Correio Eletrônico

utilizando MSP430 e sensor ultrassônico HC-SR04

Mattos, A. Guilherme
Universidade de Brasília
Faculdade Gama
Brasília, Brasil
guilherme.mattos.gam@gmail.com

Souza, Paulo Augusto M. F. de
Universidade de Brasília
Faculdade Gama
Brasília, Brasil
pauloaugustomiguelfonseca@gmail.com

Abstract — Serial communication is communication made bit by bit by the system. These days , most of the components and systems, use this type of communication. In MSP , we can be asynchronous transmission , it no clock signal , and communication must be previously known by the receiver and transmitter. Thus, there is a data transmission rate, referred to as baud rate , which is the time it takes to transmit bits in a time interval .

Keywords— MSP430, interrupts, Timer_A, mail

I. OBJETIVOS

Utilizar o microcontrolador MSP430, juntamente com um sensor ultrassônico para o projeto que consiste em avisar moradores de condomínios ou apartamentos quando as cartas de suas caixas de correio chegam e quantas cartas chegaram.

II. INTRODUÇÃO

O microcontrolador tem um sistema de clock, fazendo assim que ele tenha um tempo de funcionamento. O Timer_A é um timer que usa o clock do MSP, e também o mais completo. Possui um registrador, TAR, de 16 bits, o que permite a escolha do sistema de clock. Após escolhido o clock, então é feita a divisão da frequência desse clock, que pode ser de 1,2,4,8 vezes e logo em seguida o modo de contagem. Isso tudo é feito para que haja um melhor uso, dependendo da aplicação.

Comunicação serial é a comunicação feita bit a bit pelo sistema. Nos dias de hoje, a maioria dos componentes e sistemas, usam esse tipo de comunicação, como por exemplo a feita por periféricos que utilizam Usb. No MSP, podemos ter a transmissão assíncrona, nela não há sinal de clock, e a comunicação deve ser previamente conhecida pelo receptor e transmissor. Com isso, há uma taxa de transmissão de dados, chamada de baudrate, que é o tempo que se gasta para transmitir bits em um intervalo de tempo.

Esse método, permite por exemplo fazer o envio de caracteres. Com isso, definido, pode-se fazer envio de mensagens para usuários. Isso pode ser feito pelo UCAxTXBUFx, que é o buffer responsável pela transmissão de dados assíncronos. Com isso precisa ser feito uma definição do baudrate, e assim com as entradas seriais, pode ser escrito uma mensagem por exemplo para o usuário.

Os sensores ultrassom contém seu funcionamento baseado no principio físico de reflexão de onda. O sensor envia ondas, e calcula a distância, com o calculo do tempo em que se gasta para a onda ir e voltar para o sensor. A partir da variação de tensão na saída do sensor, é possível calcular a distância em que se encontra o objeto. No HC-SR04, temos os pinos, Vcc e GND, que são usados para alimentação do sensor, tem-se o pino Trigger, que é a entrada do sensor, para que seja enviado para ele sinais, para que sejam gerados sinais para se medir a distância. E também o pino Echo, que conforme a distância varia, a tensão varai no pino, de Vcc quando ele está longe, e 0V, quando o objeto está perto.

III. DESENVOLVIMENTO

A. Descrição do Software

O código para realização deste projeto consiste em 3 partes principais, a utilização do sensor ultrassônico, a comunicação serial e a lógica para contar e zerar o número de cartas.

Foram declaradas algumas variáveis globais que serão utilizadas nas interrupções, para o cálculo das distancias capturadas pelo sensor, como a variáveis “sensor” e “distancia”, e variáveis que podem ser modificadas de acordo com o horário que se queira iniciar o programa. A primeira função declarada no código é para o uso da comunicação serial, “uart_send” e juntamente com a utilização do comando “sprintf” pode-se escrever na tela a comunicação que será feita com o usuário.

