Interrupciones

Las interrupciones en los microcontroladores pueden ser de dos tipos: de hardware y de software.

Consisten en el cambio en la secuencia de ejecución de un programa cuando se da un determinado evento de hardware.

El microcontrolador ejecuta una rutina asociada con el evento de hardware implicado, conocida como "rutina de servicio".

Una vez que concluye la ejecución de la rutina de servicio, el microcontrolador debe retornar a la secuencia de ejecución que fue interrumpida.

Ejemplos

Para un determinado MCU, eventos de hardware podrán ser entre otros: que el puerto serie asíncrono haya recibido un byte y éste esté listo para ser leído, que se de un flanco de bajada en un determinado bit de puerto, etc.

En esta exposición, a los diversos eventos de hardware que pueden hacer que se de una interrupción, se les denomina INSTANCIAS DE INTERRUPCIÓN.

Requerimiento

Cuando una instancia de interrupción se presenta, se dice que se tiene un requerimiento de interrupción (RI).

En resumen, las interrupciones de hardware en los microcontroladores permiten manejar eventos de hardware de manera eficiente, ejecutando rutinas de servicio asociadas a cada evento. Al presentarse una instancia de interrupción, se tiene un requerimiento de interrupción que el microcontrolador debe atender. Es importante conocer los eventos de hardware que pueden generar interrupciones en un MCU para programar adecuadamente su manejo.

El MC9S08SH32 cuenta con 33 vectores de interrupción, numerados del 0 al 32.

Cada vector de interrupción corresponde a un evento específico y tiene asociada una rutina de servicio.

Prioridad

Las interrupciones en el MC9S08SH32 tienen prioridad predecible.

Las interrupciones de menor número tienen mayor prioridad que las de mayor número.

Las interrupciones de igual número tienen prioridad según el orden en que ocurren.

Habilitación y deshabilitación

Las interrupciones se habilitan y deshabilitan mediante el registro CCR (Condition Code Register).

Para habilitar las interrupciones, se debe establecer el bit I del registro CCR en 1.

Para deshabilitar las interrupciones, se debe establecer el bit I del registro CCR en 0.

Ejemplo

Supongamos que se desea manejar la interrupción correspondiente al vector 12, asociado al evento de recepción de datos por el puerto serie asíncrono.

Para habilitar esta interrupción, se debe establecer el bit PEIE del registro SCIxC2 en 1 y el bit RIE del registro SCIxCR2 en 1.

Se debe también establecer la dirección de la rutina de servicio correspondiente en la tabla de vectores de interrupción.

Cuando se reciba un byte por el puerto serie asíncrono, se generará una instancia de interrupción y se ejecutará la rutina de servicio asociada al vector 12.

En resumen, el MC9S08SH32 cuenta con 33 vectores de interrupción que permiten manejar eventos específicos de manera eficiente. Las interrupciones tienen prioridad predecible y se habilitan y deshabilitan mediante el registro CCR. Es importante conocer cómo se manejan las interrupciones en este MCU para programar adecuadamente su uso en aplicaciones específicas.

PASO 1: Guarda en el stack cinco bytes en el siguiente orden:

- PCL (byte bajo del PC de retorno)
- PCH (byte alto del PC de retorno)
- Registro X
- Registro A
- Registro CCR

PASO 2: El bit “I” se pone en uno lógico

PASO 3: Lee el valor del vector de interrupción asociado, esto desde el par de direcciones, propias de la instancia de interrupción implicada.

PASO 4: Salta a la dirección denotada por el vector de interrupción capturado en el paso anterior, esto es, inicia la ejecución de la rutina de servicio asociada con la instancia de interrupción para la cual se está respondiendo.

