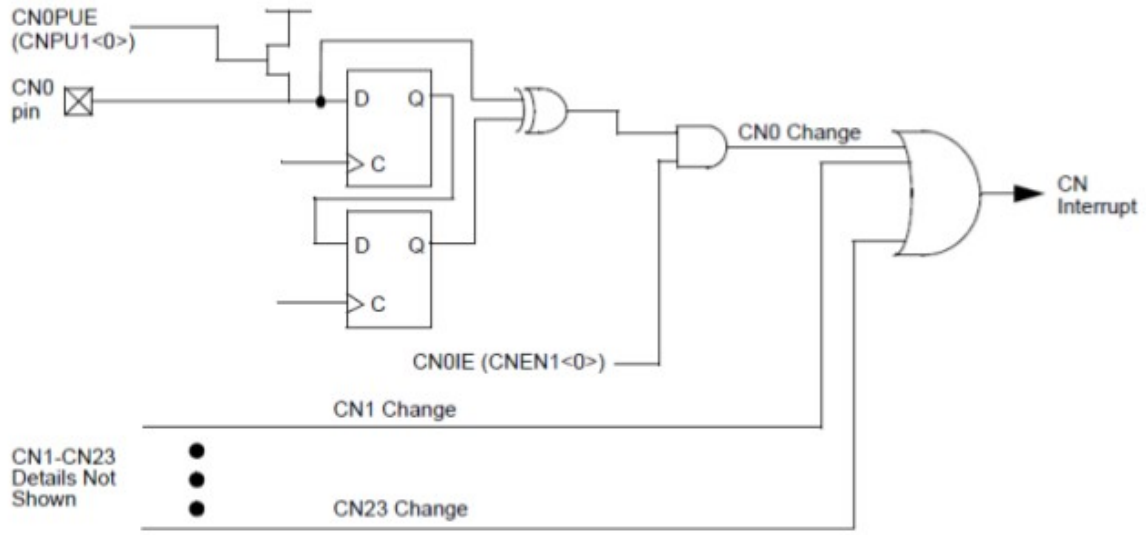
Módulo CN (*Change Notification*)

El módulo CN del microcontrolador PIC24 tiene 24 pines de este tipo asociados (CN0 – CN23) que pueden generar una petición de interrupción externa (CN). Es posible generar una interrupción cuando en un pin de tipo CN se detecta un cambio de tensión (H → L o L → H, de ahí el nombre). Para ello, el módulo CN dispone de dos registros, CNEN1 y CNEN2 (*Input Change Notification Interrupt Enable*) con los que se habilita o deshabilita individualmente cada una de los 24 patas susceptibles de generar interrupción. La figura muestra el diagrama de bloques que permite esta funcionalidad.

El registro CNEN1 contiene los bits CN15IE-CN0IE, mientras que el CNEN2 contiene los bits CN23IE-CN16IE. Poniendo a '1' el bit CNxIE, se habilita la interrupción del pin CNx (a '0' se deshabilita).



Además de un bit para cada uno de los pines, el módulo CN tiene un bit de habilitación/deshabilitación general de las interrupciones: bit CNIE en el registro de control IEC1.

Cada una de las 24 entradas asociadas a CN, tiene la posibilidad de utilizar una resistencia de pull-up interna. Para gestionar eso se utilizan otros dos registros del módulo CN: CNPU1 y CNPU2, que contienen los bits CnxPUE para cada uno de los 24 pines. Un '1' en estos bits, habilita el pull-up, mientras que un '0' lo deshabilita.

Resumiendo, Si se quiere gestionar algún pin de tipo CN por interrupción, será necesario habilitar la interrupción de dicho pin en alguno de los registro CNEN (CNEN1 o CNEN2), así como la interrupción general del módulo en el registro IEC correspondiente.

La rutina de atención correspondiente al módulo CN se denomina **`_CNInterrupt`** (consúltese la tabla del vector de interrupciones). En dicha rutina escribiremos el código que queremos que se ejecute cada vez que el módulo interrumpa, y dicho código dependerá del comportamiento que se quiera conseguir.

La cabecera de la rutina de atención a esta interrupción será la siguiente:

`void _ISR_NO_PSV _CNInterrupt()`

y antes de acabar, su última instrucción pondrá el flag de interrupción IF asociado al módulo CN a 0: **`IFS1bits.CNIF = 0`**

```
void _ISR_NO_PSV _CNInterrupt()
{
    ...//  codigo apropiado al tratamiento deseado
    _IFS1bits.CNIF = 0; // borrar el flag de interrupciones de CN
}
```

¡CUIDADO!: El espacio en blanco entre la macro **`_ISR_NO_PSV`** y el nombre de la rutina, **`_CNInterrupt()`**, es totalmente necesario. Si no se pone no hay error de compilación pero el código generado es incorrecto. De hecho, esa declaración de la rutina de atención es una abreviatura, ya que la macro **`_ISR_NO_PSV`** corresponde a la siguiente definición (la tenemos en el fichero **`commons.h`**):

```
#define _ISR_NO_PSV __attribute__((interrupt, no_auto_psv))
```

Ideas generales a tener en cuenta cunado escribamos una rutina de atención:

- La ejecución de una rutina de atención debe ser rápida y eficiente, siendo una característica importante que no consuma mucho tiempo. Así pues, evitaremos dentro de ellas funciones complejas que sobrecarguen la CPU (por ejemplo, instrucciones de cálculo como multiplicaciones, divisiones etc).
- JAMÁS se debe escribir dentro de una rutina de atención un bucle de espera, ni nada que se le parezca.
- En general, el objetivo de la rutina de atención será obtener los datos asociados al evento ocurrido para dejarlos en variables a utilizar por el programa principal, que será el encargado de procesar dichos datos y actuar en consecuencia.