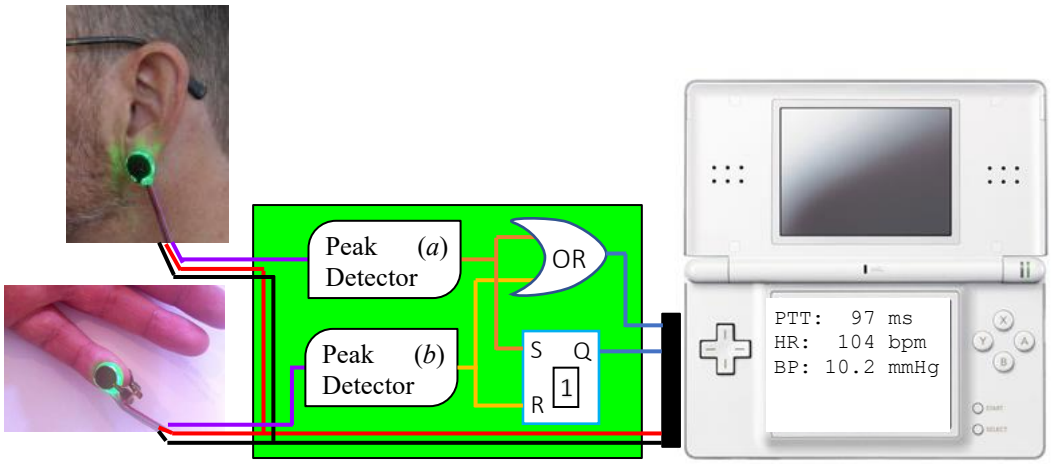
Problema 33: Tiempo de tránsito del pulso (Ex. 1ª Conv. 2020-21)

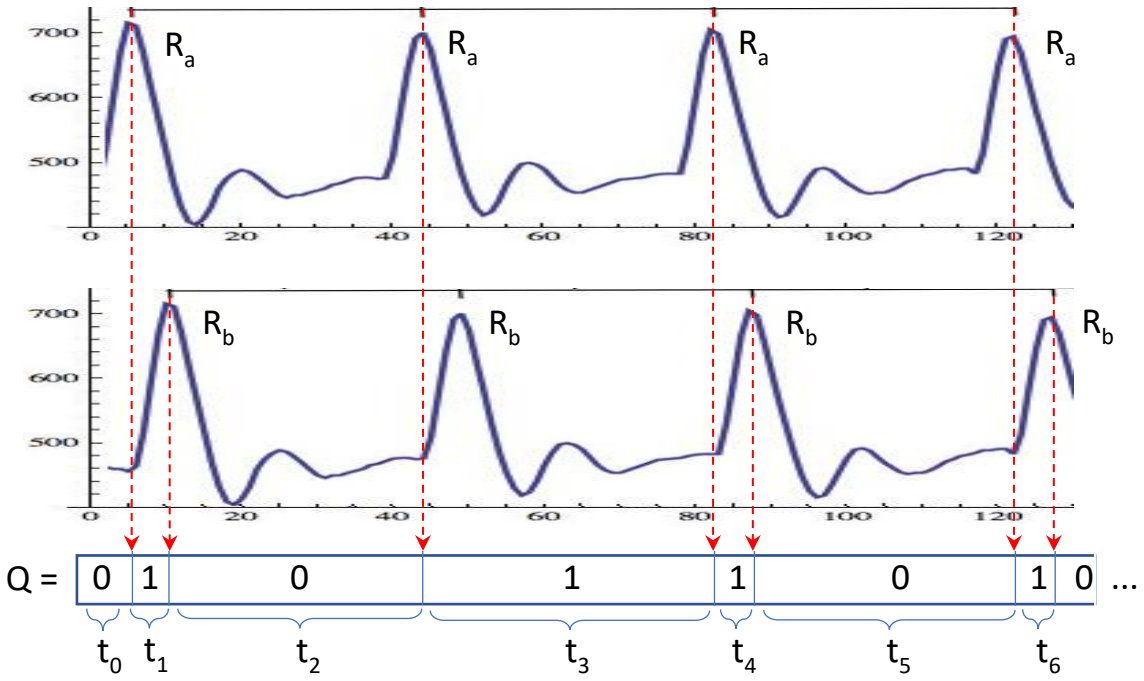
Se propone conectar a la NDS una interfaz electrónica equipada con dos sensores de pulso cardíaco (por ejemplo, del fabricante pulsesensor.com). Uno de los sensores se colocará en el lóbulo de la oreja izquierda y el otro en el extremo de un dedo de la mano izquierda. El siguiente esquema muestra la conexión de los sensores con una interfaz electrónica diseñada expresamente para comunicarse con la NDS:



