Trabalho Final

Controle para Automação (2020/2)

Fernando Cardoso Guimarães

Universidade de Brasília

Trabalho Final

- Simulação de problema de automação modelado como SED.
- Dividido em três partes:
  1. Modelagem do SED usando autômatos e implementação de código em C simulando a execução do sistema.
  2. Modelagem do SED usando redes de Petri e implementação de código em Matlab, C/C++ ou Python capaz de monitorar o sistema.
  3. Implementação de código em Ladder considerando a execução do sistema por um CLP.
- O trabalho é individual.
- Entrega do relatório em PDF e dos códigos-fonte: 19/05/21 até a 00h00 via Classroom.

Trabalho Final - Avaliação

- A nota final do Trabalho, N_T, de acordo com o plano de ensino será dada por

$$N_T = \frac{T_1 + T_2 + T_3}{3} \ ,$$

onde:

- $T_1 \in [0; 10]$: Nota da parte 1.
- $T_2 \in [0; 10]$: Nota da parte 2.
- $T_3 \in [0; 10]$: Nota da parte 3.

Trabalho Final – Regras

- Os detalhes a respeito do que deverá ser feito são dados no fim desta apresentação na forma de 6 especificações.
- As especificações serão sorteadas pelo professor entre os alunos, não estando sujeitas à escolha por cada um.
- O relatório deverá ser direto e sucinto. Deverá possuir apenas três tópicos, onde em cada um será feita a descrição da modelagem e dos códigos implementados em cada parte do trabalho.
- Os códigos-fonte e o PDF do relatório deverão ser enviados numa pasta compactada .rar ou .zip via Classroom, na qual deverá estar presente um arquivo LEIAME.txt com instruções de compilação e execução.

Parte 1 – Modelagem por autômatos e código em C

- Nesta parte do trabalho é preciso modelar o sistema usando autômatos. Os eventos serão associados a dois tipos de ocorrência:
  - Intervenção do operador por meio das botoeiras, chaves, etc.
  - Leitura de sensores.

- Os estados do SED devem estar associados aos estados dos equipamentos como: lâmpada ligada, motor desligado, etc.

- A implementação em C parte do pressuposto que o código rodará em um microcontrolador responsável pela execução das tarefas especificadas a partir da leitura das entradas do usuário e dos valores dos sensores.

- O código criado deverá usar como base aquele fornecido pelo professor.

Parte 2 – Monitoramento por redes de Petri

- Nesta parte do trabalho é preciso modelar o sistema usando redes de Petri. A interpretação dos eventos é a mesma que para os autômatos.

- Os estados da rede também deverão estar associados aos estados dos equipamentos.

- A implementação do código em Matlab, Python ou C/C++, deverá empregar as equações de estado para simular o comportamento do sistema, tendo como entrada sequências de disparos de transições definidas no modelo.

Parte 2 – Monitoramento por redes de Petri

- Especificamente, o código deverá:

  - Ter uma função ou método para fornecer todas as sequências possíveis de disparos de transições até um limite de N disparos (argumento da função).
  - Ter uma função que cheque se determinada transição está habilitada em certo estado.
  - Ter uma função que retorne o estado da rede de Petri recebendo como argumento de entrada uma sequência de disparos de transições. Essa função deverá usar a equação de estado e checar a habilitação de cada transição da sequência, retornando uma mensagem de erro se alguma não estiver habilitada (sequência inválida).

Parte 3 – Linguagem Ladder para CLPs

- Nesta parte do trabalho, usando como orientação os modelos criados nas outras partes, deverá ser criado um código na linguagem Ladder para CLPs, supondo-se que esse tipo de equipamento será usado para se atender aos requisitos do problema.

- Será preciso fornecer um diagrama com as conexões das chaves, boteiras, sensores e equipamentos nas saídas e entradas do CLP fictício.

