2007 COBENGE
XXXV Congresso Brasileiro de Educação em Engenharia

UM LABORATÓRIO VIRTUAL PARA APLICAÇÕES EM ENSINO DE AUTOMAÇÃO E CONTROLE

Italo Seschini – iseschini@yahoo.com.br
José Maria Galvez – jmgalvez@ufmg.br
Universidade Federal de Minas Gerais, Departamento de Engenharia Mecânica
Av. Antônio Carlos, 6627
CEP 31.270-901- Belo Horizonte – Minas Gerais

Resumo: *Este trabalho apresenta o projeto e implementação de uma bancada de teste portável, altamente versátil e de baixo custo para a análise e experimentação de sistemas de automação. Plantas virtuais foram implementadas em computador utilizando o ambiente Windows(R) utilizando a linguagem Visual Basic. As plantas virtuais permitem emular plantas reais em todos os seus detalhes de operação. Desta forma, elas podem ser utilizadas em todas as etapas envolvidas num projeto experimental de automação e controle. Qualquer tipo de controlador externo, um CLP por exemplo, poderá ser utilizado para implementar e testar algoritmos de controle. As ações de controle são transmitidas à planta virtual através da porta paralela do computador. O sistema permite a fácil implementação de funções de controle básicas e avançadas tais como: aquisição de dados, supervisão de variáveis, temporização de eventos, lógica de relés, controladores PID, etc.*

Palavras-chave: *Laboratório Virtual, Ensino de Automação, Educação em Engenharia.*

1 INTRODUÇÃO

Com o recente e continuo desenvolvimento tecnológico mundial, as instituição de ensino de um modo geral encontram-se na difícil situação de tentar acompanhar a rápida evolução tecnológica contando apenas com escassos recursos financeiros para laboratórios e infra estrutura. É freqüente encontrar instituições de renome com laboratórios obsoletos devido aos altíssimos custos necessário para manter atualizados os seus laboratórios de ensino. Neste contexto, diversas tentativas tem sido realizadas para suprir esta demanda de mercado a custos compatíveis com os recursos econômicos existentes.

Recentemente, laboratórios virtuais tem se tornado populares no ensino de ciências e tecnologia básicas (física, química, eletrônica, etc.), devido principalmente à revolução da informática, com os preços cada vez mais baixos e com o aumento da capacidade computacional dos computadores de pequeno porte.

No contexto do ensino de automação e controle pode-se encontrar no mercado diversos equipamentos, nos quais a planta é uma simplificação de um processo real e onde, em geral, o controlador é "emulado" num computador pessoal, nestes casos uma interface computador/planta (em geral constituída por uma placa de aquisição de dados com conversores A/D e D/A) permite a interação entre o controlador emulado e a planta. Na configuração clássica, uma mudança do processo (uma nova planta) exige um novo investimento para a compra do novo equipamento. Por outro lado, o usuário não tem a oportunidade de lidar diretamente com um controlador real existente no mercado.

O enfoque apresentado no presente trabalho segue uma filosofia diferente, neste caso, a planta é implementada virtualmente num computador pessoal, utilizando VBasic da Microsoft(R) como linguagem de programação.

A comunicação entre a planta virtual e o mundo real e realizada através de uma interface que utiliza a porta paralela do computador como canal de comunicação. Esta interface é totalmente transparente para o usuário do sistema. Ela é utilizada para implementar fisicamente as entradas e as saídas da planta virtual que são apresentadas ao usuário final na forma de um painel de terminais.

O painel de terminais permite a execução de diversas tarefas de supervisão e controle. Entre elas: a aquisição de dados de temperatura, de pressão, do histórico de eventos, etc. Permite também o controle em malha fechada do processo virtual através de um controlador externo. Os terminais de entrada e saída operam com correntes e tensão padronizadas, o que permite um interfaceamento simples com qualquer equipamento externo.

A principal vantagem desta configuração é que o usuário tem a oportunidade de interagir diretamente com o sistema de controle real (externo) e com uma planta virtual que se apresenta através de um painel de terminais ("conectados" aos seus sensores virtuais) exatamente como uma planta real.

Outra característica importante da configuração proposta é que ela facilita a mudança do tipo de planta a ser controlada (pela simples troca do software); permitindo, desta forma, apresentar novos desafios ao aluno através da mesma montagem experimental.

