APLICAÇÃO DE UM SISTEMA DE CONTROLE AUTÔNOMO DISTRIBUÍDO (SCAD) EM UM KIT DIDÁTICO PARA ENSINO DE PROGRAMAÇÃO

Pedro Carvalhaes Dias [1]; Maurício Silveira [2] ; David Bianchini [3]
Pontifícia Universidade Católica Campinas, Engenharia Elétrica
Rodovia D. Pedro I, km 136, Parque das Universidades
CEP : 13086-900 , Campinas, SP
pedro@carval.com.br

[2] Pontifícia Universidade Católica Campinas, Engenharia Elétrica
Rodovia D. Pedro I, km 136, Parque das Universidades
CEP : 13086-900 , Campinas, SP
msilveira@puc-campinas.edu.br

[3] Pontifícia Universidade Católica Campinas, Engenharia Elétrica
Rodovia D. Pedro I, km 136, Parque das Universidades
CEP : 13086-900 , Campinas, SP
davidesperantista@gmail.com

Resumo: *Neste trabalho é apresentada uma sugestão de aplicação de um sistema de controle autônomo distribuído desenvolvido (o SCAD) na área de ensino, como um kit didático direcionado ao ensino de programação. Ele serve para aulas básicas no ensino de programação e também como uma ferramenta auxiliar no ensino de protocolos de comunicação. O objetivo principal da proposta é estimular os alunos sinestéticos, proporcionando um ambiente de ensino mais visual e até mesmo fisicamente interativo.*

Palavras-chave: *Kit Didático, Programação, Controle Distribuído, Cinestética, Ensino Prático.*

1. INTRODUÇÃO

Existem basicamente 3 tipos de alunos no que diz respeito à facilidade de aprendizado: o *auditivo*, o *visual* e o *cinestético* (CASAU,2008). O *auditivo* é aquele que aprende melhor ouvindo o professor, participando de debates. O *visual* é o que aprende melhor através de estímulos visuais, sejam estes slides ou anotações. Já o *cinestético* é aquele que aprende mais através de interações mecânicas com o objeto de estudo, sendo que em estudos recentes, foi verificado que 36% dos alunos são cinestéticos (TEIXEIRA,2008), sendo, portanto, muito importantes as atividades dirigidas a este tipo de aluno. Normalmente, esse tipo de aluno tem melhor desempenho em aulas de laboratório, onde tem maior interação física com o tópico estudado.

O *kit didático* apresentado neste trabalho funciona como uma ponte para o aluno *cinestético*. A programação ainda basicamente visual, mesmo sendo um tópico que desperta ambos os lados do cérebro; tanto o racional quanto o criativo. Com o *kit didático* baseado no SCAD, ela pode "saltar" para fora do visual, e entrar no mundo físico, servindo como estímulo extra ao aluno *cinestético*.

O SCAD (DIAS e SILVEIRA, 2007) é um sistema baseado em microcontroladores, que serve como “ponte de comunicação” entre mais de um sistema. Ele é composto por três grandes blocos, sendo eles o Mestre, o SCAD e seus Slaves. Como padrão, o Mestre é um programa em um computador, que se comunica através de uma porta RS232 com o SCAD. O SCAD recebe as instruções do Mestre e repassa o comando, via I^2C, ao Slave correspondente. Os Slaves recebem o dado do SCAD, e realizam sua função, seja realizando um processo próprio, ou retornando um dado ao SCAD e/ou Mestre.

Essa configuração faz com que o sistema seja muito maleável quanto ao método de ensino escolhido, em diversos tópicos do ensino de programação. Pode se utilizar o sistema para ensinar programação de inteligência artificial, automação e controle, protocolos de comunicação, e outras disciplinas associadas à área. O sistema pode ser adaptado para diversas linguagens de programação, desde que se tenha disponível um compilador da linguagem escolhida para o microcontrolador.

2. FUNCIONAMENTO DO SCAD

O SCAD, como mencionado anteriormente, é composto por três grandes blocos, sendo eles o Mestre, o SCAD em si, e os Slaves. Cada um deles será descrito de forma mais detalhada nos tópicos seguintes.