- As aplicações recomendadas para a simulação são o LDMicro e PLC Fiddle (online):

  https://cq.cx/ladder-pt.html; https://www.plcfiddle.com/

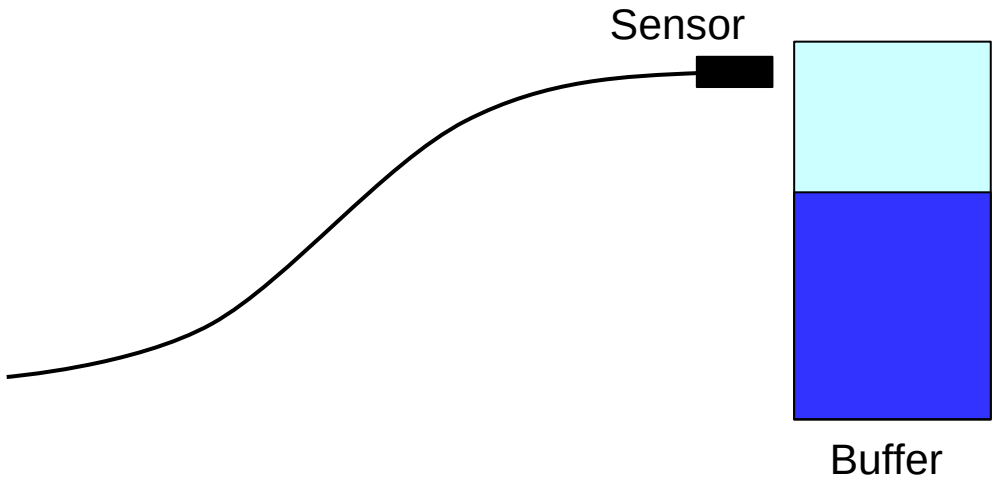
- Outros programas podem ser adotados, mas com a anuência do professor. Quem optar pelo PLC Fiddle deverá fornecer o *link* quando da entrega.

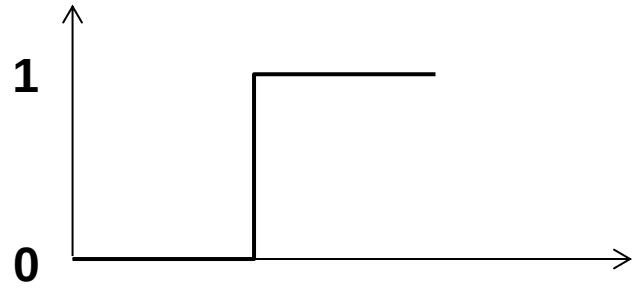
Trabalho Final - Especificações

- Especificação 1: simulação de *buffer* cheio + especificações comuns.
- Especificação 2: simulação de buffer vazio + esp. comuns.
- Especificação 3: simulação de contador de peças + esp. comuns.
- Especificação 4: simulação de partida temporizada de motor + esp. comuns.
- Especificação 5: simulação de sistema de arrefecimento por limiar de temperatura + esp. comuns.
- Especificação 6: simulação de sistema de iluminação automática + esp. comuns.

Trabalho Final - Especificações

- Especificação 1: simulação de *buffer* cheio + especificações comuns.
  - Simular um sistema que detecta se um buffer está cheio.
  - Usar um sensor de proximidade que, caso seja ativo (saída 1), desliga um motor. Se o buffer for esvaziado, o motor não liga automaticamente, mas deve ser ligado por uma botoeira NA.
  - O sistema também deverá contar com dois sinalizadores: quando o buffer não está cheio, um deles fica permanentemente ligado; quando o buffer está cheio, o primeiro desliga enquanto o outro começa a piscar a 2Hz. Se o buffer esvazia, o primeiro volta a ligar e o segundo desliga, automaticamente.

