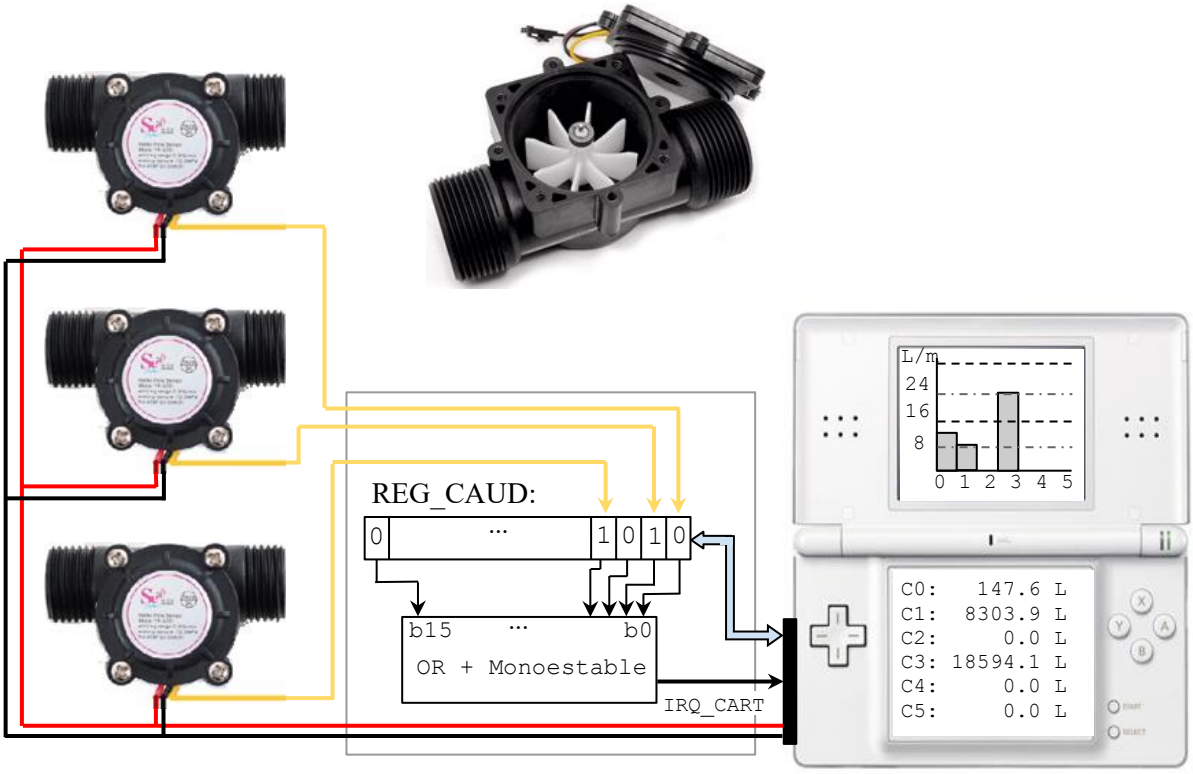
Problema 32: Caudalímetros (Ex. 2ª Conv. 2019-20)

Se propone conectar a la NDS un conjunto de hasta 16 caudalímetros tipo YF-S201 (hoja de especificaciones en hobbytronics.co.uk/datasheets/sensors/YF-S201.pdf). Este tipo de sensores constan de una hélice dentro de un receptáculo que se conecta a una tubería de 1/2 pulgada de diámetro. El paso del líquido hace girar la hélice, la cual incorpora un imán que provoca pulsos electrónicos sobre un transistor (efecto Hall). El transistor y el resto de los componentes electrónicos están fuera del receptáculo, de modo que no entran en contacto con el líquido. A continuación se muestra la foto de un sensor con el receptáculo abierto, junto con un esquema de conexión de varios caudalímetros con una interfaz electrónica que se conecta al puerto GBA de la NDS:



La interfaz proporciona un único registro de 16 bits, de nombre simbólico REG_CAUD. El propósito de este registro es doble:

- Lectura: en cada bit se puede leer un 1 si el caudalímetro que está conectado a ese bit ha generado un pulso desde la última vez que se reinició el bit,
- Escritura: se pueden poner a 0 un conjunto de bits del registro, escribiendo una máscara con un 1 en cada bit a reiniciar.

Además, la interfaz incluye una puerta OR que agrupa todos los bits del registro, de modo que genera una petición de interrupción (IRQ_CART) cuando hay algún bit a 1. Un circuito

monoestable adicional asegura que, cuando se escribe el registro, la señal IRQ_CART se desactiva durante unos microsegundos, para volver a reactivarse en caso de que se hubieran quedado algunos bits a 1 sin reiniciar.