Tiempo de latencia:

11Tb<=Tlat<=22Tb. Donde Tb es el periodo del reloj de bus. Por ejemplo, si éste

es de 20 MHz, Tlat podrá estar entre 0.55 µs y 1.1 µs

FORMATO DE UN PROGRAMA QUE USE INTERRUPCIONES

En general, un programa que maneje interrupciones, estará integrado por cuatro bloques funcionales, denominados aquí como BLOQUE1, BLOQUE2, PLOQUE3 y BLOQUE4.

ESTRUCTURA DE UN PROGRAMA QUE USE INTERRUPCIONES

BLOQUE 1
BLOQUE 2
BLOQUE 3
BLOQUE 4

A continuación se menciona que deben contener cada uno de los bloques.

En general, el BLOQUE 1 contendrá:

- Código que inicialice los periféricos del MCU que se usen, para fines de la funcionalidad de éstos en la aplicación
- Código que inicialice en uno lógico a todos los habilitadores locales, propios de cada una de las instancias de interrupción, que se usen en la aplicación
- Al final de este bloque deberá estar la instrucción **CLI**, lo cual hará que el MCU pueda responder a los requerimientos de interrupción implicados en la aplicación

En general, el BLOQUE 2 contendrá:

- Código que por lo regular es un lazo donde el MCU efectúa acciones propias de la funcionalidad básica de la aplicación.

En general, el BLOQUE 3 contendrá:

- Código de cada una de las rutinas de servicio asociadas con cada una de las instancias de interrupción implicadas en la aplicación.

  Es importante destacar que las rutinas de servicio de interrupción, deben concluirse con la instrucción RTI, y no con la instrucción RTS.

- Código de cada una de las rutinas llamables con la instrucción JSR que se usen en la aplicación, éstas desde luego que se concluyen con la instrucción RTS.

FORMATO DE LAS DECLARACIONES PARA COLOCAR VECTORES DE INTERRUPCIÓN

Considerando las direcciones recabadas, la declaración de la colocación del vector asociado con la instancia 11 sería:

```
org $d7e8
dw servint11
```

Para la instancia 20 la declaración de colocación del vector sería:

```
org $d7d6
dw servint20
```

Manejo LCD conectado al MCU

Esta tarjeta cuenta con un módulo LCD alfanumérico de 2 líneas por 16 caracteres, conectado al puerto C del MCU.

Configuración

Para configurar el LCD, se debe inicializar el puerto C del MCU como salida.

Se debe también configurar el LCD en modo de 8 bits, estableciendo los bits correspondientes del registro de control.

Luego se deben enviar una serie de comandos al LCD para configurar su funcionamiento, como establecer el número de líneas, el tamaño de los caracteres y la posición del cursor.

Escritura

Para escribir en el LCD, se deben enviar primero los datos al puerto C del MCU.

Luego se debe establecer la señal RS del LCD en 1 para indicar que se van a enviar datos y no comandos.

Se debe establecer también la señal E del LCD en 1 para indicar que se va a realizar una escritura.

Finalmente, se debe establecer la señal E del LCD en 0 para completar la escritura.

Ejemplo

¡Supongamos que se desea escribir la cadena “Hola mundo!" en la primera línea del LCD.

Primero se deben enviar los caracteres correspondientes al puerto C del MCU: 'H', 'o', 'l', 'a', ' ', 'm', 'u', 'n', 'd', 'o', '!', y los bits correspondientes de control (RS=1, E=1, E=0).

Para establecer la posición del cursor, se debe enviar el comando correspondiente al LCD, por ejemplo: 0x80 para la posición 0 de la primera línea.

Luego se debe enviar la cadena de caracteres como se explicó anteriormente.

En resumen, para manejar el LCD conectado al puerto C del MCU presente en la tarjeta FACIL_08SH se deben configurar el puerto como salida y el LCD en modo de 8 bits, y luego enviar comandos y datos al LCD a través del puerto C del MCU. Es importante conocer la configuración y los comandos necesarios para utilizar adecuadamente el LCD en aplicaciones específicas.

$$cmdpos = 128 + (nren - 1)64 + ncol - 1 \cdots\cdots (1)$$

Donde nren es el número del renglón (1 ó 2),y ncol (1 a 16), es el número de la columna, a donde se va a colocar el siguiente carácter que se envíe al LCD como byte dato, cuyo valor es el código ascii del caracter que se va a desplegar.

