Hardware & software:

Acesso ao módulo Lynx AC1160-VA

1 Introdução

Parte do curso de Laboratório de Automação usa o módulo de aquisição de dados Lynx AC1160-VA, utilizado para atuação e coleta de sinais das plantas a serem controladas. O módulo possui dezesseis entradas analógicas (das quais no máximo duas serão utilizadas) e duas saídas analógicas. Neste documento denominaremos de *leitura* o procedimento de coletar dados da planta através das entradas analógicas e de *escrita* o procedimento de enviar valores calculados digitalmente para a planta através das saídas analógicas. A temporização do processo, bem como o processamento das entradas e cálculo das saídas são realizadas por meio de um computador digital conectado ao módulo, conforme a Figura 1.

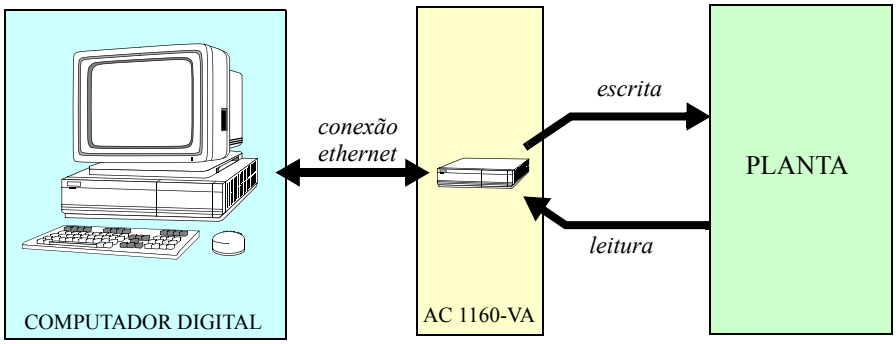


Figura 1: Diagrama de ligações

A programação do computador é feita com o uso do pacote Matlab e de sua linguagem (denominada *M* neste documento) e de duas bibliotecas: *SRT* (Soft Real Time) para controlar a temporização e *DAS* (Digital Acquisition System) para comunicação e configuração do módulo, desenvolvidas especialmente para o Laboratório de Automação.

As plataformas de hardware e software empregadas são apresentadas a seguir.

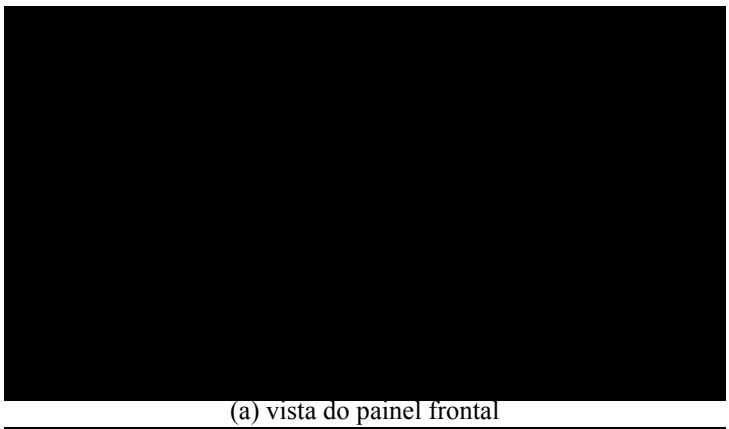
2 O módulo Lynx AC1160-VA

O módulo Lynx AC1160-VA (veja a Figura 2) é produzido pela Lynx Tecnologia Eletrônica e possui 16 entradas analógicas que utilizam um conversor A/D de 16 bits operando em uma faixa pré-programada de $\pm 10\mathrm{V}$ com impedância de entrada de $100\mathrm{k}\Omega$. Dessas, somente duas são utilizadas, denotadas por INP0 e INP1. As 2 saídas analógicas têm resolução de 12 bits, operam na faixa de $\pm 10\mathrm{V}$ e admitem uma carga mínima de $2\mathrm{k}\Omega$, portanto muito cuidado deve ser tomado ao se conectar as saídas analógicas para não danificar o equipamento. As saídas analógicas são denotadas por OUT0 e OUT1.