Na função principal são definidas as saídas na P1 o Bit 0 que será o Led que será ligado apenas quando o sensor detecta uma distância menor que 25 cm, e o Bit como o trigger do sensor ultrassônico. Como entrada foi utilizado o Bit 3, que foi conectado ao ECHO do sensor, essa entrada pode habilitar a interrupção com base na borda de subida, e a interrupção causada por essa porta, juntamente com a interrupção do timer A0, que é chamada a cada milissegundo, são responsáveis pelo cálculo que converte o **tempo** que o

sensor demora pra retornar a responder em **comprimentos**, com base nesse cálculo a variável distancia recebe o valor da distância em cms que o sensor detecta.

A interrupção do TA1, é chamada a cada 0.1 segundos e utilizando variável "decsegundos' a cada interrupção ela é incrementada, indicando que a cada décimo de segundo essa variável recebe mais um, logo após utilizando ifs se essa variável chegar ao valor 10, ela é zerada e incrementada a variável "segundos" que representa os segundos, e quando essa variável chega a 60, ela é zerada e incrementada a variável "minutos", que segue a mesma lógica, zerando ao chegar em 60 e incrementando a variável "horas", e assim é criada a lógica pra indicar o horário que a carta chegou na caixa de correio do morador . de interrupção, para sair da interrupção e para que ela possa ser habilitada novamente.

Na função principal, a variável "x" representa o número de cartas presentes na caixa de correio e ela inicia com valor "0", usando "while x=0" manda pro usuário a mensagem indicando que ele tem 0 mensagens e logo após incrementando a variável x para não ficar repetindo a mensagem na tela. Se a distância for menor que 25 cm, menor que a distância da porta até o fundo da caixa, significa que entrou alguma carta, e o usuário recebe na tela que chegou uma carta, incrementa o x novamente e a medida que vai detectando novas cartas incrementando, se o usuário abrir a caixa, ou seja se a distância for maior que a da caixa, no caso se a distância for maior que 30cm, o programa aguarda 15 segundos e zera novamente a variável x indicando que o morador retirou as cartas e podendo repetir esse processo .

Caso o número de cartas passe de 9, a altura do sensor pode afetar o cálculo e mesmo sem entrar cartas ele pode detectar como se estivessem e para evitar que o programa fique acusando infinitas cartas foi definido esse limite de 9 cartas, e se chegar a 10 o valor de x o programa manda para o usuário a mensagem indicando que ele deve verificar seu correio pois está cheio.

B. Descrição de hardware.

Como mostrado na figura 1, o hardware consiste da Launchpad e do sensor ultrassônico HC-SR04. Os pinos Vcc e GND, foram ligados nos respectivos na placa, no qual são usados para alimentação do sensor. O pino P1.1 ligado ao Trigger do sensor, ele serve como saída da placa, para mandar o comando para o sensor emitir ondas e fazer a medição da distância do objeto. O Echo do sensor foi ligado ao pino P1.3, e foi definido como entrada. Conforme a distância do objeto do sensor, a tensão no Echo varia, isso faz com que fosse detectado o objeto a uma certa distância. Quanto mais perto do sensor a tensão tende a 0, e quanto mais longe, mais se aproxima de Vcc.