Neste trabalho, uma simples máquina ferramenta é apresentada como exemplo de aplicação, entretanto, outras plantas estão disponíveis para utilização imediata.

2 A INTERFACE PC / CLP

Para realizar a comunicação entre o CLP e a porta paralela (LPT) do PC foi projetada e montada uma interface que recebe os sinais de ambos equipamentos.

A porta paralela padrão possui 12 pinos de saída (PC → Mundo), e apenas 5 pinos de entrada (PC ← Mundo). Com isso ficamos limitados a receber apenas 5 sinais de controle para a planta virtual o que seria uma característica limitadora para a implementação de plantas mais complexas. Uma solução, nesses casos, seria necessário a utilização de uma segunda porta paralela. Uma outra forma de resolver este problema é utilizar um sistema de multiplexação de sinais.

Neste trabalho um sistema de multiplexação foi implementado utilizando 4 multiplexadores de 8 entradas, o que corresponde a um total de 32 entradas. As saídas (multiplexadas) dos 4 multiplexadores são então conectadas às 4 entradas da LPT (restando ainda 1 para uso futuro).

Para controlar os multiplexadores são necessários 3 bit's de controle ($2^3 = 8$). Com este objetivo foram utiliados 3 pinos de saída da LPT do registrador de controle (neste caso os pinos 1, 14 e 16). Os bit's de controle são atualizados por um contador módulo 8 (0 a 7) incluído no próprio programa da planta virtual.

Nos 9 pinos restantes foram utilizados como caminho para os sinais dos sensores e alguns comandos específicos da planta. Finalmente, cabe observar que alguns bits do registrador de controle trabalham com lógica negativa o que deve ser levado em consideração no projeto das interfaces. As Figuras 1 e 2 apresentam os esquemas do fluxo simplificado da entrada e saída de dados do PC.

