicos de GPIO

PxIN: registro de entrada
PxOUT: registro de salida
PxDIR: registro de dirección
PxREN: registro de resistencia
PxSEL: selección de función para asignar pines a funciones

Temporizadores

Son herramientas versátiles que permiten ejecutar tareas a intervalos regulares

Registros clave de timer

TAxCTL: Control de temporizador
TAxCCTLn: Control de cada canal de captura/comparacion
TAxR: Registro de contador del temporizador
TAxCCRn: Registro de comparación/captura

Modulación por ancho de pulso PWM

El PWM es una técnica útil para controlar dispositivos como LEDs y motores, se puede ajustar la intensidad y la velocidad de manera eficiente

Ciclo de trabajo

es una señal de trabajo PWM se refiere al porcentaje del período de la señal en el que está en alto frente al tiempo total del ciclo

Convertidor analógico-digital ADC

convierte señales analógicas en valores digitales procesables. La resolución determina la cantidad de niveles discretos que puede representar

UART

Es un periférico de comunicación serial ampliamente utilizado en los microcontroladores. Permite la comunicación entre dispositivos seriales utilizando un protocolo asincrono La velocidad de transmición debe coincidir entre los dispositivos que se comunican, ademas, ambos dispositivos deben usar el mismo formato de datos (número de bits, paridad, bits de parada).

Materiales y Metodos

Ejercicio 1

Se configuró el puerto 1.0 como salida, después se le envió un 1 logico

Ejercicio 2

Se configuró el puerto 4.7 como salida, después se le envió un 0 logico, después el programa entra en un ciclo de decremento y comparacion, al llegar a 0 se envió un 1 logico al puerto 4.7 y se repite el proceso

Ejercicio 3

Se configuró el puerto 4.7 como salida, luego se conmutó el estado del puerto, después el programa entra en un ciclo de decremento y comparacion, al llegar a 0 se conmutó el estado del puerto 4.7 y se repitió el proceso

Ejercicio 4

Se configuró el puerto 4.7 como salida, luego se apagó. Se configuró el puerto 1.1 como entrada y se habilitó la resistencia en el puerto 1.1 como pull-up, después se agregó la condicion de que si se presionó el botón, se enciende el puerto 4.7 y en caso contrario se apagó

Ejercicio 6

Este codigo permitió que el led1 (4.7) y el led2 (1.0) se conmuten en secuencia cada vez que el botón conectado a 1.1 se presionó. Cuando se soltó el botón, ambos leds se apagaron

Ejercicio 5

Este codigo permitió que un led conectado a el pin 4.7 se encendiera y apagara cuando se presionó y soltó el botón conectado al pin 1.1 respectivamente. La resistencia de pull-up en 1.1 se aseguró que el estado del botón sea consistente cuando no se presionó.

Ejercicio 7
Este programa controló 2 leds de forma alternada utilizando la operación xor para cambiar sus estados. El bucle asegura que este proceso de conmutación y el retardo se repitan continuamente mientras el microcontrolador estuvo alimentado y en funcionamiento.

Ejercicio 8
El timer A0 generó interrupciones periódicamente según el valor de TA0CCR0. En cada interrupción, se verificó el valor de c, cuando c alcanzó 20, se conmutó el estado del del 1.0

Ejercicio 9
El timer A0 se configuró para utilizar la fuente de reloj ACLK, que es una fuente de reloj externa o un oscilador de baja frecuencia. Al estar en modo continuo, el temporizador no se detuvo y continuó generando interrupciones en intervalos regulares. Cada vez que se generó una interrupción se conmutó el estado del led 1.0

Ejercicio 10
El programa utilizó el timer A0 para controlar el parpadeo de los leds basado en la entrada del botón. Cuando se presionó el botón, el programa generó un tono alto y conmutó el estado de los leds. Cuando el botón no se presionó, se generó un tono bajo y se mantuvieron los leds en un estado particular.

