SFR04.C

```
#include "srf04.h"



void Init0_Sonar(void)  // init totale
{
 TRIGGER=RESET;
 sonar.srff.flags=0;    // raz flags
 sonar.distance=100;    // 1m en init
 sonar.cptflash=0;
 Init_Sonar();
 stop=danger=CLEAR;
}



void Init_Sonar(void)              // init preserve distance et flag trigger
{
 sonar.cpt=sonar.ptmr0=sonar.ltmr0=0;
 sonar.srff.flags &= 0b11000000;   // raz flags pas flash et col
 sonar.timer=0;                    // inutile d'initialiser l et p tmr0
}



// demande mesure
void Dem_Mes_Sonar(void)
{
 unsigned char li;

  if(sonar.srff.flag.trig)
    {
    Init_Sonar();

    for(li=0, TRIGGER=SET; li<16; li++);   // gen pulse
//  TRIGGER=SET;
    TRIGGER=RESET;

    sonar.srff.flag.dem=sonar.srff.flag.run=SET;
    }
}



// si mesure OK
void Conv_Mes_Sonar(void)
{
 unsigned int ldistance;

  if(sonar.srff.flag.done==SET) // si le créneau à été vu
    {
    // raz des flags sonar en préservant les flags leds
    sonar.srff.flags&=0b11000000;

    // recupère la durée du créneau en TICKs de T0
    ldistance=((sonar.cpt*256)-(unsigned int)sonar.ptmr0)+(unsigned int)sonar.ltmr0;

    // transforme la durée en µsec (TICK en 1/10µsec)
    ldistance =_________________;
    // transforme en cm
    ldistance =_________________;

    // distance en cm stockée dans la structure sonar
    sonar.distance = (long int)(DIST_DANGER-5);
    //sonar.distance = (long int)ldistance;
    }
}



void TMR0_Sonar(void)   // a faire dans int TMR0
{
   sonar.timer++;       // inc timer demande mesure
   sonar.cptflash++;
                        // gestion pulse > 1 tour compteur T0 (pb roll-over)
   if(sonar.srff.flag.cpt)
        {
         sonar.cpt++;
        }

   if(sonar.timer == NB_SONAR) // demande to do?
    {
    sonar.srff.flag.trig=SET;
    }
   if(sonar.cptflash == NB_FLASH) // flash leds
    {
```

```
    sonar.cptflash=0;
    sonar.srff.flag.ledflash=SET;
    }
}



void int_Sonar(void)
{
 if(sonar.srff.flag.dem && PULSE)  // int front montant
    {
    sonar.srff.flag.dem=CLEAR;
    sonar.ptmr0=TMR0;  // recup TMR0
    sonar.srff.flag.wait=sonar.srff.flag.cpt=SET;
    }
    else if(sonar.srff.flag.wait && !PULSE) // int front desc...
        {
        sonar.srff.flag.wait=sonar.srff.flag.cpt=CLEAR;
        sonar.ltmr0=TMR0;  // recup TMR0
        sonar.srff.flag.done=SET;
        }
}



void LedCol_Flash(void)
{
    // le bit drapeau sonar.srff.flag.ledflash
    // est mis à 1 FLASH_SEC*2 fois par seconde
 if(sonar.srff.flag.ledflash)
    {
      sonar.srff.flag.ledflash=CLEAR;
      LEDCOL^=1;
    }
}



void Gere_Led_Sonar(void)
{
 if(sonar.distance < DIST_STOP)
        {
        LEDCOL=LEDON;
        stop=sonar.srff.flag.ledcol=SET;
        }
    else
      {
       stop=sonar.srff.flag.ledcol=CLEAR;
       if(sonar.distance < DIST_DANGER)
            LedCol_Flash();
         else LEDCOL=LEDOFF;
      }
}
```