Projeto 1: Um semáforo

Criado pelo Prof. Tiago Fernandes Tavares, Março de 2017
Modificado pelo Prof. Antonio Quevedo, Fevereiro de 2018

A Avenida Roxo Moreira separa a Unicamp de um bairro cheio de restaurantes. Embora haja uma faixa de pedestres para travessia, os carros nem sempre param sob demanda. Como consequência, há risco de atropelamento nessa faixa. Pensando em diminuir esse risco, nossa equipe projetará uma prova de conceito para um semáforo que fecha para os carros sob demanda de pedestres que, porventura, estejam esperando para atravessar. Porém, é sabido que é perigoso que motoristas fiquem parados na rua durante a noite. Portanto, o semáforo também deve ser desativado, piscando amarelo para os motoristas e vermelho para os pedestres, durante todo o período da noite.

1 Objetivos

Ao fim deste projeto, o aluno deverá ser capaz de:
1. Projetar e montar o circuito para conectar LEDs ao microcontrolador,
2. Projetar e montar o circuito para conectar botões pushbutton ao microcontrolador,
3. Utilizar interrupções temporais e máquinas de estado para coordenar processos em tempo real.
4. Tomar decisões de projeto para resolver um problema real e simular os resultados dessa decisão.

2 Teoria

1. Ao projetar uma máquina de estados em *software* (uma máquina de Mealy ou de Moore, por exemplo), como é possível representar os estados da máquina? E as transições?
2. Como é um circuito que permita acender um LED? Qual é a queda de tensão em um LED vermelho? E num amarelo? E num verde? Qual é a corrente que passa pelos LEDs para que eles acendam?
3. Como é possível usar uma chave mecânica para aplicar tensões Vcc e terra em um ponto específico de um circuito? Qual é a função do resistor de *pull-up*, nesse caso?

2.1 Máquina de Estados

Uma máquina de estado é uma abstração, isto é, um modelo que pode ser usado para analisar determinados tipos de sistemas. Os sistemas aos quais esse modelo se aplica são aqueles que, a cada instante de tempo, assumem um, e apenas um, dentre *N* estados possíveis. Para cada estado, o sistema assume um comportamento diferente e, dependendo de suas entradas, transiciona para outro estado.
Poderíamos analisar o voo de um avião como uma máquina de estados. Inicialmente, ocorre o estado de embarque. Depois, há o estado de decolagem. Após, o estado de voo de cruzeiro, o pouso e, por fim, o desembarque. Perceba que o comportamento do avião é diferente em cada um desses estados. A transição entre os estados depende da aquisição de dados dos sensores de bordo (GPS, altímetro, velocímetro, etc.) e da comparação desses dados com um plano de voo.
Para implementar uma máquina de estados em *software*, precisamos de uma variável para representar (armazenar) o estado em que nos encontramos. Também, precisamos de uma função que implemente as regras de transição de estados. O exemplo mostrado abaixo implementa uma máquina de dois estados (0 e 1) que muda de estado em todos os acionamentos.

```c
int estado = 0; /* Estado atual da maquina
    */

void maq_estados() {
  /* Esta implementacao usa uma estrutura
      de if-then-else explicita */
  if (estado == 0) {
    estado = 1;
  }
  else {
    estado = 0;
  }
}
```

2.2 LED

O LED (*Light-Emitting Diode*) é um diodo que emite luz na passagem de corrente. Um possível modelo para seu funcionamento elétrico é:
1. Se a tensão aplicada no LED é inferior a um limiar v_l, a corrente que passa por ele é nula;
2. Caso contrário, a corrente que passa pelo LED é tão grande quando possível e observa-se uma tensão de v_l nos terminais do LED. A tensão v_l varia de acordo com a cor do LED e depende de características do material semicondutor do qual ele é feito.
Esse modelo significa que uma fonte de tensão conectada diretamente ao LED fornecerá teoricamente corrente infinita. Isso rapidamente levará o LED (e, possivelmente, a própria fonte de tensão) a queimar. Por esse motivo, é preciso conectar o LED em série a um resistor, como mostra a Figura 1.

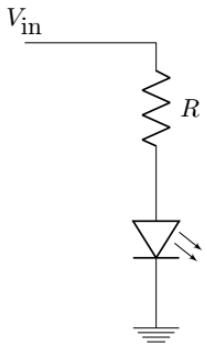

Figura 1: Circuito para acender um LED.

No circuito da Figura 1, observamos uma malha na qual passa uma corrente i. Assumindo que a tensão fornecida pela fonte é maior que o limiar de tensão para acender o LED ($V > v_l$), podemos verificar que a tensão no resistor é igual a (pela Lei de Kirchoff das Tensões) $V_r = V - V_l$. A Lei de Ohm nos permite inferir que a corrente na malha é igual a:

$$i = \frac{V - V_l}{R}.$$

Isso significa que é possível variar a corrente que passa pelo LED alterando a resistência do elemento colocado em série a ele. Uma corrente muito baixa levará a um brilho muito baixo, ao passo que uma
corrente muito alta poderá danificar os dispositivos conectados.

2.3 Chave mecânica

Um dispositivo *push button* é uma chave mecânica que é ativada quando pressionada. Ao ser ativada, a chave fecha o contato entre dois pontos. Assim, pode ser conectado como mostra a Figura

2.
No circuito, temos duas situações possíveis: a chave aberta e a chave fechada. Quando a chave está aberta, não passa corrente no circuito e, portanto, a tensão no resistor é zero. Quando a chave está fechada, o resistor é conectado à fonte, e, portanto, a tensão no dispositivo é *V* e a corrente no circuito é *V/R*.