O módulo se comunica com o computador através de uma conexão Ethernet 10BaseT de 10Mbit/s.

Adicionalmente o módulo possui entradas e saídas digitais e diferentes opções de sincronismo. Esses recursos não serão utilizados no curso e não podem ser acessados através da biblioteca DAS

O módulo possui três indicadores luminosos (LEDs) no painel traseiro que apresentam informações



(a) vista do painel frontal

(b) vista do painel traseiro

Figura 2: O módulo Lynx AC1160-VA

diversas sobre seu funcionamento.

a) ATV (LED verde) sinaliza o estado do módulo (veja a Tabela 1 a seguir);

b) LAN (LED amarelo) sinaliza a troca de mensagens na conexão Ethernet;

c) LINK (LED amarelo) sinaliza que a conexão Ethernet está ativada.

O principal indicador é o LED ATV. A Tabela 1 apresenta a codificação do estado do módulo em função das mensagens do indicador.

Tabela 1: Indicador luminoso ATV

Mensagem	Significado	Observações
Aceso	Módulo ligado, mas não configurado.	Imediatamente após ser ligado.
OFF + 1P[a]	Módulo com configuração válida.	Não esperado em situação normal.
OFF + 2P	Módulo executando aquisição de dados.	Após o comando das_start.
OFF + 3P	Módulo em pausa na aquisição.	Não esperado em situação normal.
OFF + 4P	Módulo encerrou aquisição de dados.	Após o comando das_stop.
OFF + 5P (ou mais)	Erro de comunicação ou operação.	Não esperado em situação normal. Pode indicar problemas na conexão de rede.
Apagado	Módulo desligado ou em falha.	

a. OFF (pausa apagado) + 1P (1 piscada)

3 O módulo de conexão

Para se facilitar o manuseio do sistema, a conexão com o módulo Lynx AC1160-VA é feita através de um módulo de conexão, cujo painel é mostrado na Figura 3.

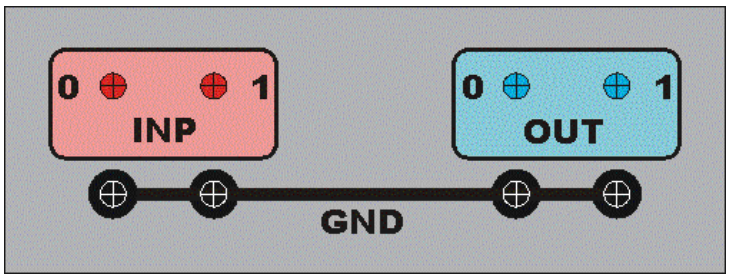


Figura 3: Painel de conexões

Os terminais de referência GND são conectados entre si e devem ser conectados aos seus correspondentes nos sistemas a serem controlados.

IMPORTANTE

Jamais ligue os terminais OUT0 ou OUT1 diretamente a um terminal de terra ou aos terminais GND.

Jamais ligue os terminais OUT0 ou OUT1 a uma fonte de tensão (como um gerador de funções ou a rede elétrica).

Jamais ligue os terminais INP0 ou INP1 a fontes de tensão acima da faixa de $\pm 10\,\mathrm{V}$ (como por exemplo a rede elétrica).

4 Plataforma de software

4.1 Soft real-time vs. hard real-time

Programas de computador que implementam algoritmos de controle devem operar em tempo real. Os programas de computador com que interagimos normalmente, tais como gerenciadores de planilhas, editores de texto, navegadores, etc., não têm esse requisito. Um programa convencional é dito correto se quando alimentado com as informações adequadas executa operações corretas na ordem correta. Um programa em tempo real deve adicionalmente executar as operações corretas nos instantes corretos.

