//#include<FreeRTOS.H>  
//#include<task.h>  
#include "C\_Extensions.h"  
#include "../../\_4\_Drivers/system/systemInit.h"  
#include "../../\_4\_Drivers/uart/uart.h"  
#include "../../\_4\_Drivers/gpio/gpio.h"  
#include "../../\_4\_Drivers/delays/delays.h"  
#include "../../\_3\_Hal/ST7920/ST7920.h"  
#include "../../\_2\_System/Timers/Timers.h"  
  
#include "main\_prv.h"  
//#include "timers.h"  
  
  
static void ExecuteTasks(TASK\_TYPE \* task\_list);  
void External\_Input\_OutputTasks(void \*);  
void Internal\_ProcessTasks(void \*);  
  
void Timer1ISR(void)\_\_irq //  This ISR is invoked every 1msec  
{  
  
Timers\_\_ServiceMs();  
Timers\_\_ServiceHMS();  
  
T1IR = ( T1IR | (0x01) ); ;  
    // Do interrupt code here  
  
    //UART\_TxString(C\_UartZero\_U8,"Timer interrupt occured\n");  
    VICVectAddr = 0x00;  
    T1PR  = 60; // 1 msec time  
    T1TCR = 0x01;  
}  
  
void TimerIntterupt\_\_Config()  
{  
  
  
  
//VICDefVectAddr  = (unsigned long int)timer0ISR; do not use when more than one interrupts are used  
VICVectAddr1 = (unsigned long int)Timer1ISR;  
VICVectCntl1 = 0x25;                 //bit 5 is enabled the slot to produce unique interrupt address for enabled timer0 interrupt  
VICIntEnable |= 0x00000020;          //enable timer0 interrupt  
  
// Init Timer - 1  
T1IR  = 0xff;         //clr the pending flags of interrupt  
T1PR  = 60;         //60 clock cycles = 1 uS  
T1PC  = 0;  
T1MCR = 0x0003;       //MR0I=interrupt when TC=MR and reset  
T1MR0 = 1000;         //match the final value (using )  
T1TCR = 0x02;         //reset the timer  
T1TCR = 0x01;         //enable the timer  
  
  
}  
  
int main()  
{  
SystemInit();  
UART0\_Init\_115200();  
UART1\_Init\_115200();  
  
UART\_TxString(0,"Security System Started\n");  
  
while(1)  
{

UART\_TxString(0,"Security System Running\n");  
UART\_TxString(1,"P1D1P2D2");// uart 1 is connected to Nodemcu , ESP8266  
DelayMS(3000);//3 sec delay  
  
}  
DelayMS(1000);  
  
}  
   
void External\_Input\_OutputTasks(void \*s)  
{  
//Initialize the External Input task modules  
ExecuteTasks(External\_InputInitialization);  
    for(;;)  
{  
  
    ExecuteTasks(External\_Input\_Tasks);  
    UART\_TxString(C\_UartZero\_U8,"Running External Input \_Output tasks\n");  
    }  
}  
   
void Internal\_ProcessTasks(void \*s)  
{  
//Initialize the Internal Process task modules  
ExecuteTasks(Internal\_ProcessInitialization);  
    for(;;)  
    {  
ExecuteTasks(Internal\_Process\_Tasks);  
    UART\_TxString(C\_UartZero\_U8,"Running Internal Process tasks\n");  
    }  
}  
  
  
  
static void ExecuteTasks(TASK\_TYPE \* task\_list)  
{  
  while (\*task\_list != NULL\_TASK)  
  {  
    (\*task\_list)();  
    task\_list++;  
  }  
}

|  |  |
| --- | --- |
|  |  |