Ejercicio 11
El programa controló el estado de 2 leds mediante un botón. Cada vez que el timer 0, generó una interrupción (aproximadamente cada 0.1 segundos) se verificó el estado actual y el estado del botón. Dependiendo de estas condiciones, se cambió el estado de la máquina de estados, y se ajustó la salida para controlar el encendido y apagado de los leds

Ejercicio 12
Se configuraron los pines para controlar los leds, buzzer y el botón. Luego se utilizaron 2 timers para generar interrupciones periódicas y controlar el estado de los leds con maquina de estados y generar diferentes tonos

Ejercicio 13
Este codigo implementó una máquina de estados que controla la generación de tonos en un buzzer según el estado del botón. Utiliza 2 timer para manejar la lógica de la máquina de estados y para generar los tonos. Cada estado de la maquina corresponde a una nota de la cancion "estrellita donde estas", generando asi una secuencia musical cuando se preciona el botón

Ejercicio 14
El programa entró en el modo de bajo consumo y habilitó las interrupciones globales. Cuando se presionó el botón se genero una interrupción, la rutina de interrupción cambió el estado del led y limpió la bandera de interrupción para futuras interrupciones

Ejercicio 15
El codigo configuró el timer A0 para generar un señal PWM, esta tiene un periodo determinado por TA0CCR0 y un ciclo de trabajo del 65% determinado por TA0CCR2

Ejercicio 16
Se configuró el ADC para leer el voltaje analógico presente en el pin 6.5. Cada segundo se inició una conversión y se espero a que se completó, cuando se completó en la rutina de interrupción dependiendo de el valor leido en ADC12MEM1 se encendió o apagó el led

Ejercicio 18
Se configuró el ADC para leer una señal analógica, se encendió y configuró en modo secuencial. Cuando se completó una conversión, se genero una interrupción que lee el valor convertido y lo utilizó para ajustar el ciclo de trabajo de 2 señales PWM. Se utilizaron 2 timer para generar 2 señales PWM

Ejercicio 19
Se configuraron 2 pines como salida para leds
Se configuró el timer para generar una interrupción cada segundo.
Se configuraron los pines 6.1 y 6.2 como entradas analógicas y se seleccionó la función especial de adc para esos pines
Con la interrupcion del time se inicia manualmente una nueva conversion del ADC, y con la interrupcion de la ADC se verifican los canales con los valores convertidos y dependiendo del valor leido se encienden o apagan los leds

Ejercicio 20
Se encendió la ADC y se conectó la fuente de relos del timer. Cuando se completó una conversión, la rutina de interrupción calcula la temperatura medida con el sensor de temperatura y luego la calibra para obtener la temperatura del ambiente

Ejercicio 21
Se encendió y configuró la ADC para tomar muestras constantemente y cada vez que se terminaba una conversión se calculaba la temperatura del ambiente, después se calibraba la temperatura y se guardaba en un arreglo, al conseguir 20 datos, se aplicaba un filtro promediador para obtener una temperatura del ambiente mas precisa

Ejercicio 22
Se configuró el ADC para medir la temperatura, tambien se estableció un temporizador para generar interrupciones periodicas

Se utilizó un buffer circular para almacenar las ultimas mediciones de temperatura y calcular un promedio, mejorando la precisión

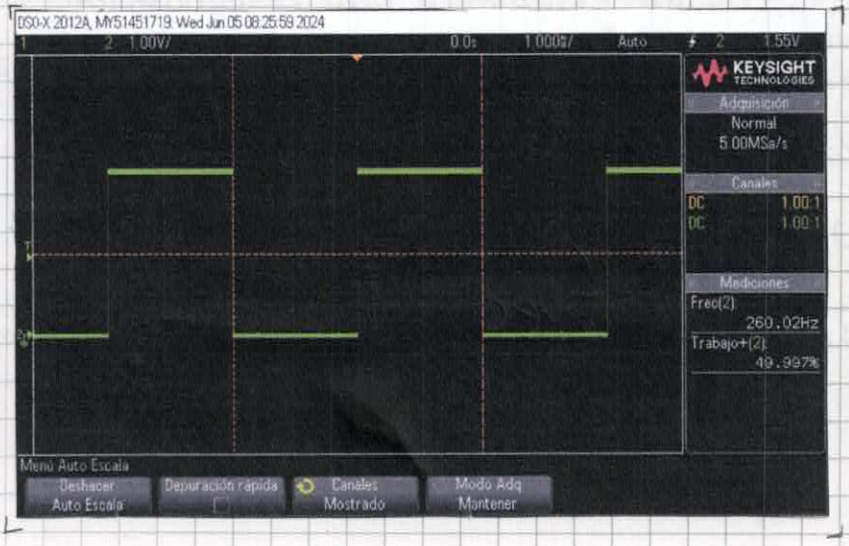
El ADC generó una interrupción cuando completó una conversión y el temporizador generó interrupciónes para iniciar nuevas conversiónes.