Usualmente espera-se que um programa convencional, em um ambiente como o Matlab (sem as bibliotecas apresentadas a seguir, obviamente), deva ser executado mais rapidamente em um computador de maior capacidade de processamento. Um programa em tempo real (como os que serão feitos neste curso), por outro lado, deveria levar o mesmo tempo de execução, qualquer que fosse o desempenho do computador utilizado.

Os algoritmos de controle que implementamos normalmente assumem que o cálculo de atualização da saída do controlador seja realizado a intervalos de tempo estritamente constantes e sem falhas. Esse requisito nos leva à necessidade dos programas operarem em tempo real.

Nos computadores de hoje, a temporização dos processos em andamento é realizada pelo sistema operacional. Sistemas operacionais de uso geral, como MS-Windows, Linux ou MacOS realizam essa operação visando o máximo desempenho geral do sistema, o que impede que um programa qualquer tenha controle sobre os tempos em que ele é executado. A qualquer momento o sistema operacional pode interromper a execução de um programa para executar outra tarefa que, segundo critérios de escalonamento, deva se sobrepor a

ele. Normalmente esse tipo de intervenção é imperceptível ao usuário, ou inócua em seus efeitos, porém pode prejudicar ou mesmo impedir o correto funcionamento de um programa que implemente um algoritmo de controle. Nesses sistemas operacionais de tempo real dizemos que o escalonamento é *não-determinístico*, no sentido de que não se pode prever ou garantir como ele será feito.

Existem sistemas operacionais cujo escalonamento de processos é feito visando preservar a execução em tempo real de programas pré-determinados, permitindo se escalonar a ocorrência de eventos com grande precisão temporal. Exemplos de sistemas operacionais que operam ou podem operar sobre este princípio são VxWorks, QnX, LynxOS, RT-Linux e extensões de tempo real para o MS-Windows. Nesses sistemas operacionais de tempo real dizemos que o escalonamento é *determinístico*, no sentido de que ele pode ser imposto pelo usuário.

De modo geral, a operação em tempo real é muito mais problemática quando envolve atividades como comunicação com dispositivos, interface gráfica, programação orientada a eventos, etc, que tornam a programação e utilização de um sistema determinístico muito mais complexa. Por outro lado, sob condições controladas e requisitos pouco estritos de desempenho, é possível obter resultados aceitáveis com sistemas operacionais não-determinísticos operando em tempo real.

Nesses casos utilizamos o termo tempo real não-determinístico (ou *soft real-time*), em oposição a tempo real determinístico (*hard real-time*). Por uma questão de conveniência e para evitar a sobrecarga excessiva associada às implementações em tempo real determinístico, o curso de Laboratório de Automação adota uma abordagem não-determinística, com o sistema operacional MS-Windows e ambiente Matlab.

Os programas de controle são escritos em M com o auxílio de duas bibliotecas: SRT, para a temporização; e DAS para configuração e comunicação com o módulo Lynx AC1160-VA. Como se verá a seguir, as implementações são bastante simples e adequadas ao ambiente do laboratório, onde questões como desempenho e confiabilidade podem ser, até certo ponto, negligenciadas.

4.2 A biblioteca SRT

A biblioteca SRT possui duas funções, que usadas em conjunto, permitem que a temporização dos programas seja feita com um grau de determinismo frequentemente aceitável. As funções são as seguintes.

a) `srt_conf`

Propósito: Especifica a frequência de amostragem e inicializa os timers do sistema.

Sintaxe:
```
srt_conf(T)
ou
ST = srt_conf(T)
```

Comentários: Esta rotina deve ser utilizada uma única vez imediatamente antes do loop principal de controle (mais sobre isso a seguir). O comando `srt_conf(0.1)` por exemplo, faz com que a temporização seja feita a uma frequência de 10Hz (isto é, um período de amostragem de 0,1s). É importante observar que a faixa de frequências em que o sistema pode operar obedece a certos limites. O limite superior é dado pelo tempo exigido pelo módulo para realizar a conversão dos dados analógicos em valores digitais e sua transmissão, e também pelo processamento dos dados transmitidos por parte do computador e pela quantidade de cálculos a serem executados pelo programa no tempo alocado. Um limite máximo de frequência de 100Hz, que é mais do que suficiente para os objetivos do curso, foi pré-programado na biblioteca.
O argumento `T` é o período de amostragem em segundos e o resultado `ST` (opcional) é o código de erro da função (`ST=0` para operação bem sucedida e valores diversos conforme o problema observado).

b) `srt_waitbusy`

Propósito: Controla a temporização.