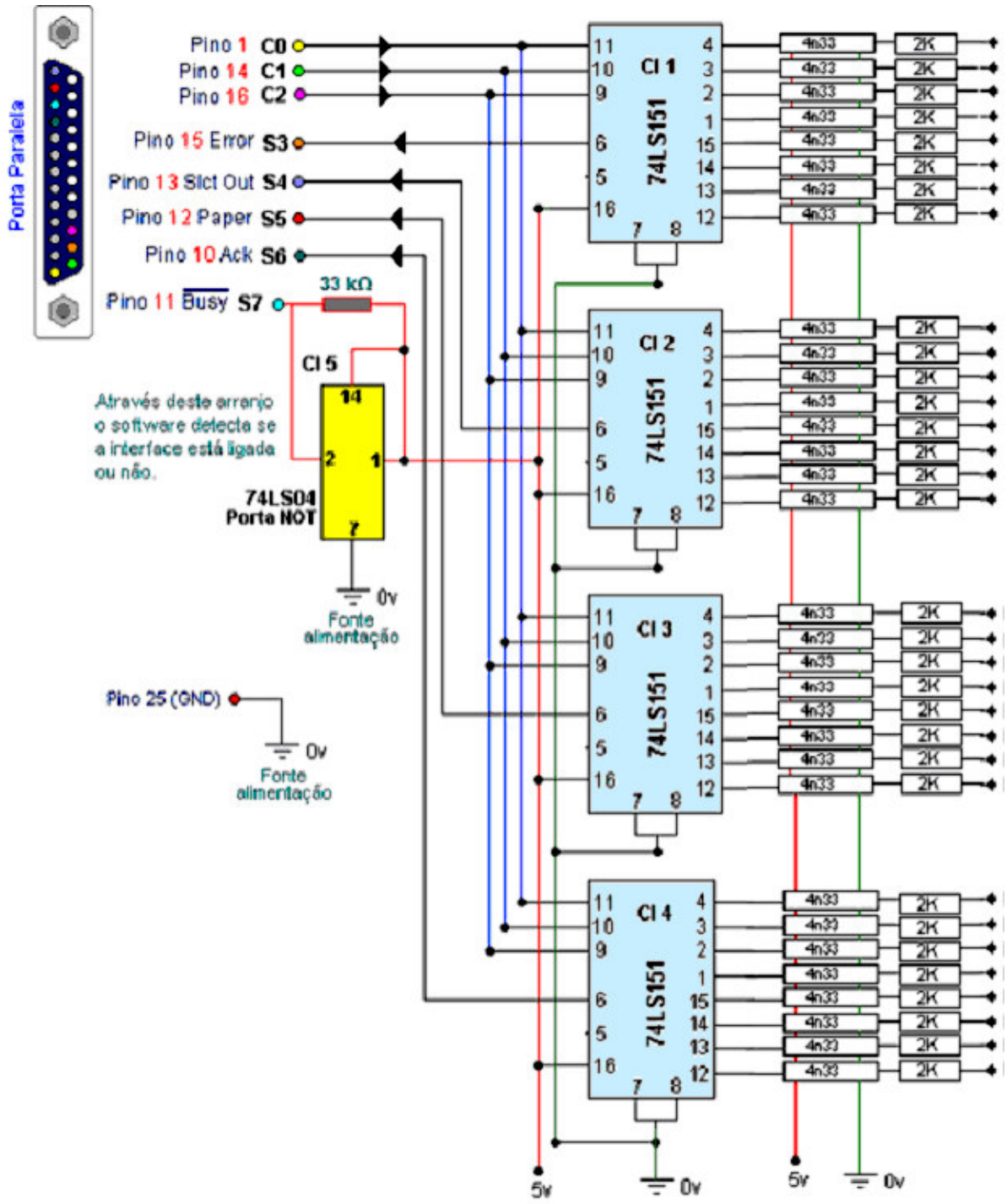

Figura 1. Interface para Entrada de Dados no PC

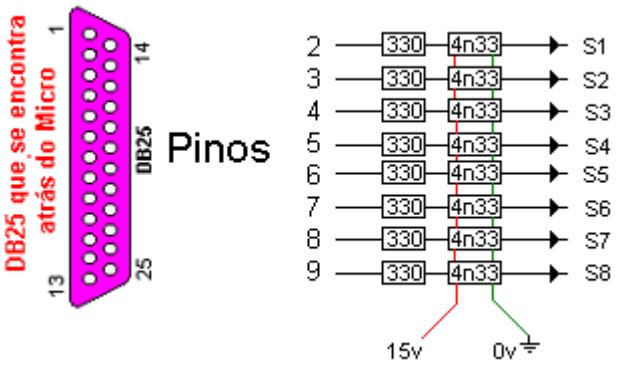

Figura 2. Interface para Saída de Dados do PC.

3 ACESSANDO A PORTA PARALELA

Para acessar a porta paralela (LPT) é necessária à utilização de uma biblioteca dll, a inpout32.dll, disponível na Internet no link: http://www.logix4u.net/inpout32.htm. Para rodar programas que utilizam esta biblioteca é preciso colocar a mesma na mesma pasta do programa. A maioria do material pesquisado informa que para WinXP o procedimento acima seria suficiente para que a porta fosse liberada, mas na prática verificou-se que o WinXP ainda bloqueava o acesso. O problema foi resolvido utilizando programa e o procedimento disponível no link: http://www.rogercom.com/pparalela/DriverNT_2000_XP.zip. No Visual Basic a biblioteca é declarada como mostrado abaixo:

```
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAddress As Integer) As Integer
Private Declare Sub Out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
```

3.1 Funcionamento

- **Endereços das Portas**

O computador nomeia as portas paralelas, chamando-as de LPT1, LPT2, LPT3 etc, entretanto a porta física padrão de computador é a LPT1, e seus endereços são: 378h (para enviar um byte de dados pela porta), 378+1h (para receber um valor através da porta) e, 378+2h (para enviar dados). Às vezes pode está disponível a LPT2, neste caso seus endereços são: 278h, 278+1h e 278+2h, com as mesmas funções dos endereços da porta LPT1 respectivamente. Nos programas desenvolvidos foram utilizados valores decimais de endereço, Figura 3.

Nome da Porta	Endereço de memória	Endereço da Porta		Descrição
LPT1	0000:0408	378 hexadecimal	888 decimal	Endereço base
LPT2	0000:040A	278 hexadecimal	632 decimal	Endereço base

Figura 3. Endereços das Portas Paralelas Utilizadas.

- **Os Registradores**

Utilizando a porta paralela conectada a uma impressora, os endereços tem nomes sugestivos, como mostrado na Figura 4.