En dicha interfaz hay un circuito electrónico *Peak Detector* para cada sensor, capaz de detectar el pico R de la señal generada por el sensor (ver gráficos en la siguiente página). La salida de cada detector de picos es un pulso electrónico de unos 5 microsegundos de duración, cuyo propósito principal es generar una interrupción en el computador. Debido a que solo disponemos de un único pin para enviar señales de petición de interrupción (IRQ_CART), las salidas de los dos detectores se han unido con una puerta OR, de modo que se generará una petición de interrupción si se detecta un pico R en alguno de los dos sensores.

Además, se ha incorporado un biestable S-R (Set-Reset) asíncrono, que memoriza un 1 cuando S = 1 y memoriza un 0 cuando R = 1. Cuando las dos entradas S y R valen 0, el biestable mantiene el último valor memorizado (en el esquema se muestra un 1, pero puede ser un 0). La salida Q envía el contenido del biestable hacia el computador mediante el bit 7 de un registro de Entrada/Salida de 16 bits mapeado en la dirección absoluta de memoria 0x0A000000. Además, puede ser que los otros bits del mismo registro se utilicen para otras tareas que no se detallan en este enunciado (tareas independientes).

El gráfico de la página siguiente muestra un ejemplo de evolución temporal (cronograma) de las señales de los sensores. Como los sensores se encuentran en posiciones del cuerpo separadas, las ondas de presión generadas por el latido del corazón siempre llegan a los sensores en momentos distintos. El propósito principal del programa a implementar será calcular la diferencia de tiempo en la detección de los picos R_a y R_b de un mismo latido, lo cual se denomina **tiempo de tránsito del pulso** (PTT: *Pulse Transit Time*).



(gráfico basado en contenidos de https://community.wolfram.com/groups/-/m/t/272663)

En el gráfico anterior se muestra la generación de las interrupciones debidas a la detección de un pico R_a o R_b, mediante flechas rojas de línea discontinua. También se muestra la evolución del valor de la salida Q del biestable S-R, así como una representación simbólica de los tiempos entre interrupción e interrupción (t_0, t_1, t_2, etc.).

Como se puede observar, se ha omitido la interrupción correspondiente al segundo pulso R_b. Está omisión representa la posibilidad de que cualquiera de los dos detectores de pico falle puntualmente, puesto que existen múltiples factores que pueden perturbar el funcionamiento del circuito (ruido de señal, mala colocación del sensor, etc.). Sin embargo, vamos a suponer que nunca fallarán los dos sensores simultáneamente (en el mismo latido).

Para mantener la máxima precisión posible en los cálculos posteriores, se pide explícitamente que se ignoren los valores erróneos. Para ello, hay que comprobar si el valor Q ha cambiado respecto a la interrupción anterior. En el ejemplo anterior, cuando el detector de picos *b* falla, Q no cambia porque la entrada Reset del biestable no se activa, mientras que la entrada Set del biestable recibe dos activaciones consecutivas. En este caso, el valor de t_3 se debe descartar.

El programa a realizar deberá capturar todos los tiempos válidos entre picos R, distinguiendo entre tiempos cortos y tiempos largos. Vamos a suponer que los tiempos cortos siempre serán los capturados con los picos R_b, es decir, cuando el biestable pase de 1 a 0, mientras que los tiempos largos serán los casos opuestos. Todos estos tiempos se irán almacenando en sendos vectores circulares de `MAX_T` posiciones cada uno (ver definiciones de variables globales).