Sintaxe:
```
srt_waitbusy
ou
TR = srt_waitbusy
```

Comentários: Esta rotina deve ser incluída no loop principal de controle preferencialmente como o último comando dentro do loop (mais sobre isso a seguir). A função executa um loop de espera (daí o nome *waitbusy*) para controlar a temporização do programa. A duração do loop de espera é exatamente o tempo necessário para que se complete o intervalo de tempo T, conforme definido pela última execução do comando srt_conf. O intervalo de tempo T é contado desde a última execução do comando srt_waitbusy ou desde a última execução do comando srt_conf (no caso da primeira execução do loop principal de controle). Ao contrário de um sistema operacional determinístico, onde a temporização é realizada por meio de interrupções, deixando o computador livre para realização de outras tarefas durante o tempo ocioso do programa, a biblioteca SRT mantém o processador em uso e o programa em execução, para minimizar as atuações do escalonador MS-Windows.
O resultado TR (opcional) é o tempo restante em segundos do período de amostragem após a execução do loop de espera. Idealmente este valor deve ser nulo, mas na prática ele é um número negativo (usualmente bastante pequeno) que indica o atraso ocorrido no procedimento. Atrasos são cumulativos durante a execução do programa.
Caso ocorra um atraso maior que meio período de amostragem, uma mensagem (código R501) é imediatamente apresentada na tela de comandos do Matlab.

As duas funções, quando utilizadas em conjunto e com as funções da biblioteca DAS, permitem que se faça implementações bastante simples de algoritmos de controle.

4.3 A biblioteca DAS

As funções da biblioteca DAS são responsáveis pela comunicação com o módulo e por sua configuração. Há quatro funções simples que executam as principais tarefas, e que em conjunto com as funções da biblioteca SRT permitem a implementação de sistemas de controle bastante sofisticados. As funções são as seguintes.

a) `das_start`

Propósito: Configura o módulo Lynx AC1160-VA e inicia o processo de aquisição.

Sintaxe:
```
das_start
ou
ST = das_start
```

Comentários: Esta rotina deve ser utilizada uma única vez no início do programa, antes do loop principal de controle. As outras rotinas da biblioteca DAS só operam corretamente após esta função ser executada, e portanto esta deve ser a primeira rotina desta biblioteca no fluxo de execução de qualquer programa. O resultado `ST` (opcional) é o código de erro da função (`ST=0` para operação bem sucedida e valores diversos conforme o problema observado).
Uma mensagem é apresentada na tela de comandos do Matlab informando se a configuração foi bem sucedida ou então indicando o problema observado.

b) `das_readAD`

Propósito: Lê dois canais do conversor A/D.

Sintaxe: `[INP0,INP1]=das_readAD`

Comentários: Esta rotina lê dois canais do conversor A/D do módulo Lynx AC1160-VA e os escreve nas variáveis `INP0` e `INP1`. Os valores obtidos são dados em Volts. Em sua implementação usual, `das_readAD` deve ser colocada no loop principal de controle. A rotina somente opera corretamente se `das_start` tiver sido executada previamente.

c) `das_writeDA`

Propósito: Escreve em dois canais do conversor D/A.