Nome	Endereços LPT1	Endereços LPT2	Descrição
Registro de Dados	888	632	Envia um byte para a impressora
Registro de Status	889	633	Ler o Status da impressora
Registro de Controle	890	634	Envia dados de controle para a impressora

Figura 4. Registros Utilizados

- **O Conector DB25**

O DB25 é um conector que fica na parte de trás do gabinete do computador, e é através deste, que o cabo paralelo se conecta ao computador para poder enviar e receber dados. No DB25, um pino está em nível lógico 0 quando a tensão elétrica no mesmo está entre 0 à 0,4V. Um pino se encontra em nível lógico 1 quando a tensão elétrica no mesmo está acima de 3.1 e até 5V. A Figura 5 mostra o conector padrão DB25, com 25 pinos, onde cada pino tem um nome que o identifica:

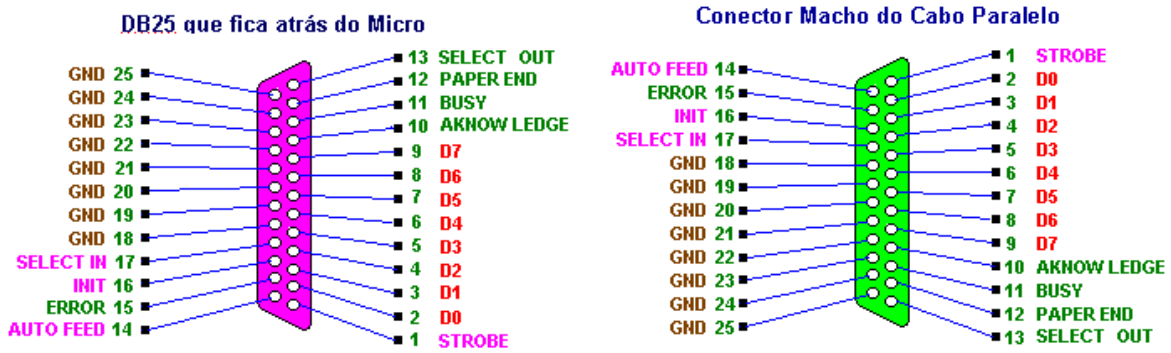

Figura 5. Pinagem do Conector DB25.

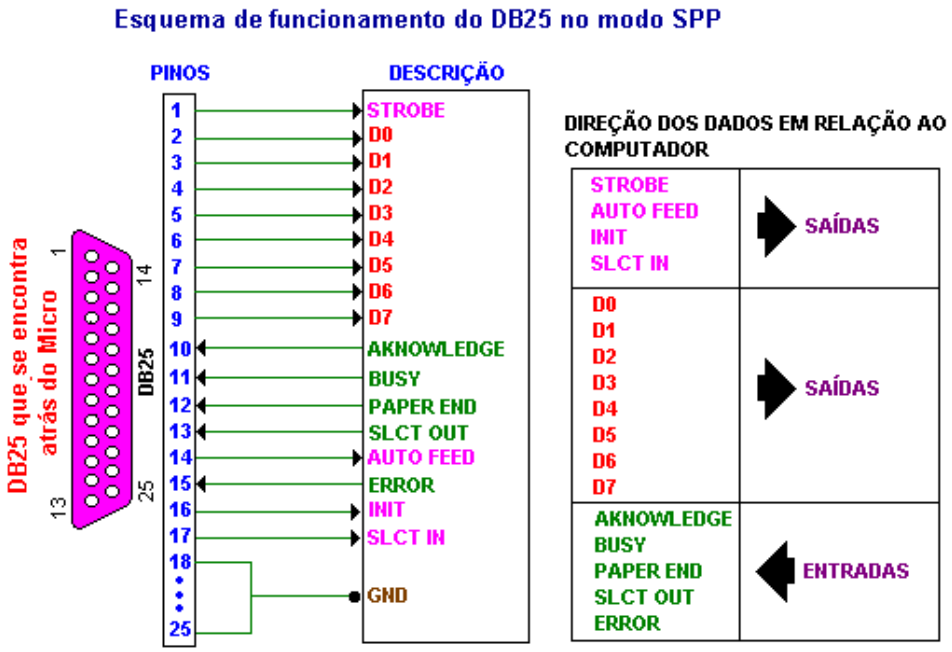

Figura 6. O Conector DB25 no Modo SPP.