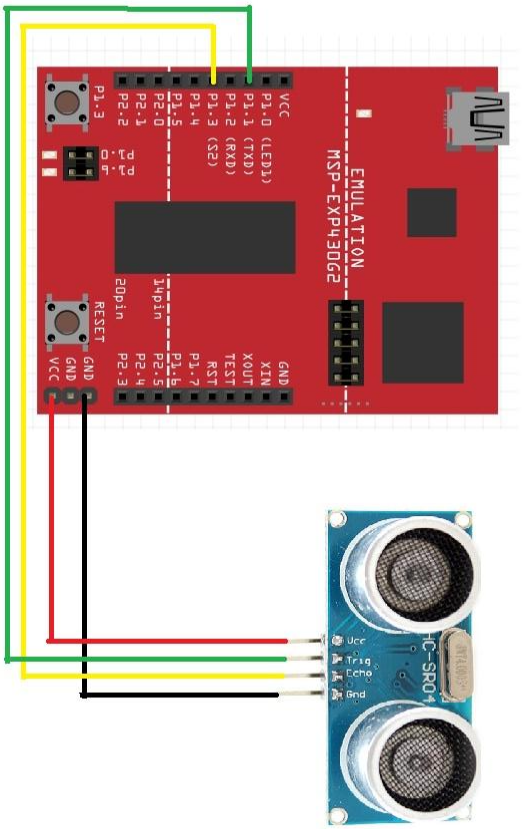

Figura 1 – Montagem do Hardware

IV. RESULTADOS

Os problemas encontrados na execução deste projeto foram como resetar a contagem após o morador recolher suas cartas e como por um limite se as cartas chegarem à altura do sensor e contar infinitamente as cartas.

Para resetar a quantidade de cartas assim que o usuário retirá-las a primeira ideia foi usar um botão, no entanto haveria problemas no caso do usuário esquecer, então a partir do próprio sensor foi criada a lógica que se ele medir uma distância maior que a da caixa, no caso maior que 30 cm, ele zera a variável que conta o número de cartas. E foi exatamente o que aconteceu, no momento que abre a porta do correio o sensor mede distâncias superiores as limitadas pela caixa, e após 15 segundo, que foi um tempo estimado em que o usuário retiraria suas cartas e teria fechado sua caixa de correios, aparece corretamente a mensagem que tem 0 cartas.

Para limitar a quantidade, e não gerar problemas fazendo infinitas contagens, foi definido o limite de 9 cartas, a partir de 10 o sistema manda uma mensagem para o usuário verificar sua caixa de correio, e mesmo incrementando a variável o programa não mostra mais para o usuário nenhuma mensagem

até que ele retire suas cartas e zere novamente a contagem, isso porque os ifs que são responsáveis pela comunicação do usuário só geram mensagem com x menor ou igual a 10.

V. CONCLUSÃO

Esse projeto foi baseado na ideia de utilizar a internet para alguma aplicação, “internet das coisas”, mesmo não utilizando no projeto, foi utilizado a comunicação serial para simular como seria a comunicação feita ao usuário, no caso esse mesmo projeto pode ser reproduzido e aplicado utilizando módulos de comunicação e transmissão utilizando internet que ser funcionamento seria totalmente válido. Tempo é algo valioso, e muitas vezes as pessoas que moram em condomínios ou apartamentos esquecem de retirar ou verificar suas cartas, e para não perder tempo indo verificar a caixa de correio, com a possibilidade de ter ou não alguma correspondência, esse projeto visa dar uma certa comodidade, auxiliando o controle de uma caixa de correio, por meio eletrônico, por isso o nome de Correio eletrônico foi escolhido para este projeto.

REFERÊNCIAS

[1] John H. Davies, MSP430 Microcontroller Basics, Elsevier, 2008.

[2] Software: fritzing (org).