2.1 Descrição do Mestre

Sugere-se que o mestre seja um programa em um computador, utilizando uma porta RS232 para comunicação. O programa pode, por exemplo, receber dados do usuário, e transmiti-los aos Slaves. Como padrão, esta comunicação com o SCAD é dada de maneira simples, com 2 bytes por transmissão, sendo que o primeiro é o endereço do Slave de destino, e o outro a informação com a qual o Slave trabalhará. Alguns bytes de “endereço de Slave” são reservados nesse protocolo, de forma que o Mestre possa fazer pedidos diretamente ao SCAD como, por exemplo, leitura de dados armazenados em todos Slaves.

2.2 Descrição do SCAD

O SCAD é composto basicamente por um único Microcontrolador, e suas entradas e saídas de comunicação. Como padrão, a porta de comunicação com o Mestre é uma RS-232, de forma a facilitar o uso de um computador como Mestre, e a comunicação com os Slaves através do protocolo I^2C, tornando possível conectar diversos Slaves simultaneamente na mesma porta do SCAD. O SCAD, como proposto, recebe 2 bytes do Mestre, sendo um a informação e o outro o endereço do Slave destino, e analisa se a informação é de leitura ou de escrita. Se a informação for de escrita, o SCAD simplesmente envia a informação pelo BUS do I^2C para o Slave correspondente. Se for de leitura, envia a informação ao Slave, espera pela resposta, e repassa a resposta para o Mestre.

Alguns comandos específicos podem ser associados ao SCAD, como “Leitura do dado X de todos os slaves”. Isso faria com que o SCAD enviasse para todos os slaves a instrução X de leitura, e repassasse ao mestre em uma única transmissão.

2.3 Descrição dos Slaves

Os slaves, também são compostos por microcontroladores e, como padrão, falam com o SCAD através de uma porta de comunicação I^2C. Dessa forma, é possível que vários Slaves

se comuniquem com o SCAD por uma mesma linha de comunicação. Sugere-se, dependendo da proposta do professor, que os Slaves realizem tarefas relativamente simples, como acender ou apagar LEDs, retornar informações de sensores, ou até mesmo controlar servo mecanismos, baseados na informação recebida pelo SCAD. Por exemplo, um Slave que controle 8 LEDs, pode receber um Byte de 8 bits, por exemplo 00100100b, onde cada bit represente o estado de um LED, e acenda os leds correspondentes, isto é, no nosso exemplo, o terceiro e o sexto LED, de maneira similar a como se trabalharia com uma porta paralela. Ou, em outro exemplo para um Slave, o Mestre pode transmitir a direção e o número de passos que um motor de passo deve dar para o Slave, sendo o bit mais significativo a direção, e os 7 bits restantes o número de passos. Ou ainda outro exemplo de Slave, este pode retornar a informação de um sensor de temperatura ao SCAD, lendo um sinal analógico, e transformando este em um byte que será retransmitido ao SCAD.

2.4 Descrição dos processos e rotinas do SCAD

A Figura 1 mostra um diagrama de blocos simplificado do funcionamento do SCAD, um pouco mais detalhado que a descrição apresentada no tópico **2.2**. Nos tópicos seguintes comentaremos brevemente o que cada um destes blocos representa.

