

Practica 2 - Temporizador

Henry José, Chacón Barillas, 202002535
Escuela de Mecánica Eléctrica, Facultad de Ingeniería, Universidad de San Carlos de Guatemala

La práctica consiste en realizar un programa el cual pueda controlar el tiempo de encendido y apagado de un led, utilizando una temporización.

I. OBJETIVOS

- Aplicar la inicialización de los GPIO's del microcontrolador y crear un programa que sea capaz de realizar retardos.
- Crear un programa que sea capaz de poner en alto o en bajo un pin del microcontrolador.
- Comprender los retardos a modo de contadores.
- Calcular los valores necesarios de acuerdo al reloj del microcontrolador.

II. MARCO TEÓRICO

A. Keil μVision

Es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) ampliamente utilizado para el desarrollo de software embebido, especialmente en microcontroladores basados en la arquitectura ARM. Fue desarrollado por Keil, una empresa que ahora es parte de ARM Holdings, y es uno de los entornos de desarrollo más populares para programar y depurar microcontroladores.

Características:

- Soporte de Microcontroladores: Keil uVision es compatible con una amplia gama de microcontroladores basados en ARM, como los de la serie Cortex-M, y también soporta otros microcontroladores como los basados en 8051.
- Compilador ARM (ARMCC): Incluye el compilador de ARM, que optimiza el código para la arquitectura ARM, permitiendo un rendimiento eficiente y un uso reducido de memoria.
- Depuración y Simulación: Ofrece herramientas avanzadas de depuración y simulación que permiten al usuario probar y depurar el código en tiempo real o a través de simulaciones de hardware.

- Editor de Código: Tiene un editor de código integrado con soporte para sintaxis destacada, autocompletado, y herramientas de búsqueda que facilitan la escritura y edición de código.

- Gestión de Proyectos: Permite la gestión de proyectos con múltiples archivos, automatización de tareas de compilación y enlace, y configuración de opciones de compilación.

- Configuración de Periféricos: Para microcontroladores compatibles, uVision incluye herramientas gráficas que permiten configurar fácilmente los periféricos, pines, y otros recursos del microcontrolador.

- Soporte para Lenguaje Ensamblador y C/C++: uVision permite escribir código en ensamblador, C y C++, lo que brinda flexibilidad en la programación y optimización del código para aplicaciones embebidas.

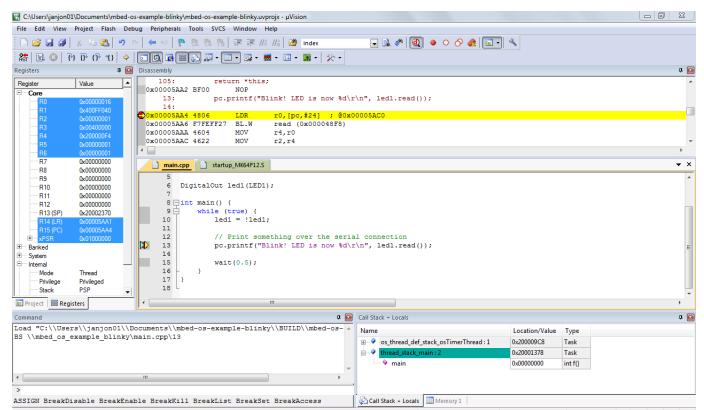


Figura 1: Entorno de Keil μVision

III. DIAGRAMA DE FLUJO

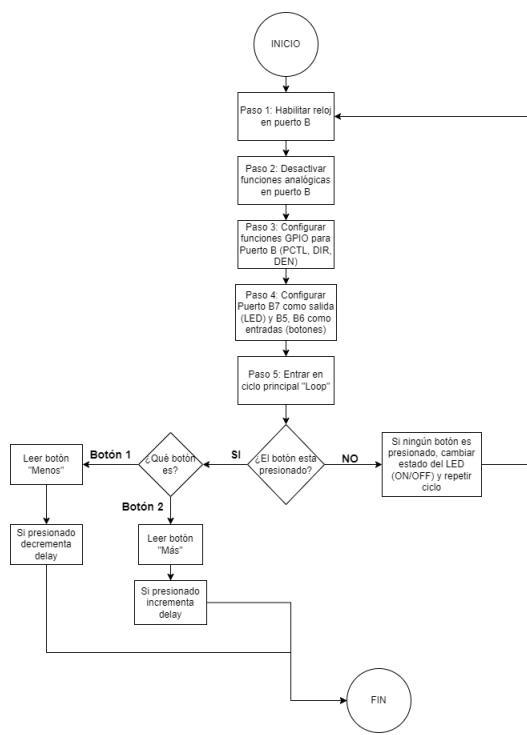


Figura 2: Algoritmo del programa realizado

IV. DISEÑO EXPERIMENTAL

La práctica consiste en realizar un programa el cual realice la función de un temporizador. La duración del retardo (Temporizador) tendrá un rango de duración de 1 a 5 segundos mostrado por medio de un led encendido. El tiempo de encendido será controlado utilizando 2 pulsadores de los cuales uno será para aumentar la duración del temporizador y otro para disminuirla, esto se hará en saltos de 1 segundo. Se usará un tercer pulsador llamado start el cual tendrá la función de iniciar el temporizador después de tener definido el tiempo de encendido.