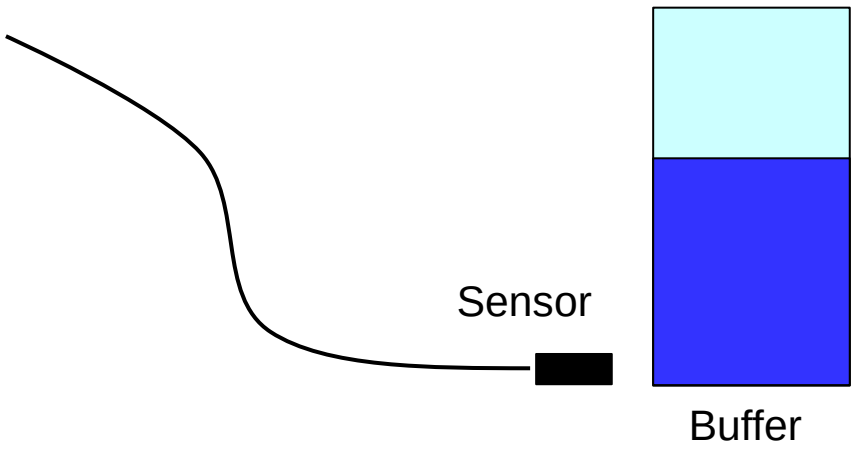


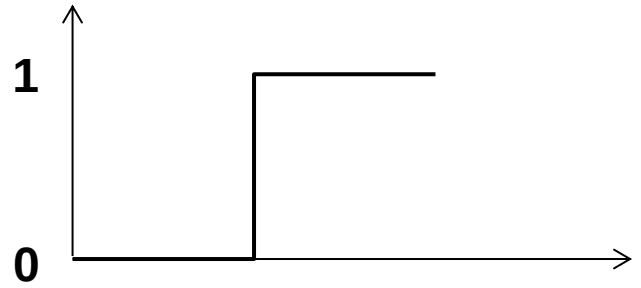


$$\text{saída do sensor} = \begin{cases} 0, & \text{presença não detectada} \\ 1, & \text{presença detectada} \end{cases}$$

Trabalho Final - Especificações

- Especificação 2: simulação de *buffer* vazio + especificações comuns.
  - Simular um sistema que detecta se um buffer está vazio.
  - Usar um sensor de proximidade que, caso fique inativo (saída 0), liga um motor. Se o buffer começar a encher, o motor continua ligado, podendo ser desligado apenas por uma botoeira NF.
  - O sistema também deverá contar com dois sinalizadores: quando o buffer não está vazio, um deles fica permanentemente ligado; quando o buffer está vazio, o primeiro desliga enquanto o outro começa a piscar a 2Hz. Se o buffer começa a encher, o primeiro volta a ligar e o segundo desliga, automaticamente.