3.2 Exemplo e Teste :

Foi desenvolvido um exemplo simples para testar o envio e recebimento de sinais através da porta paralela. Neste aplicativo, se pode escrever valores arbitrários na porta paralela, assim como ler os valores contidos na porta paralela. Vale lembrar que os valores mostrados estão no formato decimal (convertido do valor binário). Os valores do registrador de controle não conferem com os digitados pelo usuário devido ao fato de que apenas os bits menos significativos estão disponíveis e que alguns bits trabalham com lógica negativa (invertidos). Esse fato deve ser levado em consideração durante a implementação de um projeto que for utilizar esse recurso. A Figura 7 mostra um exemplo de aplicativo

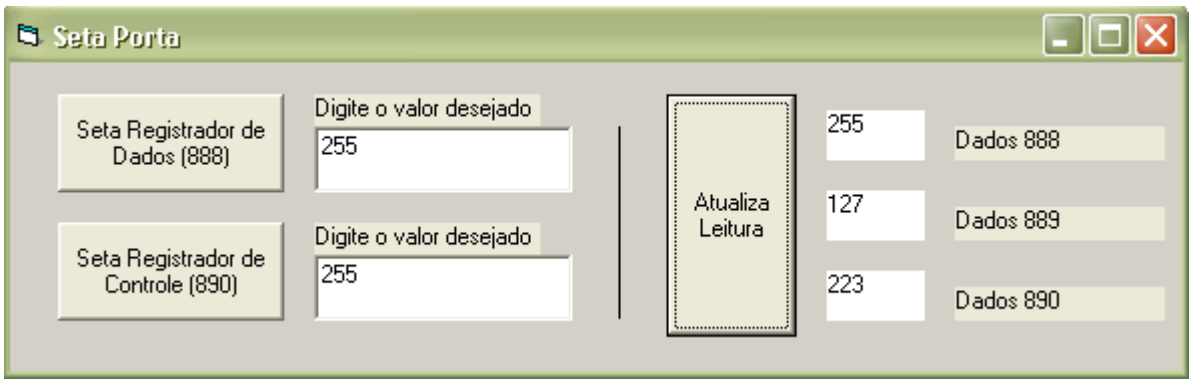

Figura 7. Exemplo do Aplicativo Implementado.

A Figura 8 apresenta o código fonte do aplicativo mostrado acima.

```
'/// Declaração de uso da inpout32.dll
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAddress As Integer) As Integer
Private Declare Sub Out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
'/// Ação do botão Seta Registrador de Dados (888)
Private Sub Command1_Click()
    Out 888, Text1.Text
End Sub
'/// Ação do botão Seta Registrador de Dados (890)
Private Sub Command2_Click()
    Out 890, Text2.Text
End Sub
'/// Ação do botão Atualiza Leitura
Private Sub Command3_Click()
    Label6.Caption = Inp(888)
    Label7.Caption = Inp(889)
    Label8.Caption = Inp(890)
End Sub
```

Figura 8. Código Fonte para Gerar o Aplicativo Exemplo da Figura 7.

4 A PLANTA VIRTUAL

O projeto da planta virtual aqui apresentado e um sistema simples que consiste numa furadeira automática com furo simples. Nessa aplicação é possível explorar alguns conceitos básicos como reconhecimento de sensores, sinais de emergência e atuação de motores. Entretanto, o objetivo principal foi a sua utilização como teste da interface eletrônica entre o PC e o CLP.

4.1 Especificações do Sistema

- Uma Esteira levará a peça não furada para a Mesa de Furação. A Esteira é tracionada por um motor que será acionado pelo CLP;
- O Motor-Esteira deverá ser acionado quando um sensor de presença (Sensor-Esteira) posicionado no início da esteira indicar que existe uma peça aguardando. Porem a motor não deverá ser ligado caso o sistema esteja desligado ou em estado de emergência;
- O Sensor-Esteira é ativado quando há uma peça sobre a esteira. E desativado quando a peça chega à Mesa de Furação;
- Temos também uma Furadeira que é acionada por um motor para descer a furadeira (Motor-Desce) e para subir (Motor-Sobe);
- O Motor-Desce é deve ser acionado quando um sensor de presença (Sensor-Mesa), que percebe se a peça chegou à mesa, está ativado. Mas não deve operar se o sistema estiver desligado ou em emergência;
- O Motor-Sobe é acionado logo após a peça ser furada. Para isso existi um sinal (Furou) que a planta envia para o CLP. Esse motor deve obedecer à prioridade dos comandos de Desligar e emergência;
- Para ligar o sistema, basta clicar em Ligar, o mesmo serve para desligar o sistema;