Anexos

```
#include <msp430g2553.h>
#include <legacymsp430.h>
#include <stdio.h>
#define TX BIT2
int x=0;
int milisegundos;
int decsegundos=0;
int segundos=30;
int minutos=47;
int horas=15;

int distancia;
long sensor;
char txbuf[256];

void uart_send(int len) {
  int i;
  for(i = 0; i < len; i++) {
    while (UCA0STAT & UCBUSY);
    UCA0TXBUF = txbuf[i];
  }
}


int main(void)
{
  BCSCTL1 = CALBC1_1MHZ;
  DCOCTL = CALDCO_1MHZ;                    // clock 1mhz
  WDTCTL = WDTPW + WDTHOLD;                // para WDT

  CCTL0 = CCIE;                            // CCR0 interrupt habilitada
  TACCR0 = 1000;                             // 1ms
  TA0CTL = TASSEL_2 + MC_1;                // SMCLK, upmode
////HORAS////
  TA1CCR0= 6250; //  6250*2*8uS= 0,1s
  TA1CTL = TASSEL_2 + MC_3 + ID_3 +TAIE; // SMCLK  ,  UP/DOWN MODE, /8
//////////////
  P1IFG  = 0x00;                           //limpa flags de interrupção
  P1DIR |= 0x01;                           // P1.0 como saida pro LED
  P1OUT &= ~0x01;                          // apaga o led

  _BIS_SR(GIE);                            // global interrupt habilitada

 while(1){
       P1SEL2 = P1SEL = TX;
```

```
          P1IE &= ~0x01;                        // desabilita interupt
          P1DIR |= 0x02;                        // trigger como output
          P1OUT |= 0x02;                        // gera pulse
////////////////////////////////////////
      // UART: desabilitar paridade, transmitir um byte
      // na ordem LSB->MSB, um bit de stop
      UCA0CTL0 = 0;
      // UART: utilizar SMCLK
      UCA0CTL1 = UCSSEL_2;
      // Baud rate de 9600 bps, utilizando oversampling
      UCA0BR0 = 6;
      UCA0BR1 = 0;
      UCA0MCTL = UCBRF_8 + UCOS16;
////////////////////////////////////////
          __delay_cycles(10);           //  10us
          P1OUT &= ~0x02;               // para pulso
          P1DIR &= ~0x08;                       // torna o pin P1.3 entrada (ECHO)
      P1IFG = 0x00;                 // limpa flag se nada acontecer antes
          P1IE |= 0x08;                 // habilita interupção  pino ECHO
          P1IES &= ~0x08;                       // na borda de subida
      __delay_cycles(30000);            // delay por 30ms (after this time echo times out if there is no object detected)
      distancia = sensor/58;          // converte ECHO em cm

      while (x==0)  {  uart_send(sprintf(txbuf," %d mensagens\n",x) );x++;}

      if(distancia > 35 && distancia != 0){
      x=0;
       for(int j=0;j<150;j++) //aguarda 15 segundos depois q abriu a caixa
     {
     __delay_cycles(100000);
     }
       }


     if(distancia < 15 && distancia != 0)
     { P1OUT |= 0x01;  //liga o led indicando que chegou a carta

        if(x==1)        uart_send(sprintf(txbuf," %d mensagem recebida %.2d:%.2d:%.2d\n",x,horas, minutos,segundos));
        if(x==10) uart_send(sprintf(txbuf," Verificar correio\n"));
        if(x<10&&x!=1)  uart_send(sprintf(txbuf," %d mensagens recebida %.2d:%.2d:%.2d\n",x,horas, minutos,segundos));



     x++;
```

```
    for(int j=0;j<30;j++)
    {
    __delay_cycles(100000);
    }
     }

    else P1OUT &= ~0x01;




 }
}

#pragma vector=PORT1_VECTOR
__interrupt void Port_1(void)
{
          if(P1IFG&0x08)  //verifica se tem alguma interrupção pendente
       {
        if(!(P1IES&0x08)) // verifica se é borda de subida
        {
          TA0CTL|=TACLR;   // limpa o timer A0
          milisegundos = 0;
          P1IES |= 0x08;  //borda de descida
        }
        else
        {
          sensor = (long)milisegundos*1000 + (long)TAR;     //calcula distancia do ECHO

        }
          P1IFG &= ~0x08;                                   //limpa flag
          }
}

interrupt(TIMER0_A0_VECTOR) TA0_ISR(void)
{
  milisegundos++;

}

// INTERRUPÇÃO QUE CONTA A CADA 0,1 SEGUNDOS
interrupt(TIMER1_A1_VECTOR) TA1_ISR(void)
{

  decsegundos++;

    if(decsegundos>=10) {
```

```
    decsegundos=0;
    segundos++;}

         if(segundos>=60) {
           minutos++; // INCREMENTA OS MINUTOS
           segundos=0;
        }
         if(minutos>=60)  // INCREMENTA AS HORAS
         {
           minutos=0;
           horas++;
         }
         if(horas>=24)  //CASO UNIDADE =10 ZERA UNIDADE E INCREMENTA DEZENA
         {
           horas=0;

         }


  TA1CTL &= ~TAIFG;
}
```