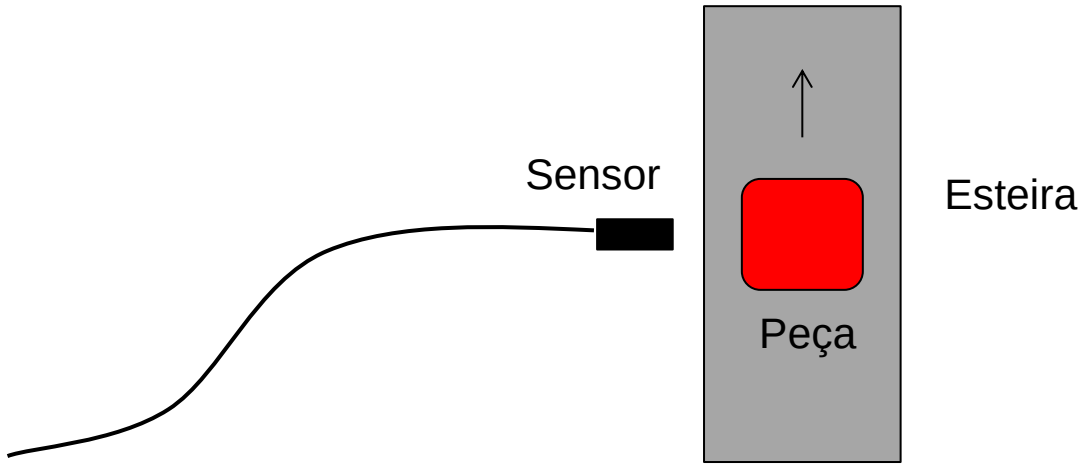


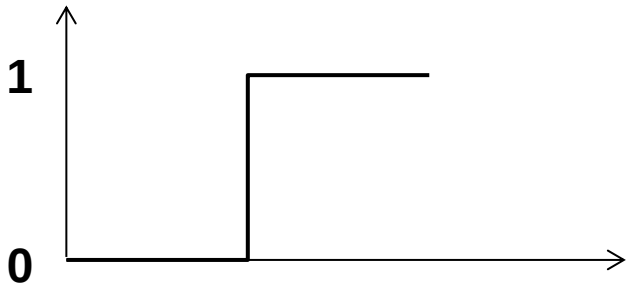


$$\text{saída do sensor} = \begin{cases} 0, & \text{presença não detectada} \\ 1, & \text{presença detectada} \end{cases}$$

Trabalho Final - Especificações

- Especificação 3: simulação de contador de peças + especificações comuns.
  - Simular um sistema que conta a quantidade de peças que passa por uma esteira numa linha de produção (deve-se usar um contador da linguagem Ladder e não um contador externo em *hardware*).
  - Usar um sensor de proximidade que, caso saia da condição inativo (saída 0) para ativo (saída 1), indica que uma peça passou pela esteira.
  - O sistema também deverá contar com dois sinalizadores: quando o número de peças é menor que 5, um deles fica ligado permanentemente enquanto o outro permanece desligado. Se o número de peças é igual a 5, o primeiro desliga e o segundo pisca a 2Hz.
  - O sistema volta ao estado inicial por meio do acionamento de uma botoeira NA.

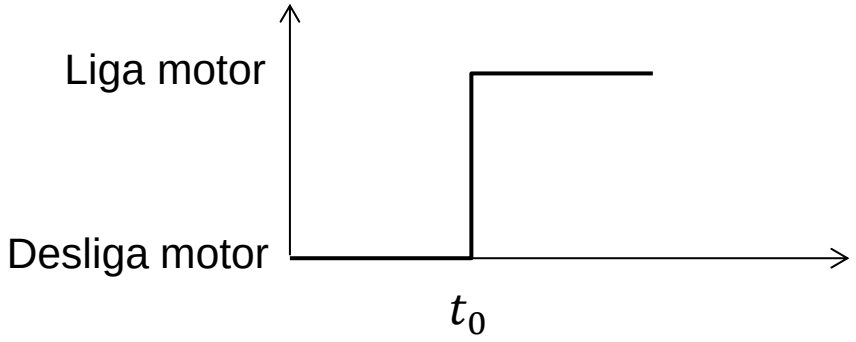


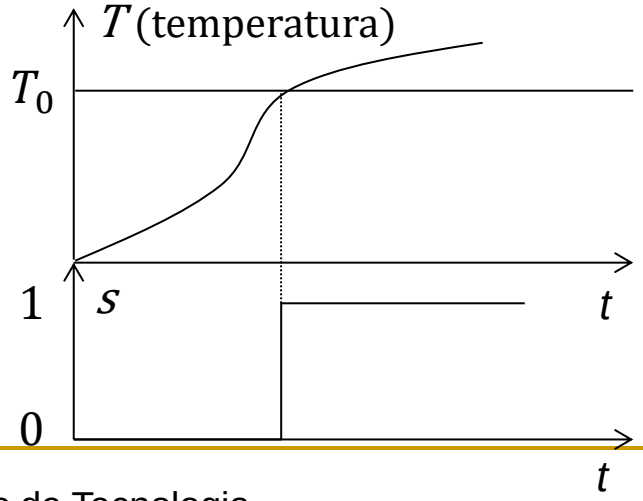


$$\text{saída do sensor} = \begin{cases} 0, & \text{presença não detectada} \\ 1, & \text{presença detectada} \end{cases}$$