Interrupciones periódicas

Las interrupciones periódicas son aquellas que se generan en intervalos regulares de tiempo.

Estas interrupciones son muy útiles en aplicaciones que requieren un control preciso del tiempo, como en sistemas de control de procesos y en aplicaciones de medición de tiempo.

Las interrupciones periódicas se pueden generar a través de una variedad de fuentes, como un temporizador interno del microcontrolador o un cristal de cuarzo externo.

Configuración

Para configurar interrupciones periódicas, se debe primero seleccionar la fuente de interrupción apropiada.

Luego, se deben configurar los registros del temporizador para establecer el intervalo de tiempo entre interrupciones.

Es importante tener en cuenta la frecuencia del reloj del microcontrolador y el valor de preescala que se va a utilizar para calcular el tiempo de interrupción.

Rutina

Una vez que se ha configurado la interrupción periódica, se debe crear una rutina de interrupción para manejarla.

Esta rutina se ejecutará cada vez que se genere una interrupción y se encargará de realizar las tareas necesarias en ese momento.

Es importante tener en cuenta que la rutina de interrupción debe ser lo más corta y eficiente posible, para no afectar el funcionamiento normal del programa.

Ejemplo

Supongamos que se desea medir la temperatura de un horno cada 5 segundos.

Se puede configurar un temporizador interno del microcontrolador para generar una interrupción cada 5 segundos.

Se puede crear una rutina de interrupción que lea la temperatura del horno y la guarde en una variable.

Luego, se puede utilizar esta variable en el programa principal para realizar acciones específicas, como encender un ventilador para enfriar el horno si la temperatura es demasiado alta.

En resumen, las interrupciones periódicas son una herramienta importante en el manejo de microcontroladores para el control preciso del tiempo. La configuración adecuada de las interrupciones y la creación de una rutina de interrupción eficiente son fundamentales para su correcto uso en aplicaciones específicas.

De acuerdo con lo explicado en la lámina anterior el número de cuentas requerido (NC) y la cuenta tope requerida estarían dados por:

$$NC = \frac{T_{ovf}}{T_{ck}} \cdots\cdots (1)$$

$$CT = NC - 1 \cdots\cdots (2)$$

Donde Tck = 1/Fck es el periodo de la señal aplicada al contador.
Si el valor de NC obtenido no es entero, se debe considerar para éste al entero más próximo, esto haría que se produzca un error

Cálculo y colocación de la cuenta tope (CT) asociada con un determinado Tovf

Asumiendo que la señal de entrada 'Se' al preescalador es el reloj de bus, Tck = Tb, véase la figura presente en la lámina 7, y que se usa un determinado preescalamiento 'pe'. Considerando las ecuaciones (1) y (2) mostradas en la lámina 4, la cuenta tope 'CT' requerida para un determinado valor del tiempo entre overflows 'Tovf', se determina mediante la siguiente ecuación:

$$CT = Int\{\frac{T_{ovf}}{peT_b}\} - 1 \cdots\cdots (3)$$

Donde Tb es el periodo asociado con la frecuencia del reloj de bus, además, se considera el hecho de que CT debe ser un entero.

TEMPORIZADORES DEL MCU MC9S08SH32

Ejemplo 1: cálculo del valor de CT para un determinado Tovf

Supóngase que se desea que el tiempo entre overflows Tovf para el contador del temporizador sea 50 mS, si el reloj de bus tiene una frecuencia Fb = 20 MHz (Tb = 50 nS), determinar los posibles pares de valores para la cuenta tope CT y el preescalamiento 'pe'.