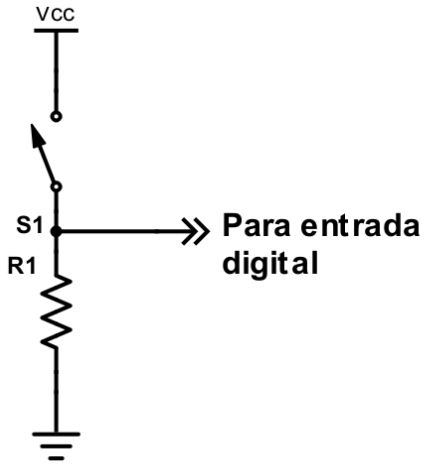

Figura 2: Circuito com uma chave.

Observe que, nessa estrutura, o resistor tem a função de induzir um estado-padrão no sistema, válido quando a chave está aberta. Com a chave aberta, sem o resistor, *Vout* seria um nó flutuante, e, portanto, sujeito a oscilações de tensão devidas ao ruído eletromagnético do ambiente.
Na estrutura da Figura 2, o resistor faz com que o estado-padrão da saída do circuito seja baixo (0*V*), e, portanto, é chamado de resistor de *pull down*. Se a chave e o resistor forem invertidos, o estado padrão do sistema passa a ser alto (*V*in), e, assim, o resistor passa a ser chamado de resistor de *pull up*.
Um problema comum nas chaves mecânicas é o “bounce”. Como o contato é mecânico, ocorre um “rebote” no momento do fechamento da chave, e ocorrem sucessivos contatos e desligamentos, gerando um trem de pulsos que dura algumas centenas de microssegundos. Uma maneira simples de resolver isto é colocando um capacitor em paralelo com a chave. Assim, no fechamento do contato a descarga ocorre instantaneamente, enquanto que a carga demora mais por conta do resistor em série. Assim, o sistema digital reconhece apenas o momento do primeiro fechamento do contato, e ao se desligar a chave, o momento do último rebote de abertura da mesma.

2.4 Sensor resistivo

Existem vários sensores simples que mudam sua resistência elétrica em função de uma variável ambiental (luz, calor, força, etc.). Se desejamos uma medida meramente qualitativa, podemos usar um divisor de tensão resistivo para gerar uma tensão proporcional (direta ou inversamente) à resistência do sensor e, assim, à variável sob medida. Para tanto, basta colocar o sensor em série com um resistor fixo, e ligar o conjunto a uma referência de tensão (por exemplo, a fonte de alimentação), medindo-se a tensão no ponto de junção do resistor fixo com o sensor (Figura 3).

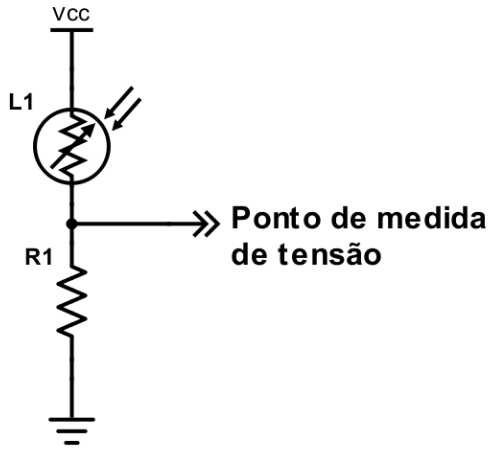

Figura 3: Divisor de tensão para sensor resistivo

Podemos fazer a medição da tensão de saída do circuito através de um conversor analógico-digital, que irá gerar um valor digital proporcional à tensão presente. No caso apresentado na figura 3, quando menor a resistência do sensor (L1), o que normalmente ocorre quando a variável ambiental aumenta seu valor, maior a tensão medida aumenta. Podemos inverter a posição do resistor fixo e do sensor, e assim inverter o comportamento do conjunto.

3 Sobre o problema

O grupo de trabalho deverá apresentar uma prova de conceito do sistema de semáforo implementado. Ele deverá funcionar da seguinte forma:

• Ao ser inicializado, está aberto para carros e fechado para pedestres;
• Quando um pedestre aperta o botão, o sistema:
1. Aguarda alguns segundos,
2. Mostra a luz amarela para os carros por algum tempo,
3. Fecha para os carros e, simultaneamente abre para os pedestres,
4. Mantém o estado por tempo suficiente para a travessia dos pedestres,
5. Pisca a luz vermelha de pedestre anunciando que irá fechar em breve,
5. Fecha para os pedestres e, simultaneamente, abre para os carros (isto é, retorna ao estado inicial).
6. À noite, o sistema de apertar o botão é desligado e o semáforo somente pisca amarelo para os carros e pisca vermelho para os pedestres, retornando ao estado original na manhã seguinte. Para esta detecção, o grupo deve utilizar um sensor tipo LDR para detectar a luz. O sistema deve ser capaz de ignorar mudanças temporárias na luz, tais como a passagem de um pássaro sobre o sensor durante o dia, ou a incidência de um farol de carro durante a noite. Ou seja, apenas alterações de luminosidade que perdurem por pelo menos alguns segundos podem mudar o modo de operação de diurno para noturno ou vice-versa.

Nesta montagem, pensando no objetivo de usar interrupções do temporizador, **não será permitido usar funções como *wait()*, *millis()* ou *delay()***. O motivo disso é que a programação baseada em interrupções será importante nas próximas instâncias desta disciplina. O CodeWarrior é capaz de gerar automaticamente código correspondente a uma interrupção periódica (componente *TimerInt*).