Trabalho Final - Especificações

- Especificação 4: simulação de partida temporizada de motor + especificações comuns
  - Simular um sistema que aciona um motor a partir de um intervalo de $t_0 = 5$ segundos contados a partir do acionamento de uma botoeira NA. O motor poderá ser desligado por uma botoeira NF.
  - O sistema também deverá contar com um sensor de temperatura, que caso indique que a mesma ultrapassa certo limiar T_0, desliga o motor. Se o limiar de temperatura for cruzado com o motor desligado, este não poderá ser ligado, mesmo com o acionamento da botoeira NA. Se a temperatura voltar a cair abaixo de T_0, o sistema volta ao estado inicial com o motor desligado.
  - Deverão ser usados, ainda, dois sinalizadores: um ficará permanentemente ligado se a temperatura estiver abaixo de T_0, enquanto o outro ficará desligado. Se a temperatura estiver acima do limiar, o primeiro desliga e o outro pisca a 2Hz.

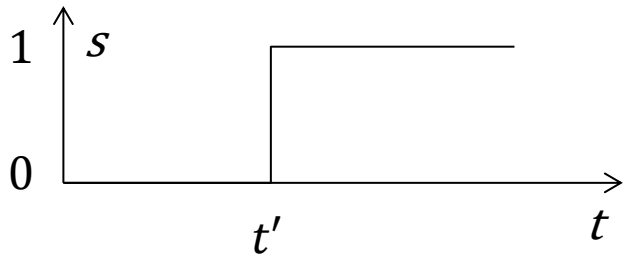




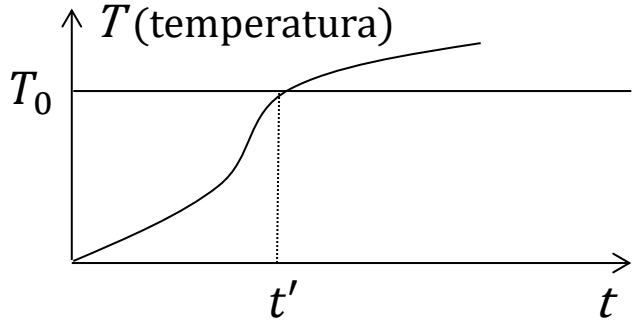
$$s = \begin{cases} 0, & T \leq T_0 \\ 1, & T > T_0 \end{cases}$$

Trabalho Final - Especificações

- Especificação 5: simulação de sistema de arrefecimento por limiar de temperatura + esp. comuns.
  - Simular um sistema que aciona uma ventoinha assim que um sensor de temperatura detecta o cruzamento de um limiar de temperatura T_0.
  - Se a temperatura voltar a cair abaixo do limiar T_0, a ventoinha desliga automaticamente.
  - Deverão ser usados, ainda, dois sinalizadores: um ficará permanentemente ligado se a temperatura estiver abaixo de T_0, enquanto o outro ficará desligado. Se a temperatura estiver acima do limiar, o primeiro desliga e o outro pisca a 2Hz.



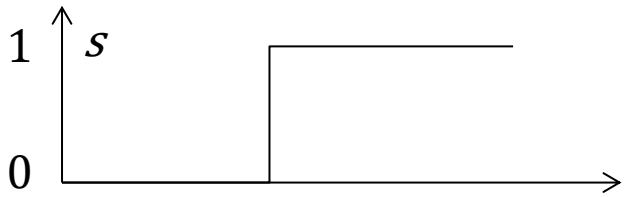
$$s = \begin{cases} 0, & T \leq T_0 \\ 1, & T > T_0 \end{cases}$$