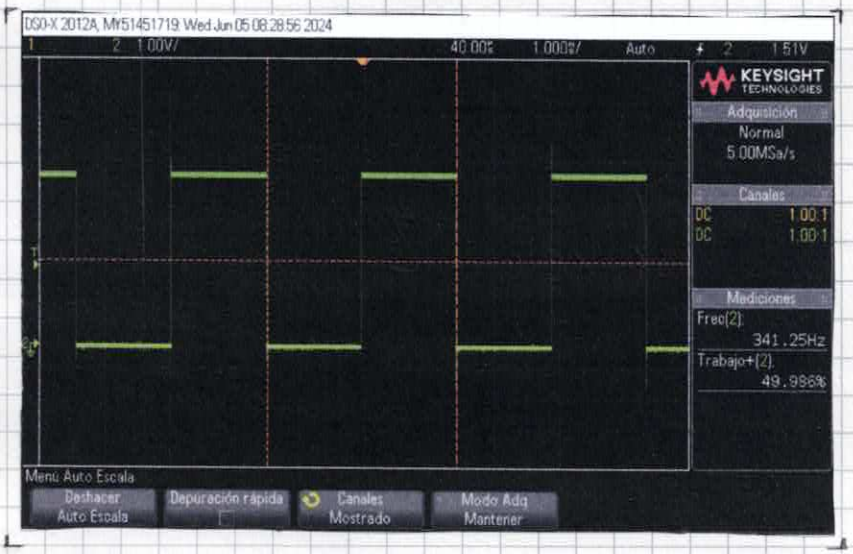
El microcontrolador entro en modo de bajo consumo entre las interrupciones

Ejercicio 23
Este codigo estableció una comunicacion serial entre el microcontrolador y la PC, configurada a 9600 baudios para recibir datos a traves de UART, responder con un eco de los datos recibidos para despues mostrar los en consola y si el dato era una "a" encender un led

Ejercicio 24
El codigo configuró el microcontrolador para comunicarse a traves de la UART, presenta un menu interactivo en la terminal y controla un led en la entrada recibida