Sintaxe:
```
das_writeDA(OUT0,OUT1)
ou
ST=das_writeDA(OUT0,OUT1)
```

Comentários: Esta rotina escreve os valores dados por `OUT0` e `OUT1` no conversor D/A do módulo Lynx AC1160-VA. `OUT0`, `OUT1` devem ser escalares reais na faixa de ± 10 . Valores correspondentes a `OUT0` e `OUT1` em Volts são produzidos na saída do conversor D/A. Em sua implementação usual, `das_writeDA` deve ser colocada no loop principal de controle. A rotina somente opera corretamente se `das_start` tiver sido executada previamente. O resultado `ST` (opcional) é o código de erro da função (`ST=0` para operação bem sucedida e valores diversos conforme o problema observado).

a) `das_stop`

Propósito: Finaliza a operação do módulo Lynx AC1160-VA.

Sintaxe:
```
das_stop
ou
ST = das_stop
```

Comentários: Esta rotina deve ser utilizada uma única vez no final do programa, depois do loop principal de controle. Esta deve ser a última rotina desta biblioteca no fluxo de execução de qualquer programa. O resultado `ST` (opcional) é o código de erro da função (`ST=0` para operação bem sucedida e valores diversos conforme o problema observado).
Uma mensagem é apresentada na tela de comandos do Matlab informando se a finalização foi bem sucedida ou então indicando o problema observado. `das_stop` escreve valores nulos nos conversores D/A antes da finalização.

4.4 Programas típicos

A seguir são apresentados uma série de programas simples explorando o uso das bibliotecas. Estes programas estão disponíveis na pasta `c:\labaut`.

Programa 1 (`prog1.m`)
Um programa simples que apenas coleta dados.

```
%Prog1.m (V1.0 RP Marques)
%Coleta de dados a 50Hz durante 10s (500 pontos)

T = 1/50;               % Frequencia de amostragem de 50Hz
INP0 = zeros(1,500);    % Vetor de 500 pontos para guardar os dados
INP1 = zeros(1,500);    % Vetor de 500 pontos para guardar os dados
das_start;              % Configura o módulo Lynx
srt_conf(T);            % Configura o sistema de tempo real
                        % (colocar imediatamente antes do loop principal)
for k=1:500                     % LOOP PRINCIPAL (início)
    [INP0(k),INP1(k)]=das_readAD; % Lê os dados

    srt_waitbusy;               % Aguarda o tempo necessário para completar
                                % o período de 0,02s
end                             % LOOP PRINCIPAL (final)
das_stop;               % Finaliza a conexão com o módulo Lynx
```

Programa 2 (`prog2.m`)
Um programa que implementa a função de transferência $\frac{Y(s)}{U(s)} = \frac{1}{s}$ a uma frequência de amostragem de 10Hz (equivalente a $\frac{Y(z)}{U(z)} = \frac{0,1}{z-1}$) em um loop infinito.

```
%Prog2.m (V1.0 RP Marques)
%Loop infinito

% Note que Y(z)/U(z)=0,1/(z-1) corresponde à eq. de diferenças
% y(k) = y(k-1) + 0,1*u(k) implementada em 10Hz

T = 1/10;               % Frequencia de amostragem de 10Hz
UK_1 = 0;               % u(k-1)
YK_1 = 0;               % y(k-1)

das_start;              % Configura o módulo Lynx
srt_conf(T);            % Configura o sistema de tempo real
                        % (colocar imediatamente antes do loop principal)

while 1,                        % LOOP INFINITO (início)
    [UK,INP1] = das_readAD;     % Lê os dados (INP1 não é utilizado)
    YK = YK_1 + 0.1*UK_1;       % Calcula y(k) = y(k-1)+0,1*u(k)
    das_writeDA(YK,YK);         % Escreve y(k) no conversor D/A
    UK_1=UK; YK_1=YK;           % Atualiza y(k),u(k) para o próximo passo
    srt_waitbusy;               % Aguarda o fim do período de 0,1s
end                             % LOOP INFINITO (final)

das_stop;               % Finaliza a conexão com o módulo Lynx
                        % (jamais será executado se o loop for infinito)
```

