Ejercicio 1:

1) a) TEMP_STATUS ⟼ 0xFFF0
ATM_STATUS ⟼ 0xFFF1
WIND_SPEED ⟼ 0xFFF2

Luego el código queda:

```
sensar:   CMP [0xFFF0], MAX_TEMP
          JG alarma
          CMP [0xFFF1], MAX_ATM
          JG alarma
          CMP [0xFFF2], MAX_WIND_SPEED
          JG alarma
          JMP sensar
alarma:   CALL sonarAlarma
          JMP sensar
```

b) En el caso de que nunca suene la alarma (los valores máximos nunca se alcanzan) vas que la lectura de cada registro de E/S se realiza una vez por cada vez que se salta a sensar. Siendo que se salta a sensar 1 vez cada 7 instrucciones, y todas ellas tardan t seg en ejecutarse, su frecuencia es de $\frac{1}{7t}$ Hz y luego el muestreo se realiza con una frecuencia de $\frac{1}{7t}$ Hz.

Ejercicio 2:

2) a) En caso de que el programa sea interrumpido durante la ejecución de la instrucción ADD R0, R1, primero se terminará de ejecutar la instrucción, lo cual hace que V=1 (se produce overflow al sumar R0 = 0x8000 con R1 = 0xFFFF) y luego el estado de los flags en ese punto (PSW) se guardará en el stack para ser restaurado luego de ejecutarse la rutina de atención a la interrupción. Seguido a ello se realizará todo el proceso para llamar a dicha rutina, donde al ejecutarse la instrucción ADD R0, R1 (con R0 = 0x0000 y R1 = 0x0000) los valores de todos los flags serán 0. Sin embargo, como la última instrucción ejecutada de la rutina es RET y no IRET, siendo que tanto el valor del PC y de PSW estaban en la pila, sólo se restaurará el PC, con lo cual el estado de los flags en dicho punto será reemplazado por el proveniente de la rutina de atención a la interrupción. Esto hace que el flag V pase de 1 a 0 y que en vez de ejecutarse la rutina Hubo Overflow (lo cual era lo esperado por el salto condicional), ejecuta no Hubo Overflow

b) Para que la ejecución de la rutina de atención a la interrupción sea transparente, habría que cambiar la instrucción RET por IRET, lo cual hace que no sólo se restaure el punto de ejecución, sino también el estado de los flags del programa previos a ser interrumpidos. Con esta modificación la ejecución de la instrucción ADD R0, R1 será la esperada, incluso si es interrumpida.