Figura 3: Circuito propuesto por el ingeniero

A. Materiales

- Laptop
- Keil μ Vision

B. Procedimiento

1. Configuración del reloj

- Habilitar el reloj del microcontrolador para el Puerto B. Esto se realiza modificando el registro $SYSTCLR CGCGPIO_R$, activando el bit correspondiente para el puerto B (bit 2).

2. Deshabilitar funciones analógicas

- Desactivar las funciones analógicas en el puerto C. Esto se logra mediante la modificación del registro $GPIO_PORTB_AMSEL_R$, que controla las funciones analógicas del puerto.

3. Configuración de GPIO

- El registro $GPIO_PORTC_PCTL_R$ se ajusta para habilitar las funciones GPIO, eliminando cualquier configuración previa de funciones alternativas.

4. Configurar entradas y salidas

- Configurar los pines de entrada y salida en el puerto B. Se utiliza el registro $GPIO_PORTC_DIR_R$ para definir el pin B7 como salida (para controlar un LED) y los pines B5 y B6 como entradas (para los botones).

5. Habilitar puerto digital

- El puerto B se configura como puerto digital habilitando el registro $GPIO_PORTC_DEN_R$.

6. Ciclo principal del programa

Se entra en un ciclo infinito (Loop) donde el programa realiza las siguientes operaciones:

- Lee el estado de los botones (en B5 y B6).
- Si el botón "Menos" (B5) está presionado, se disminuye el valor del delay, haciendo que el parpadeo del LED sea más rápido.
- Si el botón "Más" (B6) está presionado, se incrementa el valor del delay, haciendo que el LED parpadee más lentamente.
- Si ninguno de los botones está presionado, el LED cambia de estado (ON/OFF).

V. CÓDIGOS

A. Código para el calculo del seno

```

1      SYSCTL_RCGCGPIO_R    EQU 0x400FE608
2      GPIO_PORTB5          EQU 0x40005080
3      GPIO_PORTB6          EQU 0x40005100
4      GPIO_PORTB7          EQU 0x40005200
5      GPIO_PORTB_AMSEL_R   EQU 0x40005528
6      GPIO_PORTB_PCTL_R   EQU 0x4000552C
7      GPIO_PORTB_DIR_R    EQU 0x40005400
8      GPIO_PORTB_AFSEL_R   EQU 0x40005420
9      GPIO_PORTB_DEN_R    EQU 0x4000551C
10     CONSTANTE           EQU 6000000
11     DISMINUNCION        EQU 5000000
12
13
14     AREA     codigo, CODE, READONLY, ALIGN=2
15     THUMB
16     EXPORT Start
17
18 Start
19     ; Paso 1. Reloj en Puerto B
20     LDR R1, =SYSCTL_RCGCGPIO_R
21     LDR R0, [R1]
22     ORR R0, R0, #0x02 ; Habilitar puerto B (0
23     x02)
24     STR R0, [R1]
25     NOP
26     NOP
27     ; Paso 3. Deshabilitar Funciones Anal gicas
28     ; en el Puerto B
29     LDR R1, =GPIO_PORTB_AMSEL_R
30     LDR R0, [R1]
31     BIC R0, #0xE0
32     STR R0, [R1]
33     ; Paso 4. Configurar como GPIO Puerto B
34     LDR R1, =GPIO_PORTB_PCTL_R
35     LDR R0, [R1]
36     BIC R0, R0, #0xFF000000
37     BIC R0, R0, #0x0FF0000
38     STR R0, [R1]
39     ; Paso 5. Dirección del Puerto B7 (Salida)
40     LDR R1, =GPIO_PORTB_DIR_R
41     LDR R0, [R1]
42     ORR R0, R0, #0x80
43     STR R0, [R1]
44     ; Dirección Puerto B5-B6 (Entrada)
45     LDR R1, =GPIO_PORTB_DIR_R
46     LDR R0, [R1]
47     BIC R0, R0, #0x60
48     STR R0, [R1]
49     ; Paso 6. Limpiar bits función alternativa
50     ; puerto B
51     LDR R1, =GPIO_PORTB_AFSEL_R
52     LDR R0, [R1]
53     BIC R0, R0, #0xE0
54     STR R0, [R1]
55     ; Paso 7. Habilitar como puerto digital
56     ; puerto B
57     LDR R1, =GPIO_PORTB_DEN_R
58     LDR R0, [R1]
59     ORR R0, R0, #0xE0
60     STR R0, [R1]
61
62     Delay
63     ADD R2, #1
64     NOP
65     NOP
66
67     NOP
68     CMP R2, R3
69     BNE Delay
70     BX LR
71
72 led; luz por defecto
73     LDR R5, =GPIO_PORTB7
74     LDR R4, [R5]
75     EOR R4, R4, #0x80
76     STR R4, [R5]
77     LDR R2, =0
78     LDR R3, =CONSTANTE
79     B Loop
80
81 Menos; ira disminuyendo el tiempo
82     LDR R5, =GPIO_PORTB7
83     LDR R4, [R5]
84     EOR R4, R4, #0x80
85     STR R4, [R5]
86     LDR R2, =0
87     SUB R3, R10
88     B Loop
89
90 Mas; incrementar el tiempo
91     LDR R5, =GPIO_PORTB7
92     LDR R4, [R5]
93     EOR R4, R4, #0x80
94     STR R4, [R5]
95     LDR R2, =0
96     ADD R3, R10
97     B Loop
98
99 Push1; Botón de Menos
100    LDR R1, =GPIO_PORTB5
101    LDR R0, [R1];
102    BX LR
103
104 Push2; Botón de Mas
105    LDR R1, =GPIO_PORTB6
106    LDR R0, [R1];
107    BX LR
108
109 Loop
110    BL Push1
111    LDR R2, =0
112    BL Delay
113    CMP R0, #0x20
114    BEQ Menos
115    BL Push2
116    LDR R2, =0
117    BL Delay
118    CMP R0, #0x40
119    BEQ Mas
120    CMP R0, #0x00
121    BEQ led
122    CMP R0, #0x00
123    B Loop
124
125 ALIGN
126 END