- Para se colocar uma peça não furada na esteira, basta clicar em “Colocar-Peça”, repare que enquanto há uma peça na esteira ou na mesa não é possível colocar outra peça na esteira;
- Para se retirar uma peça furada basta clicar em “Retirar-Peça”, repara que só é possível se retirar à peça quando a mesma já estiver furada e a furadeira em sua posição correta;
- O comando emergência quando acionado envia um sinal para o CLP que deve desligar o sistema imediatamente;

A planta virtual é mostrada na Figura 9.

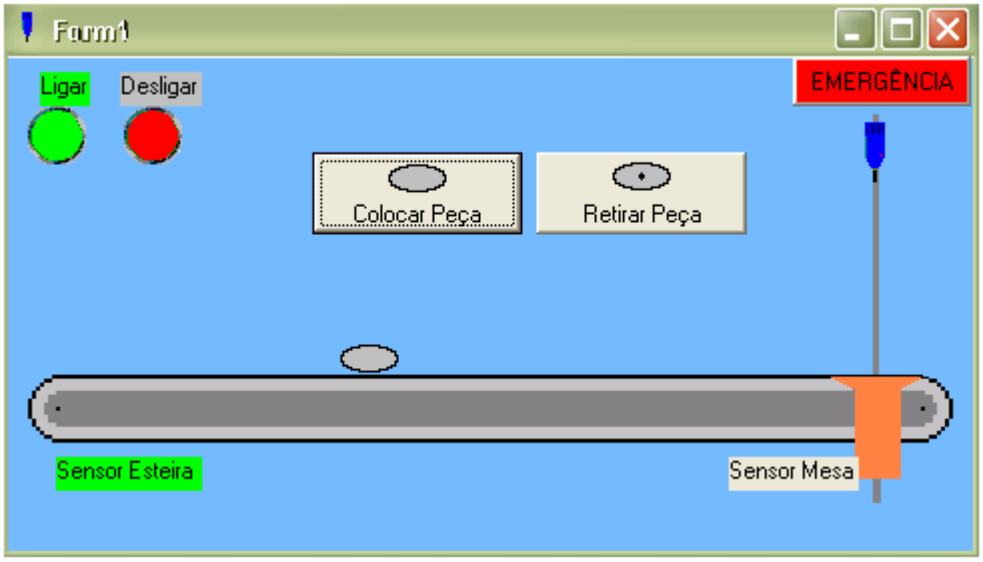

Figura 9. A Planta Virtual Utilizada como Exemplo.