Resultados

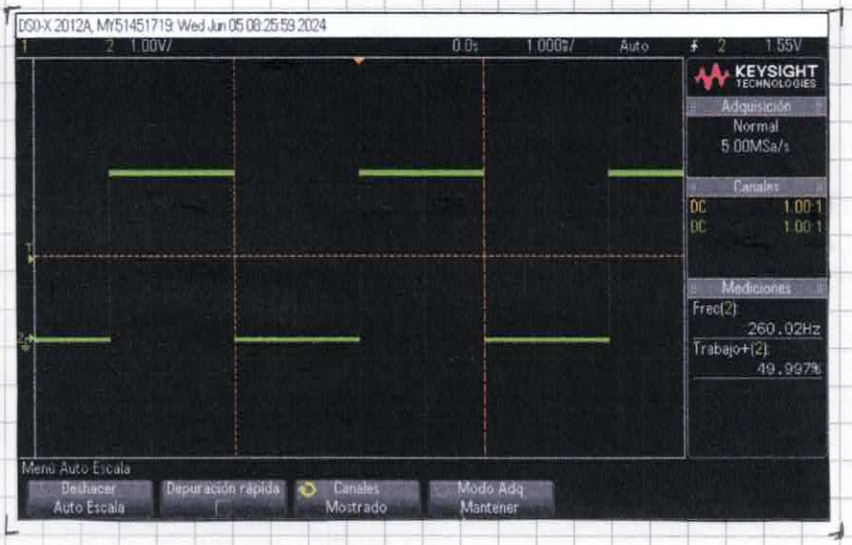
E1: Encender un led	Se encendió el led
E2: Encender y apagar un led	Se encendió el led y cuande el vegistvo llegó a 0 se apagó
E3: conmutar un led	El led apagado se encendió y cuando el vegistvo llegó a 0 se apagó
E4: Encender un led con un botón	Al pvesionar el botón se encendió el led y al soltarlo se apagó
E5: Encender y apagar un led con un botón	Al pvesionar el botón el led empezó a conmutar cada cierto tiempo y cuando se soltó el botón se apagó
E6: Alternar leds con un botón	Al pvesionar el botón los leds empezaron a pvender y apagar alternándose entve ellos y cuando se dejó de pvesionar el boton se apagaron los leds
E7: conmutar leds	Se encendieron y apagaron los leds de forma sincronizada
E8: conmutar led con timer up	se encendió y apagó un led cada cierto tiempo
E9: conmutar led con timer continuos	Se encendió y apagó un led
E10: generar tonos con timer	

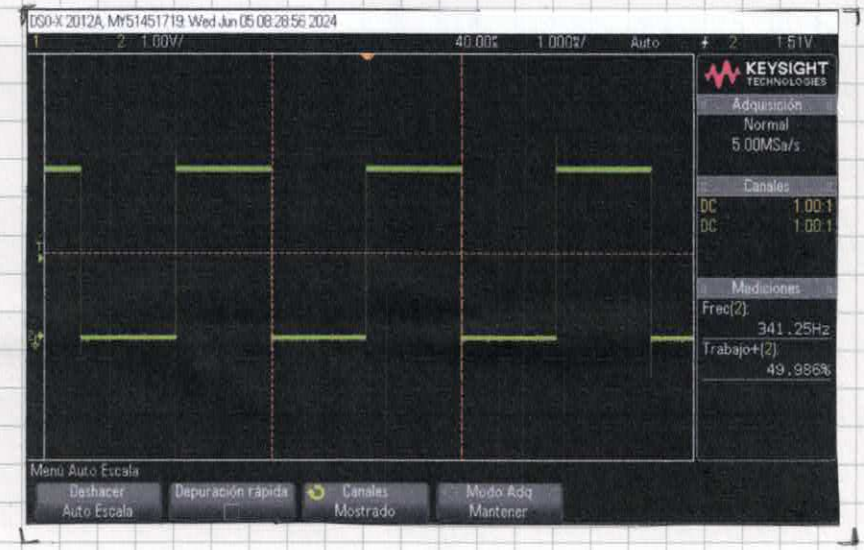


E11: máquina de estados moor

Se presionó el botón por primera vez y encendió un led, se presionó una segunda vez y encendió otro led y se apagó el primero, se presionó una tercera vez y encendieron ambos leds, se presionó una cuarta vez y ambos leds se apagaron