Programa 3 (`prog3.m`)
A mesma função de transferência num loop finito (operando durante 20s).

```
%Prog3.m (V1.0 RP Marques)
%Loop finito a 10Hz durante 20s (200 pontos)

% Note que Y(z)/U(z)=0,1/(z-1) corresponde à eq. de diferenças
% y(k) = y(k-1) + 0,1*u(k) implementada em 10Hz

T = 1/10;               % Frequencia de amostragem de 10Hz
U = zeros(1,200);       % Vetor de 200 pontos para guardar os dados
Y = zeros(1,200);       % Vetor de 200 pontos para guardar os dados

das_start;              % Configura o módulo Lynx
srt_conf(T);            % Configura o sistema de tempo real
                        % (colocar imediatamente antes do loop principal)

for k=2:200,                    % LOOP PRINCIPAL (início)
                                % (note que ele inicia em k = 2)
    [U(k),INP1] = das_readAD;   % Lê os dados (INP1 não é utilizado)
    Y(k) = Y(k-1) + 0.1*U(k-1); % Calcula y(k) = y(k-1)+0,1*u(k)
    das_writeDA(Y(k),Y(k));     % Escreve y(k) no conversor D/A
    srt_waitbusy;               % Aguarda o fim do período de 0,1s
end                             % LOOP PRINCIPAL (final)

das_stop;               % Finaliza a conexão com o módulo Lynx

t = (0:199)*T;          % Vetor com tempo em segundos
plot(t,U,t,Y);          % Traça um gráfico com u e y
```

Programa 4 (`prog4.m`)
Um programa que implementa um controlador proporcional operando a 5Hz durante 120s.

```
%Prog4.m (V1.0 RP Marques)
%Controle proporcional a 5Hz durante 120s (600 pontos)
% Controle: u(k) = Kc*e(k), onde e(k)=r-y(k)
% r é o setpoint
% y é a saída da planta
% u é a saída do controlador

T = 1/5 ;               % Frequencia de amostragem de 5Hz
U = zeros(1,600);       % Vetor de 2400 pontos para guardar os dados
Y = zeros(1,600);       % Vetor de 2400 pontos para guardar os dados
R = 0;                  % Setpoint
Kc = 1;                 % Ganho do controlador

das_start;              % Configura o módulo Lynx
srt_conf(T);            % Configura o sistema de tempo real
                        % (colocar imediatamente antes do loop principal)

for k=1:600,                    % LOOP PRINCIPAL (início)
    [Y(k),INP1] = das_readAD;   % Lê os dados (INP1 não é utilizado)
    U(k) = Kc*(R - Y(k));       % Calcula u(k)
    das_writeDA(U(k),U(k));     % Escreve u(k) no conversor D/A
    srt_waitbusy;               % Aguarda o fim do período de 0,2s
end                             % LOOP PRINCIPAL (final)

das_stop;               % Finaliza a conexão com o módulo Lynx
t = (0:599)*T;          % Vetor com tempo em segundos
plot(t,U,t,Y);          % Traça um gráfico com u e y
```

5 Resolução de problemas

5.1 Diagnóstico do módulo Lynx AC1160-VA

O diagnóstico do módulo Lynx AC1160-VA deve ser feito inicialmente a partir de seus indicadores luminosos (Figura 2) e de seus estados típicos (Tabela 1). Quando ligado, devemos ter a seguinte configuração: ATV aceso; LAN apagado (ou piscando esporadicamente); LINK aceso. Após a execução da rotina das_start devemos ter a seguinte configuração: ATV (OFF+2P); LAN aceso; LINK aceso. Após a execução da rotina das_stop devemos ter: ATV (OFF+4P); LAN apagado (ou piscando esporadicamente); LINK aceso.

Qualquer configuração diferente das acima indica problemas no sistema, e deve ser reportada ao responsável.

Em qualquer situação, se o indicador LINK estiver apagado isso é uma indicação de ausência de conexão entre o computador e o módulo, o que impede o correto funcionamento do sistema.

5.2 Diagnóstico de software

As bibliotecas SRT e DAS emitem mensagens de erro de dois tipos básicos: erros de parâmetros, denotados por PXXX e erros de runtime, denotados por RXXX. Erros de parâmetros interrompem obrigatoriamente a execução do programa, ao passo que erros de runtime apenas produzem mensagens de advertência na tela de comandos do Matlab. Apesar de não interromperem a execução do código, erros de runtime indicam situações que devem obrigatoriamente ser corrigidas.