Cuando se hayan capturado como mínimo `MAX_T/2` pulsos en ambos sensores, el programa deberá realizar la media aritmética de los últimos `MAX_T` tiempos cortos y largos (media móvil). El programa debe escribir por pantalla el tiempo de tránsito de pulso (PTT), que será igual a la media móvil de los tiempos cortos, expresada en milisegundos. Además, el programa calculará la frecuencia de los latidos del corazón (HR: *heart rate*), escribiendo por pantalla su valor entero expresado en bpm (*beats per minute*: latidos por minuto). El cálculo de la frecuencia cardíaca se puede obtener teniendo en cuenta que el periodo de un latido es la suma del tiempo corto más el tiempo largo correspondiente. El programa también realizará una estimación de la presión arterial (BP: *blood preasure*) a partir de los valores actuales de PPT y HR, utilizando una rutina de soporte ya implementada. El programa escribirá por pantalla el valor de BP en milímetros de mercurio (mmHg), utilizando un dígito decimal.

Los valores de PTT, HR y BP se escribirán cada uno en una línea individual, añadiendo una línea de separación respecto a los valores calculados anteriormente. Cada vez que se calculan los nuevos valores de estos parámetros se denomina "un ciclo de cálculo". En el esquema inicial se muestra un ejemplo de salida de información por pantalla para un único ciclo de cálculo.

El diseño del programa a realizar debe permitir entrelazar la captura de los tiempos entre picos con la ejecución de las tareas independientes a dicha captura. Para ello es necesario utilizar la rutina de servicio de interrupción de las peticiones generadas por la interfaz electrónica, que denominaremos como `RSI_Peak`. Para calcular los tiempos entre picos se deberá utilizar un cronómetro de precisión que usará los *timers* 1 y 2 de forma encadenada.

Se dispone de las siguientes rutinas ya implementadas:

Rutina	*Descripción*
`inicializaciones()`	Inicializa el *hardware* (pantalla, interrupciones, *timers*, etc.)
`tareas_independientes()`	Tareas que no dependen del cálculo del tiempo de transito del pulso, por ejemplo, cálculo del nivel de oxigeno en sangre (tiempo de ejecución < 100 ms)

`calcular_presion(`
`  unsigned short pulse_tt,`
`  float h_rate)`
	Calcula la presión arterial a partir del tiempo de tránsito del pulso (en ms) y de la frecuencia cardíaca (en bpm), devolviendo el resultado en mmHg (`float`, tiempo de ejecución < 350 microseg.)

`cpuStartTiming(int timer)`	Inicia un cronómetro de precisión usando el *timer* que se pasa por parámetro (0, 1 o 2) y el siguiente *timer*
`cpuGetTiming()`	Devuelve el contaje de tics del cronómetro de precisión desde que se inició (`unsigned int`)

`swiWaitForVBlank()`	Espera hasta el próximo retroceso vertical
`printf(char *format,...)`	Escribe un mensaje en la pantalla inferior de la NDS

Además, se propone el uso de las siguientes constantes y variables globales:

```
#define MAX_T   8                    // número máximo de tiempos
#define TICS_MS 33513                // tics por cada milisegundo

unsigned int vect_tc[MAX_T];         // vector de tiempos cortos
unsigned int vect_tl[MAX_T];         // vector de tiempos largos
unsigned char ind_tc = 0;            // índice actual en vect_tc
unsigned char ind_tl = 0;            // índice actual en vect_tl
unsigned char num_tc = 0;            // número de tiempos cortos
unsigned char num_tl = 0;            // número de tiempos largos
```

Las variables `ind_tc` e `ind_tl` permitirán indexar los vectores para saber en qué posición se debe guardar un nuevo tiempo capturado. Por otro lado, las variables `num_tc` y `num_tl` permitirán saber cuántos tiempos se han guardado desde el último ciclo de cálculo. Como los vectores serán circulares se puede suponer que siempre estarán llenos, de modo que estas variables no se deben interpretar como el número total de valores dentro de los vectores, sino como el número de valores capturados en el ciclo de cálculo actual.

Por otro lado, la definición `TICS_MS` corresponde al número de tics contabilizados por el cronómetro de precisión en un milisegundo.

A modo de ejemplo se muestran algunos rangos típicos de los valores que tiene que manejar el programa a implementar:

- Tiempo corto (PTT): 60 – 260 ms
- Tiempo largo: 210 – 1240 ms
- Frecuencia cardíaca (HR): 40 – 220 bpm
- Presión arterial (BP): 70 – 160 mmHg

En el contexto de este problema, la presión arterial se refiere a la presión intermedia entre la sistólica (máxima) y la diastólica (mínima).

Se pide:
Programa principal y variables globales adicionales en C, RSI del dispositivo en ensamblador.