E12: generar tonos y conmutar leds con maquina de estados





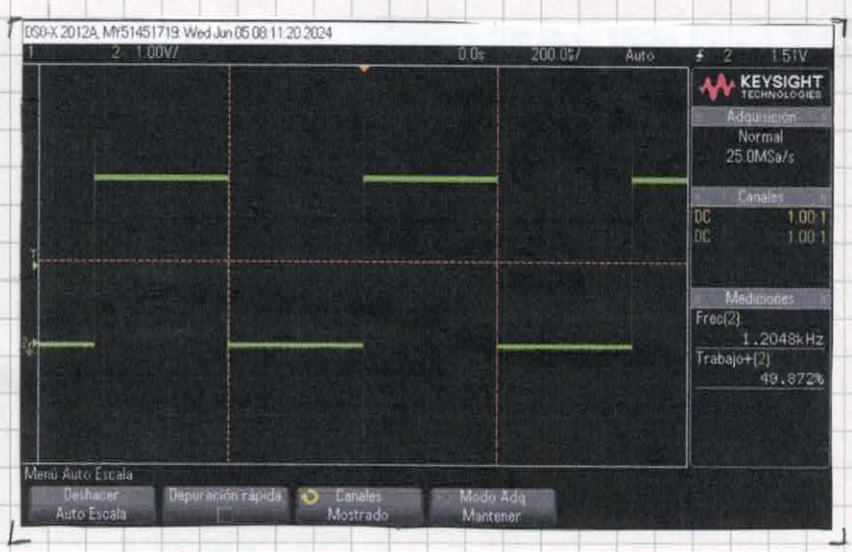
E13: cancion estrellita con máquina de estados

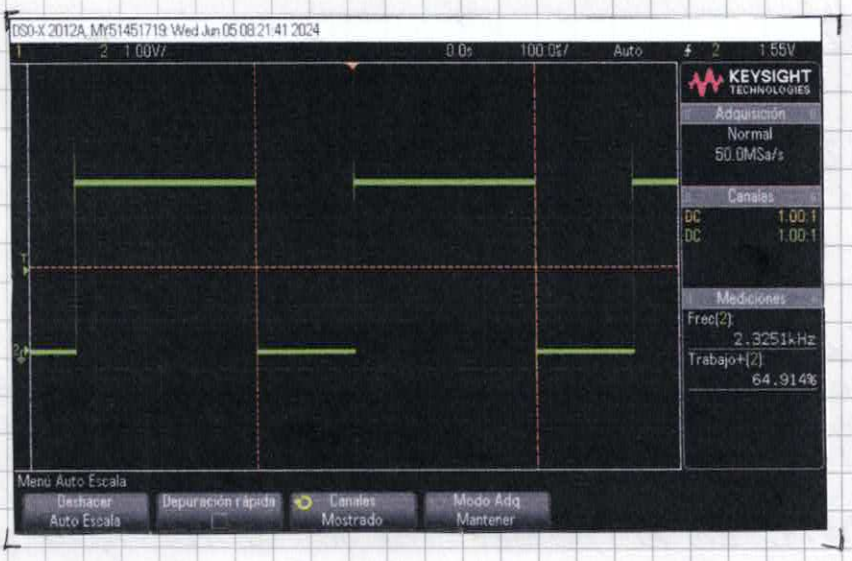
Mientras se presionava el botón cambiaba el estado y sonaba una nota de la canción, diferente cada que se presionaba

E14: genevar y reiniciar interrupciones

El led conmutó su estado cada vez que se presionó el botón

E15: genevar señal PWM





E16: Encender led con ADC	El led se encendió cuando el voltaje en el pin 6.5 fue mayor que aproximadamente 2.5v y se apagó cuando era menor o igual
E18: control de intensidad de motores con pwm	Se generó una señal pwm que permitió manejar 2 motores de manera que individualmente podíamos elegir si giraban en sentido horario o antihorario y con el potenciómetro ajustábamos la velocidad de giro
E19: ADC con varios canales	Se encendió el led en p1.0 cuando el valor de ADC12MEM1 fue mayor a 2040 de lo contrario se apagó. De forma inversa funcionó para ADC12MEM2
E20: sensor de temperatura	Temp daba valores entre 50 y 55 y temp calibrada daba valores entre 30 y 35
E21: sensor de temperatura y filtro	Temp daba valores entre 50 y 55, temp calibrada daba valores entre 30 y 35 y temp promediada daba valores entre 30 y 32
E22: sensor de temperatura y filtro optimizado	Hace lo mismo que E21 pero en este caso entraba en modo de bajo consumo entre interrupciones

E23: comunicación serial	Se escribió hola mundo en la terminal
E24: menú interactivo con comunicacion serial	Se encendió el led al presionar la letra a en el menú de encender/apagar led

