Programação em BASIC para o PIC

Vitor Amadeu Souza

Introdução

Continuando com a série do último artigo, hoje veremos os passos para o desenvolvimento de uma minuteria com o microcontrolador PIC18F1220 e o mikroBASIC. Para isso, aprenderemos a utilizar as funções de retardo disponíveis na linguagem.

A função de uma minuteria é manter ligada uma saída durante um intervalo de tempo. Nos dias de hoje em que a demanda por energia elétrica é alta, ter formas de economizá-la é uma das aplicações em que um microcontrolador pode ser utilizado. Normalmente a minuteria é utilizada em corredores ou garagens por exemplo e que a ao se detectar a presença de uma pessoa, a iluminação é acionada por um período de tempo e logo em seguida ela é desligada automaticamente. Os sensores que informam ao circuito de controle a presença de uma pessoa são chamados de sensores de infravermelho PIR e estes podem ser observados na figura 1.

Figura 1 – Sensor de Infravermelho

Internamente destes sensores, já fica instalado o circuito de minuteria e o tempo em que o mesmo ficará acionado após receber o sinal de presença é ajustado através de um trimpot. Apesar deste projeto poder ser desenvolvido por um microcontrolador, nada impede também que utilizemos lógica discreta para o desenvolvimento deste circuito. Esta é mais uma aplicação onde o microcontrolador pode ser utilizado.

Recursos de Hardware

Para o desenvolvimento deste projeto, utilizaremos a placa didática PICLAB18F1220 desenvolvido pela Cerne Tecnologia (**www.cerne-tec.com.br**). Para simular o sensor de infravermelho, utilizaremos um botão e para simular a saída de um relé o led que fica instalado na própria placa. O circuito elétrico pode ser observado na figura 2.

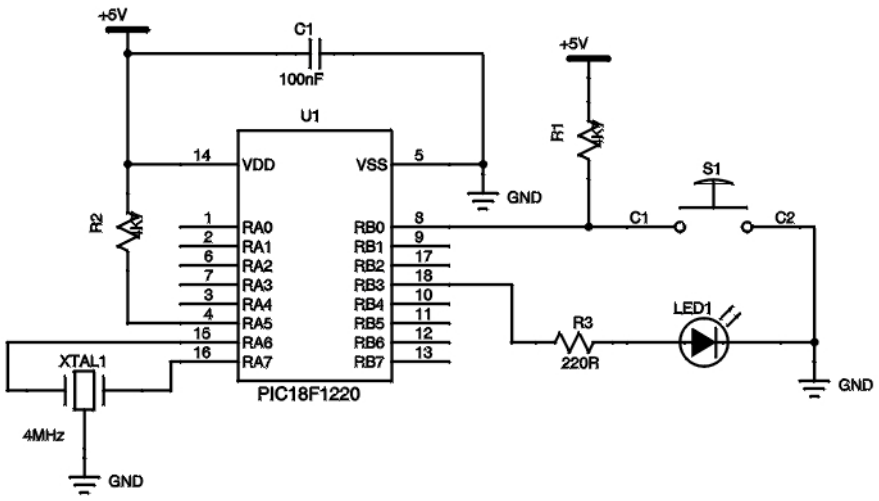


Figura 2 – Circuito Elétrico

Carta de Tempos

O exemplo funcionará da seguinte forma: Quando for detectado a presença de uma pessoa (ao pressionar o botão, no nosso caso) a saída ficará acionada durante o intervalo de 10 s. Este tempo pode ser alterado através de parâmetros de software como será visto em breve. Desta forma, a carta de tempo do nosso exemplo será como a apresentada na figura 3.

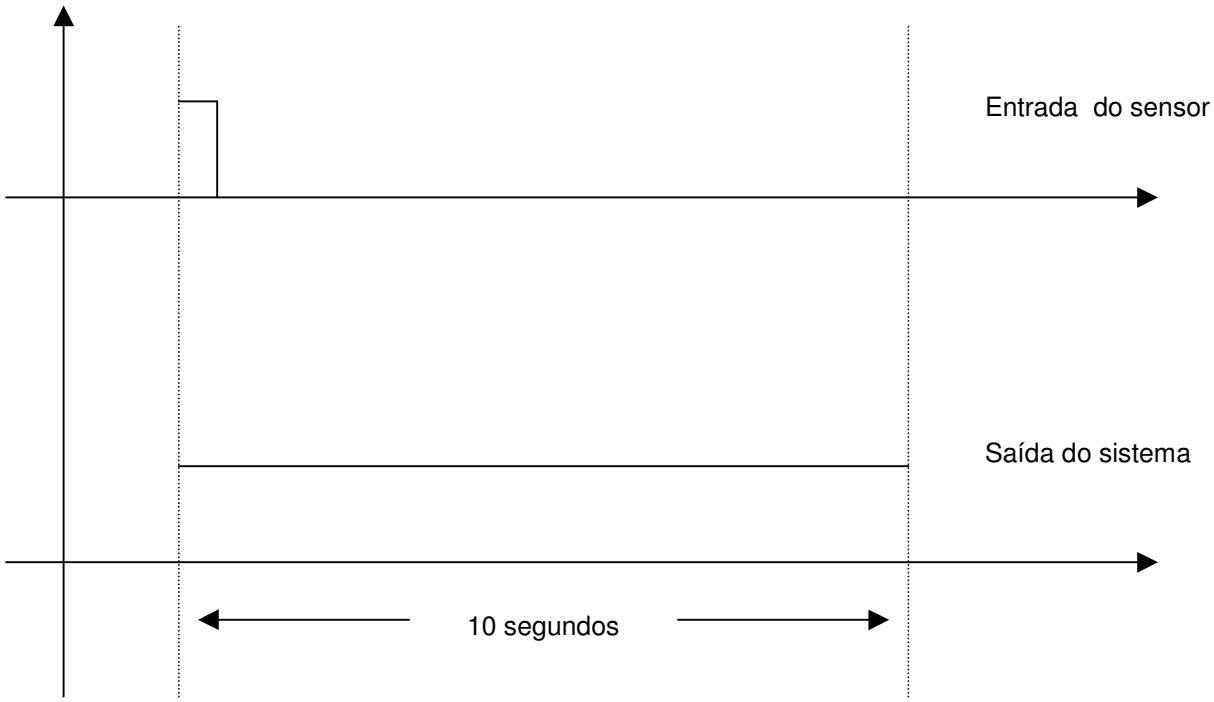
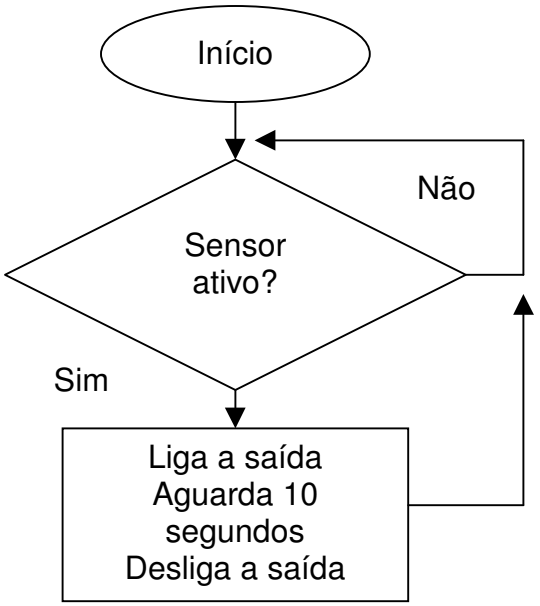