El programa a implementar debe visualizar, en todo momento, los litros que han pasado por cada caudalímetro desde que se inició dicho programa (consumo), además de visualizar gráficamente una estimación del flujo instantáneo (caudal) del líquido que pasa por cada caudalímetro.

Para poder realizar los cálculos a partir del número y la frecuencia de pulsos (Freq) que genera cada caudalímetro, el fabricante nos proporciona los siguientes datos:

- `1 pulso ≈ 2,22 mililitros`
- `1 litro ≈ 450 pulsos`
- `Flujo (litros/hora) ≈ Freq (Hz) * 7,5`
- `Rango del flujo (litros/hora) ≈ [100..1800]`

Sin embargo, queremos visualizar el flujo en litros por minuto, de modo que vamos a trabajar con las siguientes conversiones:

- `Flujo (litros/minuto) ≈ Freq (Hz) * 7,5 / 60 = Freq (Hz) / 8`
- `Rango del flujo (litros/minuto) ≈ [1..30]`

Se dispone de las siguientes rutinas ya implementadas:

Rutina	*Descripción*
`inicializaciones()`	Inicializa el *hardware* (pantalla, interrupciones, etc.)

`inicializar_timer0(`
` unsigned char preescalar,`
`  short divFreq)`
	Inicializa el *timer* 0, seleccionando una de las 4 posibles frecuencias de entrada según el parámetro `preescalar` (0: `33513,98 kHz`; 1: `523,656 kHz`, 2: `130,914 kHz`, 3: `32,7285 kHz`) y utilizando el divisor de frecuencia indicado en `divFreq`.

`tareas_independientes()`	Tareas que no dependen del cálculo del caudal ni del consumo, por ejemplo, control de la presión de cada tubería (tiempo de ejecución < 50 ms)

`grafico_barras(`
`  unsigned short bars[],`
`  unsigned char num_bars,`
`  unsigned short max_val)`
	Dibuja en la pantalla superior de la NDS un gráfico de barras según los valores del vector `bars[]`, con número de elementos `num_bars`, escalando la altura de las barras respecto a un valor máximo `max_val` (tiempo de ejecución < 5 ms)

`swiWaitForVBlank()`	Espera hasta el próximo retroceso vertical

`printf_XY(unsigned char X,`
`  unsigned char Y,`
`  char *format, ...)`
	Escribe un mensaje en la pantalla inferior de la NDS, a partir de la posición de pantalla con coordenadas `X` (núm. columna, de 0 a 31) e `Y` (núm. fila, de 0 a 23) (tiempo de ejecución < 100 µs)

Con el apoyo de las rutinas anteriores, el programa a realizar debe ejecutar las tareas independientes concurrentemente con el contaje de pulsos de los caudalímetros conectados a la interfaz, que se deberá realizar con la RSI del dispositivo. Para obtener la frecuencia de los pulsos de cada caudalímetro, se deberá utilizar la RSI del *timer* 0 configurada a un hercio. Además, cada segundo se deberá representar gráficamente el flujo de cada caudalímetro con un gráfico de barras (pantalla superior), y el contaje total de litros de cada caudalímetro como una lista de números (pantalla inferior), con un formato como el del ejemplo (para 6 caudalímetros):

```
C0:   147.6 L
C1:  8303.9 L
C2:     0.0 L
C3: 18594.1 L
C4:     0.0 L
C5:     0.0 L
```

Para gestionar el valor de los contadores y de los caudales se pide usar las siguientes variables y definición:

```
#define MAX_CAUD  6
unsigned int cont_puls[MAX_CAUD];     //contadores de pulsos totales
unsigned short caudal[MAX_CAUD];      // caudales (litros/minuto)
```

La definición `MAX_CAUD` permitirá adaptar el código fuente al número máximo de caudalímetros de cada instalación, con el fin de que el programa no tenga que realizar recorridos de vector innecesariamente largos. Se supone que los caudalímetros se conectarán todos en los bits bajos del registro REG_CAUD.

Para obtener el valor de los contadores con la máxima exactitud posible, se sugiere utilizar contadores de pulsos en vez de contadores de litros, puesto que la conversión de pulsos a litros es aproximada, de modo que es preferible realizarla con el máximo número pulsos posible (el total) para evitar pérdida de precisión.

Por otro lado, el vector de valores de caudal podría haberse definido como `unsigned char` en vez de `unsigned short`, dado el rango de trabajo del flujo en litros por minuto. Sin embargo, se ha decidido usar un tipo de datos más grande de lo necesario para que dicho vector se pueda pasar directamente por parámetro a la rutina auxiliar `grafico_barras()`. Por último, aunque el rango de trabajo del valor del flujo sea [1..30], vamos a suponer que el valor cero también será admisible como resultado de los cálculos.

Se pide:

Programa principal y variables adicionales en C, RSI del dispositivo y RSI del *timer* 0 en ensamblador.