Conclusiones

Los códigos realizados exploran diversas capacidades del microcontrolador MSP430, tales como la configuración de puertos, el manejo de interrupciones, la comunicación serial UART, el uso de ADC para medir temperaturas, y la implementación de PWM. En el primer código, se configura un led para que se encienda o apague en función del estado de un botón, demostrando un control básico de entrada/salida digital. Otro código utiliza el ADC para leer un valor analógico de un sensor y controla el ciclo de trabajo de 2 señales PWM en función de la lectura del ADC, lo cual ilustra cómo las lecturas analógicas pueden ser utilizadas para generar señales PWM moduladas.

En otro ejemplo se configura la comunicación UART para recibir datos de una terminal serial y controlar un led según la entrada recibida, lo que muestra la capacidad para manejar comunicación serial y control de periféricos en respuesta a comandos. Esta funcionalidad se amplía con un menú interactivo a través de UART, donde se muestra un menú en la terminal y se espera a que el usuario seleccione una opción, permitiendo el control de leds, generacion de tonos, pwm y lectura de un sensor de temperatura

Finalmente, se utiliza el ADC para leer valores de un sensor de temperatura interno y se aplica un filtro promediador para calcular una temperatura calibrada. Este código, implementa un buffer circular para almacenar las últimas lecturas del sensor y suavizar los datos, mejorando la precisión de las mediciones. En conjunto, estos ejemplos destacan la flexibilidad del MSP430 para manejar múltiples tareas, integrar diferentes periféricos y módulos, y aplicar técnica avanzadas de procesamiento de señales en aplicaciones prácticas.

Discusión

El uso del microcontrolador para aplicaciones diversas, como el control de leds, la generación de PWM, la lectura de sensores de temperatura y la comunicación serial, se relacionan con multiples trabajos científicos y aplicaciones prácticas en el campo de la ingeniería electrónica y la computación.

Como por ejemplo, el uso de PWM y control de leds se relaciona con aplicaciones de control digital en sistemas embebidos. Un trabajo notable en este campo es "Digital control of switching power supples" de Batarseh, que discute el uso de PWM en fuentes de alimentación conmutadas, una técnica esencial en la electrónica de potencia y control digital.

Tambien tenemos el código que utiliza el ADC para leer valores analógicos y aplicar un filtro promediador, se alinea con métodos de adquisición de datos y procesamiento de señales en sistemas embebidos. Un estudio relevante es "Microcontroller-based data acquisition systems: A practical design and implementation, que explora el uso de microcontroladores para la adquisición y procesamiento de datos en tiempo real

Otro ejemplo es el uso de UART para comunicación serial y control remoto de periféricos se relaciona con la implementación de interfaces de comunicación en sistemas embebidos. Un trabajo que abonda esta temática es "Serial communication interfaces for embedded systems" que describe las interfaces de comunicación y su importancia en el control y monitoreo de sistemas embebidos.

Por último tenemos el uso del microcontrolador para la lectura de un sensor de temperatura interno y la calibración de datos se relaciona con técnicas de medición y calibración en sensores de temperatura. Un estudio significativo en esta area es "Precision temperature sensors in CMOS technology" que discute la precisión y calibración de sensores de temperatura integrados en tecnologías CMOS

Estos trabajos científicos proporcionan una base teórica y práctica que respalda las aplicaciones y técnicas implementadas en los códigos, demostrando la relevancia y aplicabilidad de estos conceptos en el desarrollo de sistemas embebidos avanzados.

Referencias

I. Batarseh, "Digital control of switching power supplies", Wiley-Interscience, 2004

M. Kaur and B. Singh, "Microcontroller-based data acquisition system: A practical design and implementation", International Journal of Advanced Research in Computer and Communication Engineering, vol. 4, no. 1, pp. 123-127, 2015

A. S. Tanenbaum, "Serial communication interface for embedded systems", Prentice hall, 2009

A. Bakker and J. Huijsing, "Precision temperature sensors in CMOS technology", Kluwer Academic Publishers, 2000