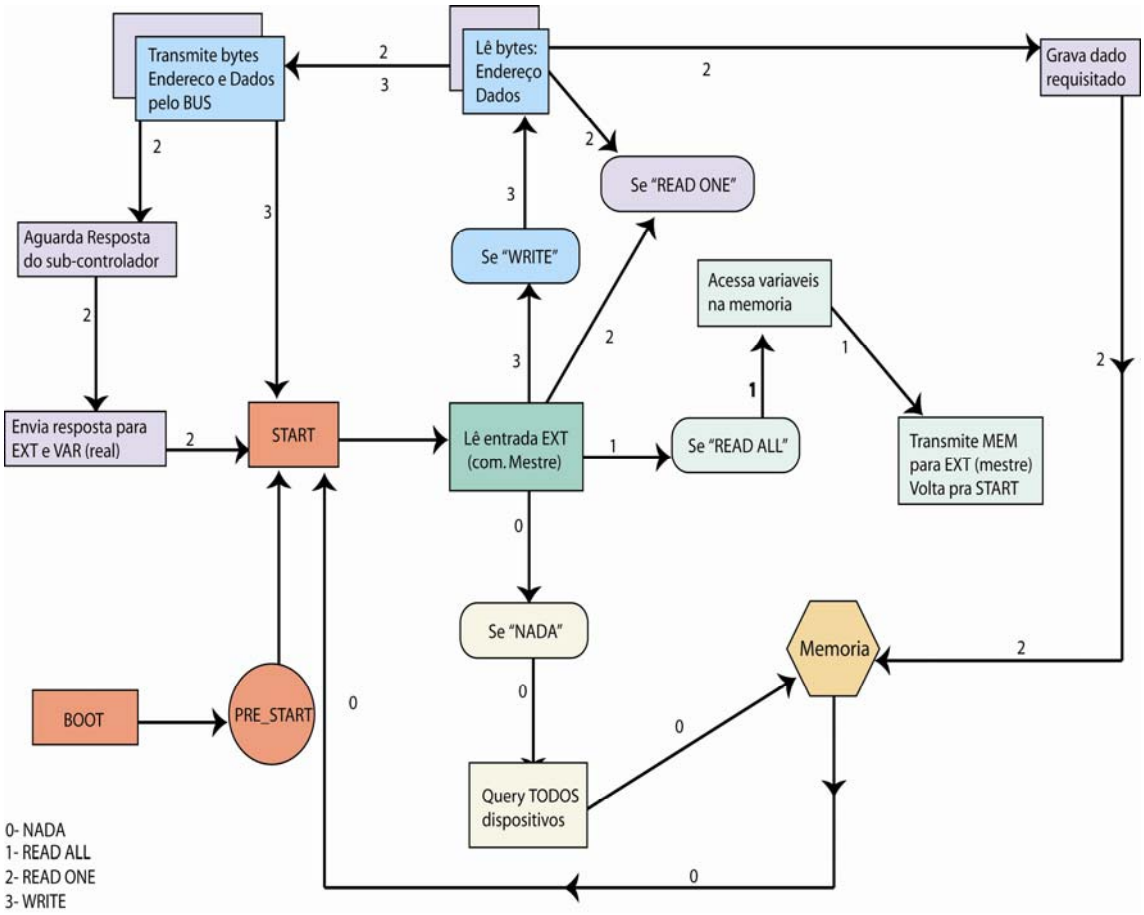

Figura 1 – Diagrama de Blocos do Funcionamento do SCAD

2.4.1 Caso o Mestre não mande nenhum comando

O mestre pode fazer diversas requisições ao SCAD, no tempo que desejar. Mas se o mestre não toma nenhuma decisão, e o SCAD fica "*inativo*", o procedimento padrão do

SCAD é de realizar uma varredura em massa, coletando os dados de resposta de todos os Slaves, e armazenando na memória interna do próprio SCAD, como a posição ***real*** de cada Slave.

Essa varredura pode ser limitada por dois controles, um deles chamado ***query_limiter*** e o outro ***always_query***. O ***query_limiter*** é uma variável que armazena um valor numérico, faz com que o SCAD aguarde "***query_limiter***" ausências de chamadas do Mestre antes de realizar a varredura completa. Já o ***always_query*** é uma variável booleana, que apenas impede ou permite que o SCAD realize essa varredura completa em estado *idle*.

Dessa forma, podemos fazer com que o SCAD realize menos varreduras por período de tempo, deixando as linhas de comunicação desobstruídas por mais tempo, ou até mesmo não realize a varredura, sempre esperando um comando direto de leitura.

2.4.2 Caso o Mestre envie um comando de *leitura única*

O mestre pode enviar um comando ao SCAD de leitura do valor de um único Slave. Quando isso ocorre, a transmissão do mestre é composta por 2 bytes de 8 bits, como tinha sido mencionado anteriormente. O primeiro byte é o do endereço do Slave, e a segunda, o comando de leitura do mesmo. O SCAD então repassa o comando de leitura do Slave, e aguarda a resposta. Quando o Slave responde via I^2C, o mestre guarda imediatamente em sua memória interna o valor como posição ***real*** do Slave e repassa esse valor ao mestre, assim como sua posição ***desejada***.