Figura 3 – Carta de Tempos

Fluxograma

O fluxograma que irá reger o funcionamento deste software pode ser apreciado na figura 4. Note que enquanto não é verificado a presença de nenhuma pessoa, o sistema fica preso em loop, testando ciclicamente a entrada do sensor. No entanto assim que este estado se altera, a saída é ligada e logo em seguida entra em uma rotina que faz com fique durante o intervalo de 10 s neste estado. Logo em seguida a mesma é desligada e volta-se a verificar a entrada do sensor.



Recursos de Software

Para execução deste exemplo, precisaremos de uma função que permita um retardo no programa. Para isso, utilizaremos a função da linguagem BASIC chamada *delay_ms(parâmetro).* Esta função faz com que o programa fique parado durante um intervalo de tempo em função de *parâmetro.* Desta forma, se chamarmos *delay_ms(1000)* o programa ficará durante o intervalo de 1 segundo "preso" nesta rotina causando desta forma o retardo necessário que precisamos. É importante salientar que o *parâmetro* máximo que pode ser passado para esta função é de 65535.

Software

O software final, que permitirá que construamos a nossa minuteria pode ser observado no box1. Observe que o leitor deve criar antes um projeto, de acordo com o primeiro artigo desta série e desta forma digitar o código abaixo.

```
program Minuteria

main:
  adcon1=%01111111      'Configura os pinos para funcionar em modo digital
  trisb=%00000001       'Configura o RB0 como entrada e restante como saída

      if portb.0 = 0 then   'Se o botão estiver pressionado...
            portb.3=1       'Liga a saída
            delay_ms(10000) 'Aguarda 10 segundos
            portb.3=0       'Desliga a saída
      end if

      goto main             'Salta para main

end.
```

Box 1 – Programa do Exemplo

Vamos agora analisar melhor o código exemplo. Primeiramente, o nome do programa é informado através de *program Minuteria.* Note que neste caso o projeto foi chamado de minuteria. Logo em seguida o programa começa através do label *main.* Na próxima linha encontramos a declaração *adcon1=%01111111.* Isto foi feito pois os pinos RA0, RA1, RA2, RA3, RB0, RB1e RB4 ficam configurados automaticamente como pinos de função analógica e como no exemplo proposto esta função não é utilizada, foi necessário desligá-la através deste comando. Mais detalhes sobre os pinos de função analógica serão explorados nos próximos artigos. O comando *trisb=%00000001* faz com que o pino RB0 fique configurado como entrada e o restante como saída (observe o esquema elétrico na figura 2). Após estas configurações, inicia-se o teste do pino RB0 e caso o mesmo fique ativo (neste caso em nível lógico 0) o led conectado na saída RB3 será acionado e ficará neste estado durante o intervalo de 10 s através da função *delay_ms(10000)*. Passado este intervalo, o led é desligado e o sistema volta a testar o sensor (botão) afim de detectar algum evento e acionar novamente a saída. Observe que ao lado de alguns comandos do programa, existe um texto explicativo ao lado que inicia-se por ‘ (apóstrofo). Todo o texto que é iniciado por este caracter é chamado de comentário e o compilador no momento da compilação não trata este texto, tornando o seu uso livre.

Após o desenvolvimento deste programa, compile o mesmo e transfira o arquivo hex para a placa didática afim de validar o exercício.

Conclusão

Os microcontroladores hoje são encontrados em diversos encapsulamentos e apresentam preços muito baixos. A Microchip lançou recentemente a família PIC10 que possui 6 pinos e é excelente para aplicações como esta, em que o processamento é baixo e o produto deve ter um baixo valor agregado.

Informo mais uma vez aos leitores que tiverem alguma dúvida, sugestão ou crítica que entre em contato comigo através do e-mail **vitor@cerne-tec.com.br**.