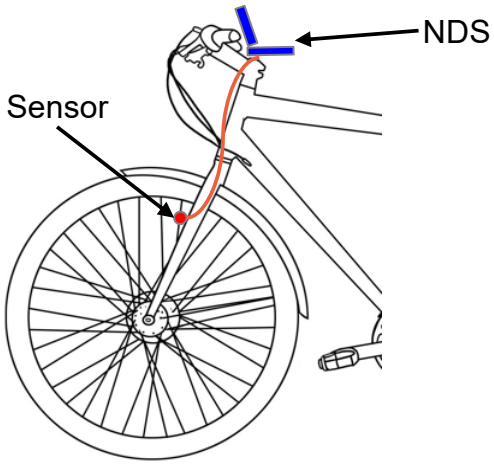
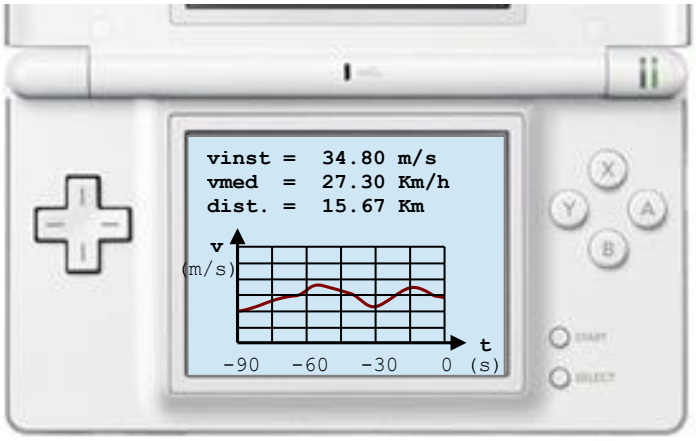
Problema 17: Velocímetro para bicicletas (Ex. 1ª Conv. 2015-16)

Se propone realizar un programa para calcular la velocidad instantánea, la velocidad media y la distancia recorrida por una bicicleta, a partir de la señal de un sensor instalado en la horquilla de la rueda delantera. A continuación se muestra un esquema del sistema y la visualización de los datos en la pantalla inferior de la NDS:





Mediante un cable, el sensor envía un pulso cada vez que un rayo de la rueda pasa por delante suyo. El cable está conectado a la NDS, de forma que el pulso genera una petición de interrupción por la línea IRQ_CART, la cual activará una rutina de servicio de interrupción específica (`RSI_sensor`). Con esta RSI se podrá estimar la distancia recorrida por la rueda en todo momento.

Por otro lado será necesario controlar el tiempo, con el fin de poder calcular velocidades. Para este propósito se propone utilizar la RSI del *timer* 0, que generará interrupciones a una frecuencia de 2 Hz. La elección de esta frecuencia es porque se requiere que los cálculos de velocidades y distancia se actualicen cada medio segundo. Sin embargo, debido al retardo que pueden introducir las tareas independientes, la actualización en pantalla de los resultados se podría demorar hasta un segundo; este comportamiento se considerará normal.

Se dispone de las siguientes rutinas, ya implementadas:

Rutina	*Descripción*
`swi 9`	Rutina de la BIOS para división entera; parámetros (R0: dividendo, R1: divisor); resultado (R0: cociente, R1: resto, R3: cociente absoluto)

`inicializaciones()`	Inicializa el *hardware* (pantalla, interrupciones, *timers*, etc.) y las variables globales
`tareas_independientes()`	Tareas que no dependen de los cálculos de velocidad y distancia (ej. monitorización del ritmo cardíaco del ciclista en la pantalla superior); t. ejecución entre 0,1 y 1 s
`scanKeys()`	Captura la pulsación actual de las teclas
`int keysDown()`	Devuelve el estado de las últimas teclas pulsadas
`swiWaitForVBlank()`	Espera hasta el próximo retroceso vertical
`representarInfo(int vinst, int vmed, int dist, short data[], int index)`	Muestra la información por pantalla inferior según el valor de los parámetros: velocidad instantánea (en cm/s), velocidad media (en dam/hora), distancia total (en cm), vector de velocidades instantáneas (en cm/s), índice posición actual del vector

Para poder realizar todos los cálculos, se proponen las siguientes variables globales:

```
unsigned short Perimetro;     // perímetro de la rueda (en cm)
unsigned char Nrayos;         // número de rayos de la rueda
unsigned short Drayos;        // número de rayos por segundo
unsigned short Vinst;         // velocidad instantánea (en cm/s)
unsigned int Vmed;            // velocidad media (en dam/hora)
unsigned int Tdist;           // distancia total (en cm)
unsigned int Ttiempo;         // tiempo total (en semisegundos)
unsigned char ind;           // indice posición actual buffer Vinst
unsigned short buffVinst[180];
```

Los valores de `Perimetro` y `Nrayos` dependerán de las características de la rueda instalada; la función de inicialización cargará el valor correspondiente en estas variables, por ejemplo, `Perimetro` = 207 cm, `Nrayos` = 32 rayos.

El valor de `Drayos` deberá contar el número de rayos que se detectan por unidad de tiempo. Este valor permitirá calcular la velocidad instantánea, que se tiene que almacenar en `Vinst`, en centímetros por segundo.

Para calcular la velocidad media, almacenada en `Vmed` (en decámetros por hora), será necesario registrar la distancia total recorrida desde que se inició el programa, así como el tiempo total. La distancia se almacenará en `Tdist` (en cm), mientras que el tiempo se almacenará en `Ttiempo` (en semisegundos). Además, si se pulsa la tecla 'START' en cualquier momento, todos los contadores se deberán poner a cero.

Por último, también se pide que cada semisegundo se almacene la velocidad instantánea en un vector de 180 posiciones, cada vez en una posición diferente (`ind`), de forma circular, es decir, cuando se llegue a la última posición se debe empezar por la primera otra vez. Este vector, de nombre `buffVinst[]`, permitirá representar la evolución de la velocidad instantánea en los últimos 90 segundos. El contenido de este vector también se deberá poner a cero al pulsar la tecla 'START'.

Toda esta información se debe pasar a la función `representarInfo()`, con las unidades indicadas para las variables globales. Internamente, esta función transformará dichas unidades a las habituales (m/s, Km/h, Km). Hay que observar que las variables globales propuestas utilizan fracciones de dichas unidades (cm/s, dam/h, cm) para no tener que operar con decimales, ya que el procesador ARM solo trabaja con valores enteros.

Se pide:

Programa principal y variables globales en C, RSI del *timer* 0 y RSI del sensor en ensamblador.