2.4.3 Caso o Mestre envie um comando de *ler todos*

O mestre pode mandar um comando de ***ler todos*** ao SCAD. Esse comando pode ter dois resultados diferentes, dependo de ***always_query***. Se o SCAD estiver configurado para fazer varreduras completas periodicamente (***always_query*** ligado), o SCAD apenas acessa sua memória, e transmite ao Mestre os valores armazenados na última varredura. Já se as varreduras periódicas estiverem desativadas, o mestre realiza uma varredura forçada, e só então repassa os valores ***reais*** e ***desejados*** para o mestre.

Dessa forma, com as varreduras periódicas ligadas, a resposta a um comando ***ler todos*** é mais rápida, mas a linha de comunicação I^2C fica ocupada por mais tempo, enquanto com as varreduras desligadas, o Mestre se comunica com os Slaves **apenas** quando diretamente requerido.

2.4.4 Caso o Mestre envie um comando de *escrita*

O mestre também pode enviar um comando de ***escrita*** ao SCAD, pelo mesmo protocolo de 2 bytes por transmissão, sendo o primeiro o endereço do Slave, e o segundo o comando a ser transmitido. Imediatamente, o SCAD guarda o comando a ser transmitido como posição ***desejada*** do Slave, e então transmite ao Slave via I^2C.

2.4.5 Posição *real* e *desejada* do Slave

Como apresentado, o SCAD armazena duas informações de posição para cada Slave. A posição ***real*** e a ***desejada***. Isso é muito útil para situações onde a condição que o Slave se encontra não é necessariamente a condição que ele deveria estar. Podemos ver isso, por exemplo, se estivermos controlando a velocidade de um motor. O Mestre pode repassar um comando de ***escrita*** para o Slave, para manter o motor de uma roda girando a 200 *rpm*. Mas, por algum imprevisto mecânico, como uma obstrução na roda o Slave não consegue funcionar

como deveria, e gira o motor a apenas 85 *rpm*. Quando o Mestre enviar um comando de ***leitura*** Slave, receberá a informação que o Slave deveria estar operando a 200 *rpm* que representa a posição ***desejada***, mas está operando a 85 *rpm* indicado a posição ***real***.

Dessa forma, o programa Mestre pode ter algoritmos de correção para imprevistos e correções adequadas para cada tipo de problema (dependendo do slave).

Também pode se usar esse sistema de posições ***real*** e ***desejada*** para se criar Slaves mistos de sensores e controle. O SCAD pode receber informações de um sensor, por exemplo, de temperatura como posição ***real*** do Slave, e enviar comandos para que o Slave ligue ou desligue um ventilador, de forma a fazer um sistema de controle de temperatura, controlado remotamente através do SCAD, aproveitando o sistema de posições ***real*** e ***desejada***.

3 EXEMPLOS DE EXPERIMENTOS PARA USO EM ENSINO

Serão apresentadas neste tópico algumas sugestões de experimentos de laboratório que utilizem o SCAD na área de programação.

3.1 Desenvolvimento de Inteligência Artificial Básica

Um sistema composto pelo SCAD e diversos Slaves, montados em uma pequena plataforma robótica de rodas, é dado pelo professor. Os Slaves são 3. Um deles recebe um byte de 8 bits, sendo os 4 bits mais significativos a direção e velocidade em que a roda esquerda da plataforma gira, e os 4 menos significativos a direção e velocidade da roda direita. Os outros funcionam como sensores, sendo eles um sonar e dois sensores de toque, um à direita e o outro à esquerda da plataforma. O aluno já recebe um programa semi-acabado (Mestre), capaz de se comunicar com o SCAD e seus respectivos Slaves, com as rotinas de controle das rodas, e de leituras dos sensores.

O aluno deve desenvolver, então, um núcleo de programa para o Mestre, que consiga movimentar a plataforma autonomamente, evitando obstáculos de maneira eficiente, a partir dos dados retornados pelas funções de chamadas de sensores dadas pelo professor.

Observe que, como as rotinas de como mover a plataforma robótica e de como recolher os dados dos sensores já estão prontas, o único trabalho do aluno será de desenvolver e programar o sistema que toma as decisões de direção e reação, baseado nos dados.

Isso faz com que um experimento simples, de tomada de decisões, possa tomar novas dimensões, com o aluno observando fisicamente o resultado de seu código, andando e correndo pelo chão, podendo ser um estímulo extra ao aluno mais *cinestético*.