4.2 O Código Fonte

O código fonte da planta virtual juntamente com as rotinas de entrada e saída de dados é apresentado a seguir:

```
'// Autor: Italo Devens Seschini
'// Laboratório de Automação e Controle DEMEC-UFMG
'// Data: 12/04/2007

'****** Entradas e Saidas*************'
Dim Ligado As Boolean          'out
Dim Furou As Boolean           'out
Dim Sensor_Mesa As Boolean     'out
Dim Sensor_Esteira As Boolean  'out
Dim Motor_Esteira As Boolean   'in
Dim Furdeira_Sobe As Boolean   'in
Dim Furdeira_Desce As Boolean  'in
Dim Emergencia As Boolean      'out
Dim Saidas As Integer          'out
Dim Status As Integer          'in
'*************************************'

'********Consantes Globais************'
Dim DeslocaPeca As Integer
```

```
Dim DeslocaFuradeira As Integer
Dim cont    As Integer
'************************************'

'********Delcaração da inpout32.dll***'
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAddress As Integer) As Integer
Private Declare Sub Out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)


'*******Definições de Inicialização***'
Private Sub Form_Load()
   Ligado = False
   DeslocaPeca = 50
   DeslocaFuradeira = 30
End Sub

'*******Ação de Ligar******************'
Private Sub ligar_Click()
   Ligado = True
   Emergencia = False
   lblLiga.BackColor = &HFF00&
   lblDesliga.BackColor = &HC0C0C0
End Sub

'*******Ação de Desligar**************'
Private Sub desligar_Click()
   Ligado = False
   lblDesliga.BackColor = &HFF&
   lblLiga.BackColor = &HC0C0C0
End Sub

'*******Ação de Emergência************'
Private Sub Emerg_Click()
   Emergencia = True
End Sub

'******Ação de Colocar peça***********'
Private Sub ColocarPeca_Click()
   If Ligado = True And Peca1.Left < 6240 And Furadeira.Top <= 480 Then
      Peca1.Visible = True
      SensorEsteira.BackColor = &HFF00&
      Sensor_Esteira = True
   End If
End Sub

'*******Ação de Retirar Peça**********'
Private Sub RetirarPeca_Click()
   If Furou = True Then
      Peca2.Visible = False
      Furou = False
      SensorMesa.BackColor = &HC0C0C0
      Sensor_Mesa = False
      Estoque(ContPecas).Visible = True
      ContPecas = ContPecas + 1
      Peca1.Left = 360
   End If
End Sub

'*******Timer que move a peça na esteira************'
Private Sub Timer_Peca_Timer()
   If Peca1.Left < 6240 Then
      Peca1.Left = Peca1.Left + DeslocaPeca
   Else
      SensorEsteira.BackColor = &H8000000F
      Sensor_Esteira = False
```

```
        SensorMesa.BackColor = &HFF00&
        Sensor_Mesa = True
    End If
End Sub

'*******Timer que Desce furadeira*************'
Private Sub Timer_Furadeira1_Timer()
    Estado_furadeira = "descendo"
    If Furadeira.Top < 1800 Then
      Furadeira.Top = Furadeira.Top + yChange
    Else
        Timer_Furadeira1.Enabled = False
        Peca1.Visible = False
        Peca2.Visible = True
        Timer_Furadeira2.Enabled = True
    End If
End Sub

'*******Timer que Sobe furadeira*************'
Private Sub Timer_Furadeira2_Timer()
    Estado_furadeira = "subindo"
    If Furadeira.Top > 480 Then
        Furadeira.Top = Furadeira.Top - yChange
    End If
    If Furadeira.Top <= 480 Then
        Timer_Furadeira2.Enabled = False
        Furou = True
    End If
End Sub
'***********************************************************'
'*****A parte do código abaixo é responsável pela*********'
'*****troca de sinais entre planta virtual e a   *********'
'*****placa de intreface entre PC e CLP.          *********'
'***********************************************************'
'*******Timer que Libera Saida e atualiza a Entrada***********'
Private Sub Timer_Entrada_Saida_Timer()
        'envia o bite de saida para o registrador de dados 87280086
    Out 888, GeraSaida(Ligado, Furou, Sensor_Mesa, Sensor_Esteira, Emergencia)
        'recebe as enttradas multiplexadas do registrador de status
    Status = Inp(889)
        'interpretador dos sinas multilexados
    Select Case (Status)
     Case 127 Or 111 Or 95 Or 79 Or 63 Or 47 Or 31 Or 15
        If cont > 2 Then    'tudo parado
            Timer_Peca.Enabled = False
            Timer_Furadeira1 = False
            Timer_Furadeira2 = False
            Exit Sub
        End If
        If cont = 0 Then    'move peça na esteira
            Timer_Peca.Enabled = True
            Timer_Furadeira1 = False
            Timer_Furadeira2 = False
            Exit Sub
        End If
        If cont = 1 Then    'desce furadeira
            Timer_Peca.Enabled = False
            Timer_Furadeira1 = True
            Timer_Furadeira2 = False
            Exit Sub
        End If
        If cont = 2 Then    'sobe furadeira
            Timer_Peca.Enabled = False
            Timer_Furadeira1 = False
            Timer_Furadeira2 = True
            Exit Sub
        End If
```

```
        Case Else           'Tudo parado
            Timer_Peca.Enabled = False
            Timer_Furadeira1 = False
            Timer_Furadeira2 = False
    End Select
End Sub


'*******Função que gera Bite de saida***********'
" Esta função verifica como estão os sensores
" e os botões de liga, desliga e emergencia, para
" com isso criar o Bite de saida.
Public Function GeraSaida(L, F, SM, SE, EMG)
    Dim aux As Integer
    If L = True Then
        aux = aux Or 1      'ativa bit Ligado
    End If
    If F = True Then
        aux = aux Or 2      'ativa bit Furou
    End If
    If SM = True Then
        aux = aux Or 4      'ativa bit Sensor Mesa
    End If
    If SE = True Then
        aux = aux Or 8      'ativa bit Sensor Esteira
    End If
    If EMG = True Then
        aux = aux Or 16     'ativa bit Emergencia
    End If
    GeraSaida = aux
End Function
'*Timer que atualiza os bits de controle dos multiplexadores*'
" A sequencia 11 , 10 , 9 , 8, 15 , 14 , 13 , 12
" é devido ao dato de alguns bits no registrador
" de controle estarem invertidos. Logo esses valores
" corrigem esse caracteistica da LPT
Private Sub Timer_Controle_Timer()
    cont = (cont + 1) Mod (8)
    Select Case (cont)
        Case 0
            Out 890, 11
        Case 1
            Out 890, 10
        Case 2
            Out 890, 9
        Case 3
            Out 890, 8
        Case 4
            Out 890, 15
        Case 5
            Out 890, 14
        Case 6
            Out 890, 13
        Case 7
            Out 890, 12
    End Select
End Sub
```

