6-4-2017

Practica 7

Uso de puertos y retardos mediante software

CARLOS OMAR CALDERON MEZA

MICROPROCESADORES Y MICROCONTROLADORES

Objetivo:

Mediante esta práctica el alumno aprenderá la forma básica de implementar retardos por software.

Equipo:

- Computadora Personal con AVR Studio

Teoría:

- Retardos por software (cálculos)

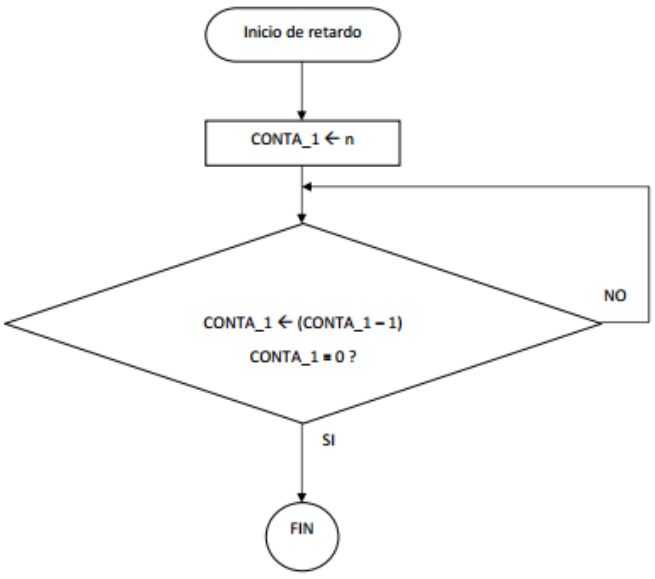
A menudo es necesario que nuestros programas usen demoras o retardos, por ejemplo, si deseamos hacer parpadear un led cada segundo evidentemente necesitaremos usar un retardo de 1s. Los retardos son prácticamente omnipresentes en nuestros programas. Y hay dos formas de hacerlos [1]:

-Por software
-Por Hardware

Retardo por Software

Los retardos por Software consisten en que el microcontrolador se quede “enciclado” durante un tiempo. Es decir, es necesario usar uno o varios contadores que deberán ser decrementados, cuando dichos contadores lleguen a 0 habrán concluido el retardo.

Diagrama de flujo:



Observe en el diagrama anterior como a una variable CONTA_1 se le asigna un numero n, posteriormente esta variable se decrementa y se pregunta si ha llegado a 0, si no ha llegado a 0 entonces se vuelve a decrementar, y así sucesivamente hasta que llegue a 0 en cuyo caso es el FIN del retardo. El programa se quedó “perdiendo el tiempo” encilado dando vueltas n veces.

Supongamos que tenemos el siguiente código [2]:

```
ldi R24,0X05        ;R24=5
nxt: nop            ;no operación
     dec R24        ; R24=R24-1
     brne nxt       ; salta a nxt si no es cero
                    ; fue cero continua aquí
```

Si queremos saber cuánto tiempo consume la ejecución de esta sección de código, es necesario hacer un análisis del código y ver cuantos ciclos de reloj son necesarios para cada instrucción y contabilizar las veces que se ejecuta cada instrucción.

Análisis:

[x]: veces se ejecuta la instrucción

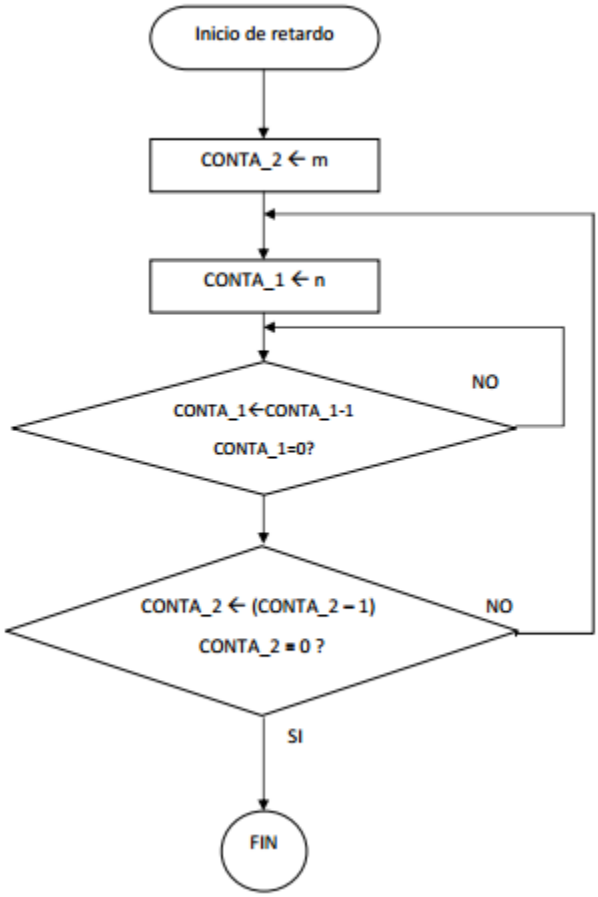
(y): número de ciclos de la instrucción

```
ldi R24,0X05        ;[1] * (1)
nxt: nop            ; [5] * (1)
     dec R24        ; [5] * (1)
     brne nxt       ; [4] * (2) y [1]*(1)
```

Ahora, para conocer el tiempo total es necesario calcular el total de ciclos de reloj del código, por tanto tenemos:
En total tenemos 1+5+5+(4*2)+1 = 20 ciclos y ahora podemos determinar el tiempo total si conocemos la frecuencia con que opera el procesador. Por ejemplo si se opera a 8MHz tenemos que un ciclo de reloj tiene un período de 1/8Mhz = 125nS, por tanto el tiempo total de la secuencia es 125uS x 20 = 2.5 uS.

Bucles anidados

Como hemos visto el retardo máximo que se puede generar de las formas enunciadas anteriormente son muy limitadas. Para generar retardos mucho mayores necesitamos usar BUCLES ANIDADOS. Estos bucles anidados concisten generar un Retardo base que se repetirá n veces, el retardo base se hace de la manera anteriormente mencionada usando un bucle que llamamos bucle interno, y al repetir este retardo base n veces estamos formando un bucle mayor o bucle externo. Veamos el ejemplo en flujo-grama:



Observe como primero se carga a la variable CONTA_2 con m, luego CONTA_1 se carga con n, luego se decrementa CONTA_1 hasta que llegue a 0 en cuyo caso decrementa CONTA_2, si CONTA_2 no es 0 entonces vuelve a cargar CONTA_1 con n y se vuelve a repetir el ciclo de decrementar CONTA_1 hasta 0, el ciclo se repite m veces hasta que CONTA_2 llegue a 0 en cuyo caso será el fin del retardo.

Desarrollo:

Implementar retardos por software:

A) 234 us
B) 5 ms
C) 666 ms

Conclusiones y comentarios

Generar retardos en un sistema basado en un microcontrolador no es tan sencillo como lo habría imaginado, para lograr un retardo exacto se tienen que plantear ecuaciones y debido a la arquitectura del atm mega2560 se encuentra limitadas las variables o los rangos para encliclar los bucles a 256 iteraciones

Dependiendo de la aplicación en la que se esté trabajando los retardos pueden ser necesarios y en ocasiones deben ser muy exactos, por ejemplo, recordando las bases de un sistema embebido categoría tiempo real tipo duro, podría controlar este sistema una situación donde los tiempos de retardo son fundamentales para el funcionamiento, y si se tarda un poco más o menos ocurrir una catástrofe o perdidas irreparables.

Bibliografía

[1] Joel Oswaldo Campos Pérez. (2009). CURSO BÁSICO DE PIC RETARDOS POR SOFTWARE. 07/04/2017, de Inventronica Grupo estudiantil Sitio web: https://campusvirtual.univalle.edu.co/moodle/pluginfile.php/161484/mod_resource/content/0/clase_3/Generar-retardos-para-PIC-en-MPLAB.pdf

[2] Leocundo Aguilar. (2010). Microcontroladores Retardos mediante software. 07/04/17, de UABC.