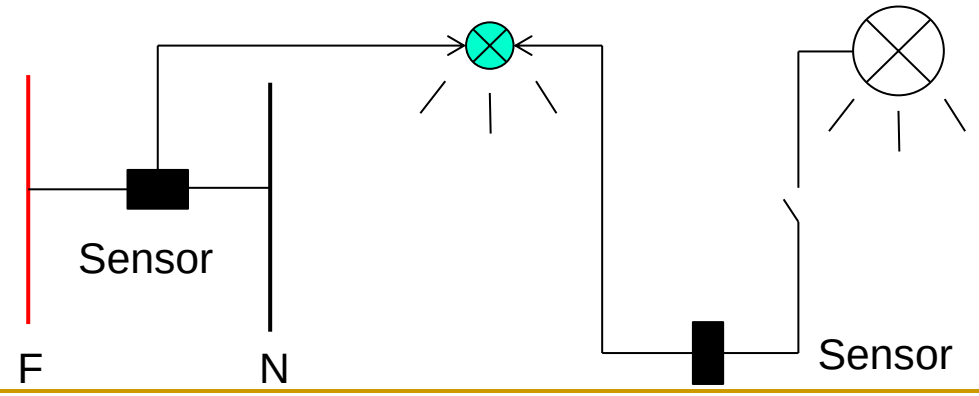
Trabalho Final - Especificações

- Especificação 6: e simulação de sistema de iluminação automática + esp. comuns

  - Simular um sistema que aciona um conjunto de lâmpadas principais se a luminosidade do ambiente externo cai abaixo de determinado patamar. Se ela volta a subir acima do patamar adotado, as lâmpadas desligam.
  - As lâmpadas poderão ser desligadas por uma chave de 2 posições. Se a chave estiver na posição OFF, as lâmpadas desligam; se estiver na posição ON, as lâmpadas ligam apenas de acordo com o sensor de luminosidade.
  - O sistema também contará com um detector de falta de energia que aciona um conjunto de lâmpadas de emergência caso deixe de detectar um certo nível de tensão na rede e se a luminosidade ambiente estiver abaixo do limiar estabelecido. Se a luminosidade ambiente estiver acima do limiar, nada acontece. Caso haja falta de energia, as lâmpadas principais não podem acender e um sinalizador também deverá ser posto para piscar a 2Hz



$$\text{saída do sensor} = \begin{cases} 0, & \text{luminosidade baixa} \\ 1, & \text{luminosidade normal} \end{cases}$$



Trabalho Final – Especificações comuns

- Além das definições particulares, as especificações de 1 a 6 deverão contar com alguns requisitos em comum.
- Em todos os casos, o sistema só começará a funcionar de acordo com as especificações dadas a partir do acionamento de uma botoeira geral NA, independentemente da previsão de outras botoeiras nos casos particulares.
- O sistema será desativado por uma botoeira geral NF. Para simplificar, pode-se escolher um estado específico no qual seria possível desligar o sistema por meio dessa botoeira.
- O estado ativo ou inativo do sistema deverá ser indicado por um sinalizador. Assim, se a botoeira geral NA é acionada, o sinalizador liga; se a botoeira NF geral é pressionada, ele desliga.

Trabalho Final – Especificações comuns

- Todos os sistemas deverão contar com um motor acionado por uma botoeira NA. O motor poderá ser desligado por uma botoeira NF.

- Para as Especificações 1, 2, 4 e 5, que já contam com motores, não será preciso usar um motor adicional. Contudo, as botoeiras acima deverão estar presentes, caso já não estejam previstas.

- Cada sistema deverá contar com uma chave comutadora de 2 posições que determinará o sentido de rotação do motor.

- O sentido de rotação do motor deverá ser indicado por dois sinalizadores: quando o motor gira no sentido horário, um fica ligado e o outro fica desligado. Se o sentido é invertido, os estados dos sinalizadores são invertidos.

Trabalho Final – Especificações comuns

- Por fim, todos os sistemas deverão contar com uma botoeira de emergência que, caso seja acionada, desativa todos os dispositivos (motores e sinalizadores) em qualquer estado em que o sistema se encontre.

- O acionamento do botão de emergência deverá ligar um sinalizador, o único elemento que poderá permanecer ligado.

- A botoeira de emergência é NF e mantém seu estado quando acionada, ela pode ser representada por uma chave de 2 posições.

- Quando a botoeira de emergência volta ao seu estado normal (NF), o sinalizador de emergência é desligado e o sistema como um todo pode retornar a qualquer estado que se queira.

- Com modelagem por redes de Petri, a comutação da botoeira de emergência (de aberta para fechada e vice-versa) poderá ser modelada por várias transições rotuladas pelos mesmos eventos, desde que isso não cause não determinismos.

Bom trabalho!