5 CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma solução alternativa para o problema da escassez de laboratórios de ensino de qualidade e de baixo custo. A idéia apresentada neste trabalho

consiste na implementação de plantas virtuais em computadores de pequeno porte que permitam a total interação do aluno com a planta virtual como se esta fosse uma planta real.

Os recursos mínimos necessários para operar a planta virtual são relativamente modestos: uma fonte de tensão para estimular as entradas do módulo de interface e algum dispositivo de leitura ou sinalização para as saídas .

A planta virtual responde a entradas reais e fornece sinais de saída reais (através do painel de terminais) dentro dos padrões industrias reconhecidos pelos dispositivos de controle comerciais.

A automação e controle da planta virtual pode portanto ser realizada através de qualquer dispositivo de controle externo e desta forma permite a experimentação de todas as etapas de um projeto de controle.

O sistema permite a fácil implementação de funções de controle básicas e avançadas tais como: aquisição de dados, supervisão de variáveis, temporização de eventos, lógica de reles, controladores PID, através do controlador externo.

Como aplicação, uma maquina de furar automática virtual foi implementada e utilizada para ilustrar as características da bancada virtual proposta.

6 BIBLIOGRAFIA

MIN, F.B.M., Parallelism in Open Learning and Working Environments, **British Journal of Educational Technology,** Vol. 25, No.2, pp. 108-112. ISSN 0007-1013, 1994.

MIN, F.B.M., **Simulation Technology & Parallelism in Learning Environments;** Methods, Concepts, Models and Systems, Publisher: Academic Book Center, De Lier, ISBN 90-5478-036-3, 1996.

SCOTT, L.G.A., **Aplicação da Metodologia** Motivação pelo Desafio ao Ensino de Controle de Sistemas Lineares, Dissertação de Mestrado, PPGEE-UFMG, 1998.

ZHU, J.J., **Motivation by Challenge for Teaching Control Systems,** IEEE Control Systems Magazine, Vol. 14, No. 5, 1994.

NOMENCLATURA

PC = *"Personal Computer" (Computador Pessoal)*
PLC = *"Programmable Logic Computer" (Controlador Lógico Programável)*
LPT = *"Line Printer"*
CLP = *Controlador Lógico Programável*
PID = *(Controlador) Proporcional-Integral-Derivativo*

A VIRTUAL LABORATORY FOR TEACHING APPLICATIONS IN AUTOMATION AND CONTROL

Abstract: *This paper presents the design and implementation of a low-cost highly versatile and portable test bench for analysis and experimentation of closed loop control. The plants to be controlled are virtually implemented in a personal computer environment using Visual Basic as the programming language. A PLC-based external controller can be used to test and implement the control algorithms that can be directly manipulated by the user. As an application a simple tool was implemented in a personal computer and used to display the features of the test bench. The system permits the easy implementation of basic and advance control functions as data acquisition, variables supervision, events timing, relays logic, PID control, etc.*

Key-words: *Virtual Laboratory, Teaching Automation, Engineering Education.*