Nem todas as possibilidades de uso incorreto das rotinas são contempladas pelos códigos acima, de modo que configurações não previstas podem gerar erros não listados neste documento.

A Tabela 2 abaixo apresenta a lista de erros de parâmetros das duas bibliotecas,.

Tabela 2: Erros de parâmetro

DAS		
erro	rotina	comentário
P101	`das_readAD`	Os dois canais INP0 e INP1 (mesmo que um deles não seja utilizado) devem ser atribuídos pela rotina `das_readAD`.
P201	`das_writeDA`	Os dois canais OUT0 e OUT1 (mesmo que um deles não seja utilizado) devem ser argumento da rotina `das_writeDA`. Usualmente pode-se utilizar o valor zero para um canal não utilizado.
P202	`das_writeDA`	Os argumentos correspondentes a OUT0 e OUT1 devem ser do tipo escalar (i.e. não podem ser matrizes, números complexos, strings, etc.)
P203	`das_writeDA`	
SRT		
P401	`srt_conf`	Ausência do timer HRPC. Indica uma incompatibilidade do hardware do computador com a biblioteca SRT.
P402	`srt_conf`	`srt_conf` aceita um único parâmetro: o período de amostragem (T).
P403	`srt_conf`	O argumento T deve ser do tipo escalar (não pode ser matriz, complexo, etc.)
P404	`srt_conf`	O escalar T deve ser um número positivo (não pode ser nulo ou negativo).
P405	`srt_conf`	A versão atual limita a frequência máxima de amostragem a 100Hz.
P501	`srt_waitbusy`	`srt_waitbusy` não aceita argumentos.

A Tabela 3 apresenta os erros de runtime. Estes erros não interrompem a execução do programa, porém indicam problemas que devem ser obrigatoriamente abordados para o correto funcionamento do programa.

Tabela 3: Erros de runtime

DAS		
erro	rotina	comentário
R001	`das_start`	Estes erros indicam falhas internas do software ou problemas de integração com o MS-Windows. A ocorrência de qualquer destes erros impede a operação do sistema.
R002	`das_start`	
R003	`das_start`	
R004	`das_start`	
R005	`das_start`	
R006	`das_start`	
R007	`das_start`	Este erro pode indicar uma falha na conexão de rede com o módulo Lynx AC1160-VA ou falha de hardware no módulo.
R008	`das_start`	Estes erros indicam falhas de hardware no módulo Lynx AC1160-VA.
R009	`das_start`	
R010	`das_start`	
R011	`das_start`	Este erro pode indicar uma falha na conexão de rede com o módulo Lynx AC1160-VA ou falha de hardware no módulo.
R012	`das_start`	Este erro indica uma falha interna do software ou problema de integração com o MS-Windows
R013	`das_start`	Este erro pode indicar uma falha na conexão de rede com o módulo Lynx AC1160-VA ou falha de hardware no módulo.
R014	`das_start`	Este erro indica uma falha interna do software ou problema de integração com o MS-Windows
R015	`das_start`	Erro inesperado de software (sem diagnóstico).
R101	`das_readAD`	Estes erros indicam falhas internas do software ou problemas de integração com o MS-Windows.
R102	`das_readAD`	
R103	`das_readAD`	
R104	`das_readAD`	
R105	`das_readAD`	Este erro pode indicar que (i) Há uma falha na conexão de rede; (ii) O computador não é capaz de efetuar as tarefas solicitadas no tempo alocado; (iii) Outros processos interferem com o programa de controle, atrasando sua execução.
R106	`das_readAD`	Erro inesperado de software (sem diagnóstico).
R201	`das_writeDA`	Estes erros indicam falhas internas do software ou problemas de integração com o MS-Windows.
R202	`das_writeDA`	
R203	`das_writeDA`	
R204	`das_writeDA`	