Primero calculamos con pe =1, usando la ecuación (3) considerando los valores explícitos implicados:

$$CT = Int\{\frac{50x10^{-3}}{(50x10^{-9})x(1)}\} - 1 = 999999$$

Dado que el valor obtenido no cabe en 16 bits, es claro que no se puede usar un preescalamiento unitario para los valores de Tovf y Tb implicados. Por lo tanto, se debe aumentar el valor de 'pe'. Puede verse que 16, es el primer valor hacia arriba de 'pe', que conduce a un valor de CT que cabe en 16 bits, este valor se obtiene aplicando la ecuación (3) con pe = 16 y el detalle de su cálculo se muestra en la siguiente lámina

Ejemplo 1: cálculo del valor de CT para un determinado Tovf

Con pe = 16 se tiene:

$$CT = Int\{\frac{50x10^{-3}}{(50x10^{-9})x(16)}\} - 1 = 62499 = 0xF423$$

Puede verse que si usamos pe = 32 ó 64 ó 128, los valores de CT obtenidos cabrían en 16 bits, y serían respectivamente: 31249,15624 y 7811.

De hecho, cualquiera de los 4 pares de valores para CT y 'pe' obtenidos, podrían usarse para que el tiempo entre overflows sea 50 mS, cuando
Fbus = 20 MHz

Ejemplo 2: Se desea que el temporizador 1 genere interrupciones periódicas cada 100 mS, suponiendo que Fbus = 20 MHz (Tb = 50 nS), determinar:

- Un par de valores viable para la cuente tope CT y el preescalamiento 'pe'.
- El valor valsc que debe cargarse en el registro TPM1SC.
- Esbozar un código en ensamblador que configure el temporizador 1 para fines de lo especificado en el enunciado de este ejemplo.

Ejemplo 2: Solución

Paso 1: Empleando la ecuación 3 puede verse que el mínimo valor de 'pe' para que CT tenga un valor que quepa en 16 bits es 32, siendo CT = 62499 = 0xF423.

Paso 2: Dado que pe = 32, a partir de la tabla 16- 2 de la página 247 del manual del MCU, se ve que npe = 5.

Paso 3: Empleando la ecuación 4, considerando que se han de generar interrupciones (h=1), cada vez que se de un overflow, el valor valsc que ha de cargarse en el registro TPM1SC es
valsc = 64(1) +8 +npe = 77 = 0x4D = 01001101.

Nótese que el bit 6 que corresponde al habilitador local TOIE se pondrá en uno lógico, lo que habilita localmente la instancia de interrupción por overflow del temporizador 1.

Un posible tramo de código en ensamblador para los fines de este ejemplo es:

```
mov #$f4,tpm1modh
mov #$23,tpm1modl ;carga cuenta tope CT

mov #$4d,tpm1sc ;pe = 32, toie = 1, Se = reloj de bus
```

Ejemplo 3: Se desea que empleando el temporizador 1 se produzca la siguiente cadencia en el ledpum de la tarjeta FACIL_08SH, (250 ms encendido/250 mS apagado), suponiendo que Fbus = 20 MHz (Tb = 50 nS), escribir un programa en ensamblador que realice esto empleando para ello el que se generen interrupciones periódicas cada 250 mS y en la rutina de servicio asociada, simplemente se complemente el bit PTA7, que es a donde está conectado el **ledpum**, véase la figura 4 del documento Gb_facil_08sh.pdf.

Empleando la ecuación (3) puede verse que el único par viable (CT, pe) es: pe = 128 y CT = 39061 = 0x9895.

Dado el valor de 'pe' requerido, npe = 7, y ya que se requiere que se de un requerimiento de interrupción cada que haya un overflow, debemos usar h=1 en la ecuación (4), por lo tanto, el valor de valsc que se debe cargar en el registro TPM1SC es:

$$valsc = 64(1) + 8 + 7 = 79 = 0x4F$$