```

Listing 1: Entorno de keil uvision

VI. RESULTADOS

A. Circuito en funcionamiento

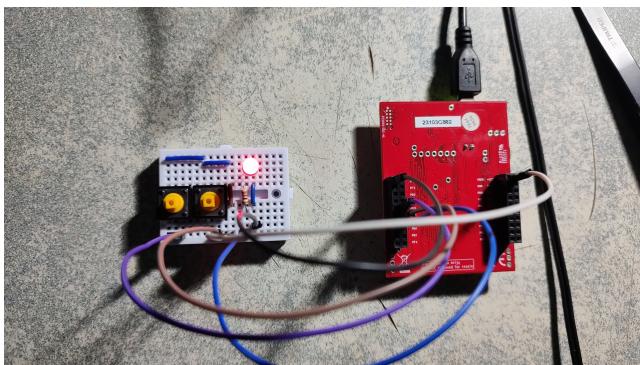


Figura 4: Circuito armado en protoboard

VII. DISCUSIÓN DE RESULTADOS

El objetivo de este programa es controlar el parpadeo de un LED conectado al pin B7 del Puerto B de la Tiva C TM4C123GH6PM. El parpadeo del LED se puede ajustar utilizando dos botones conectados a los pines B5 y B6:

A. Configuración exitosa de los pines

El código habilita correctamente el reloj para el Puerto B, configura los pines como entradas y salidas, y desactiva las funciones analógicas y alternativas, asegurando que los pines se usen como GPIO digitales.

B. Control de parpadeo

- Cuando el usuario presiona el botón conectado a B5 ("Menos"), el delay se reduce, acelerando el parpadeo del LED.
- Cuando el usuario presiona el botón en B6 ("Más"), el delay aumenta, ralentizando el parpadeo del LED.
- Si ningún botón es presionado, el LED sigue alternando su estado, cambiando entre ON y OFF cada vez que el ciclo se repite.

C. Lectura de los botones

El código verifica el estado de los botones antes de ajustar el tiempo de parpadeo. Aunque no se utilizan interrupciones (lo que podría mejorar la eficiencia del código), el enfoque por polling (consulta continua del estado) cumple su función en este contexto sencillo.

D. Limitaciones

- No se implementa ningún mecanismo para asegurar que el valor del delay no sea demasiado pequeño o demasiado grande, lo que podría hacer que el LED parpadee tan rápido que no sea visible o tan lento que parezca apagado.
- La lógica actual no debouncing (filtro de rebotes) para los botones, lo que podría provocar que las pulsaciones rápidas de los botones se detecten varias veces.

VIII. CONCLUSIONES

- Funcionamiento correcto: El código cumple con su propósito, logrando la correcta configuración de los pines del microcontrolador para controlar el parpadeo de un LED mediante dos botones.
- Mejoras posibles: Se podría implementar un sistema de debouncing para evitar lecturas erróneas al presionar los botones, además de establecer límites máximos y mínimos en el delay para evitar que el LED parpadee demasiado rápido o demasiado lento.
- Posibilidades de expansión: El programa podría mejorarse usando interrupciones en lugar de consulta continua (polling), lo que reduciría el uso de recursos del microcontrolador. También se podrían agregar más funciones, como distintos patrones de parpadeo o un indicador visual del tiempo actual de delay.
- Aplicabilidad: Este tipo de programa es ideal para proyectos simples donde se requiere control básico de hardware (GPIO), como en prototipos, entrenamientos o pruebas de control de dispositivos electrónicos.

IX. REPOSITORIO

https://github.com/Henry-Chacon/Practica_2_LAB_E5. [Fuente: Elaboración propia 2024].