Tabela 3: Erros de runtime

R205	`das_writeDA`	Este erro pode indicar que (i) Há uma falha na conexão de rede; (ii) O computador não é capaz de efetuar as tarefas solicitadas no tempo alocado; (iii) Outros processos interferem com o programa de controle, atrasando sua execução.
R206	`das_writeDA`	Este erro pode indicar uma falha na conexão de rede com o módulo Lynx AC1160-VA ou falha de hardware no módulo.
R207	`das_writeDA`	
R208	`das_writeDA`	Erro inesperado de software (sem diagnóstico).
R209	`das_writeDA`	Algum dos argumentos (OUT0, OUT1) se encontra fora da faixa admissível de $\pm 10\text{V}$. A saída efetivamente produzida será de 10V ou -10V .
R301	`das_stop`	Estes erros indicam falhas internas do software ou problemas de integração com o MS-Windows.
R302	`das_stop`	
R303	`das_stop`	
R304	`das_stop`	
R305	`das_stop`	
R306	`das_stop`	Este erro pode indicar uma falha na conexão de rede com o módulo Lynx AC1160-VA ou falha de hardware no módulo.
R307	`das_stop`	Estes erros indicam falhas de hardware no módulo Lynx AC1160-VA.
R308	`das_stop`	
R309	`das_stop`	
R310	`das_stop`	Erro inesperado de software (sem diagnóstico).
SRT		
R501	`srt_waitbusy`	Este erro indica perda de tempo real. As causas mais frequentes para esse erro são as seguintes: (i) Frequência de amostragem alta demais para a quantidade de cálculos determinada (neste caso o erro R501 ocorre com frequência elevada, possivelmente em todos os períodos de amostragem); (ii) Outros processos interferem com a execução do programa atrasando sua execução (neste caso o erro R501 ocorre esporadicamente).

5.3 Recomendações

Para evitar problemas e otimizar o desempenho dos programas, as seguintes recomendações gerais devem ser seguidas.

a) Manter os programas simples;

b) Evitar executar outros programas concomitantemente com os programas de controle;

Recomendações mais específicas incluem.

a) Evitar incluir comandos desnecessários dentro do loop principal, especialmente comandos que demandam tempo elevado para sua execução (e.g. escrita em tela, gravação em disco, gráficos, etc.);

b) Alocar as variáveis antes de sua utilização (como feito nos programas apresentados acima);

c) Em caso de interrupção de um programa (por erro ou intervenção do usuário) antes da execução do comando `das_stop` (que deve sempre ser incluído ao final do programa), o módulo Lynx AC1160-VA continuará em modo de aquisição. Para corrigir a situação deve-se executar o comando `das_stop` diretamente do prompt do Matlab.

5.4 Problemas e dúvidas frequentes

Esta seção se encontra em construção. O histórico de utilização do sistema ainda não permite avaliar com clareza quais são os problemas mais frequentemente encontrados.

a) **Algum canal de entrada (INP0, INP1) não coleta nenhum sinal ou coleta sinais estranhos.**
Diagnóstico 1: Verifique se o canal de entrada não se encontra desconectado. Canais de entrada desconectados podem copiar sinais de outros canais por indução (esta parece ser uma característica do hardware do módulo Lynx AC1160-VA).
Diagnóstico 2: Verifique o aterramento. O módulo e o sistema conectado a ele (a planta) devem ter referências comuns.

b) **O programa sendo executado parece ser diferente do programa editado.**
Diagnóstico 1: Verifique se de fato o programa executado é o programa editado ou se é uma cópia ou versão diferente.
Diagnóstico 2: O comando
```
>> clear functions
```
força o Matlab a recarregar os programas do disco (em vez de utilizar versões armazenadas na memória RAM).

5.5 Créditos

A biblioteca SRT foi escrita por R.P.Marques e aproveita código com direitos autorais de K. Wansborough. A biblioteca DAS foi escrita por T.R. Contim e R.P. Marques.

RPM/2012a