3.2 Controle de um Motor de Passo

O Professor disponibiliza o Mestre, o SCAD, e um Slave incompleto. O Mestre serve para controlar um motor de passo conectado ao I^2C do SCAD. O aluno recebe um Slave parcial, com uma rotina já pronta que recebe o Byte e o disponibiliza para o aluno. Por exemplo, a forma de controle citada na seção 2.3, onde o bit mais significativo é a direção do passo, e os 7 menos significativos constituem o número de passos, podendo serem utilizados.

O aluno deve desenvolver então um programa que, utilizando essa instrução do Mestre através do SCAD, movimente o motor da forma correspondente à enviada pelo Mestre. Desta forma, nais uma vez, o aluno mais *cinestético* poderá ser beneficiado.

3.3 Encapsulamento de Protocolos de Comunicação

O professor disponibiliza o SCAD, um Mestre incompleto e um Slave incompleto. O mestre disponibiliza palavras de 16 bits, e o Slave processa essas palavras e realiza alguma função simples, como escrever palavras num display LCD, ou movimentar um motor de passo. Mas o protocolo de comunicação disponível entre o Mestre e o SCAD (RS232) e entre o SCAD e o Slave (I^2C) está pronto para palavras de 8 bits.

Cabe ao aluno desenvolver um mini-protocolo próprio, de forma a transmitir essas palavras de 16 bits através do protocolo de 8 bits, e recuperar essas palavras no Slave, reconvertendo-as em palavras de 16 bits.

4 CONSIDERAÇÕES FINAIS

Neste trabalho foi apresentado o SCAD, uma ferramenta muito maleável, que pode ser programado praticamente em qualquer linguagem de programação e pode ser usada para uma infinidade de fins, desde princípios básicos na programação a tópicos mais avançados na mesma.

De posse do código fonte do sistema em mãos que são: os códigos padrões do Mestre, do SCAD e de um Slave, que são fornecidos com o kit; o professor pode adaptá-lo como melhor lhe aprouver, de forma a utilizá-lo como ferramenta didática.

O programa original do SCAD foi desenvolvido na linguagem de programação C, especificamente para um microcontrolador PIC16F877A. Este microcontrolador tem alguns recursos úteis para o desenvolvimento da plataforma SCAD, visto que algumas rotinas, como de comunicação RS232 e I^2C já estão embutidas no mesmo. O código é facilmente portado para outro microcontrolador, se for de interesse do professor, e o desenvolvimento das rotinas de comunicação pode até mesmo virar uma proposta para aula.

Acredita-se que o uso do SCAD em aulas práticas possa fazer com que alunos mais cinestéticos possam se interessar mais pelas disciplinas relacionadas com programação, devido à ponte que o SCAD fornece entre as linhas de código e o mundo real.

Agradecimentos

Agradecemos à PUC-Campinas pelo suporte financeiro relativo à Bolsa de Iniciação Científica que tornou possível a realização deste trabalho.

REFERÊNCIAS BIBLIOGRÁFICAS

DIAS, P. C.; SILVEIRA, M. SCAD – Sistema de controle autônomo distribuído. In: XII MOSTRA DE INICIAÇÃO CIENTÍFICA – PUC CAMPINAS, Outubro de 2007, Campinas, SP.

CASAU, P. Estilos de Aprendizaje: El modelo VAK. In: http://spu.autoupdate.com/ler.php?modulo=8&texto=363, Consultado em 3 de Junho de 2008.

TEIXEIRA, G. O Estilo de aprendizagem Individual. In:http://www.serprofessoruniversitario.pro.br/ler.php?modulo=8&texto=447, Consultado em 2 de Junho de 2008.

AN AUTONOMOUS DIGITAL CONTROL DISTRIBUTIONS SYSTEM AS A LEARNING TOOL FOR LABORATORY PROGRAMMING CLASSES

Abstract: *An autonomous digital control distribution system which can be used as a learning tool for programming classes is presented. It is useful not only to teach standard programming courses, but also as a communication protocols teaching aid. The main objectives of the proposal is to stimulate the kinaesthetic students, providing a more visual and physically interactive environment during the classes.*

Key-words: *Learning Tool, Programming, Distributed Control, Kinaesthetic, Practical Classes.*