DRIVER DA CAD12/56 PARA WINDOWS 98

PARA USO COM O LABVIEW

1. INTRODUÇÃO

1.1. Escopo

O objetivo deste documento é apresentar as funções da biblioteca DLL de acesso à placa CAD12/56 para uso com o LabVIEW.

1.2. Audiência

Esta documentação se destina aos programadores de aplicativos em LabVIEW 5.0 que realizam aquisição de sinais com a placa CAD12/56.

Para um bom entendimento desta nota de aplicação, recomenda-se que o leitor tenha conhecimento prévio dos seguintes assuntos:

- aquisição de dados
- utilização da CAD12/5
- LabVIEW 5.0 for Windows 98

1.3. Aplicabilidade

O driver WDM_1256 deve ser utilizado somente no sistema operacional Windows 98 ou compatível e com a placa A/D CAD12/56. O driver não possui suporte para DMA e ao contador 2 da placa.

Para maiores informações sobre a CAD12/56, consulte o manual de referência técnica da placa.

2. INSTALANDO O DRIVER WDM_1256

Os arquivos fornecidos para utilização do driver WDM_1256:

WDM_1256.SYS	kernel mode driver para Windows 98
WDM_1256.ini	arquivo de registro do driver
WDM_1256.DLL	biblioteca DLL de acesso ao driver
WLV_1256.DLL	biblioteca DLL de acesso ao driver para uso com o LabVIEW
WDMapi.pas	módulo em pascal para a interface com o driver
TESTE.EXE	programa de teste do driver
Delphi*.*	exemplo com programa fonte em Borland Delphi
VB5*.*	exemplo com programa fonte em Visual Basic 5.0
Read_AI.VI	exemplo em LabVIEW com aquisição por pooling
Acq_Int.VI	exemplo em LabVIEW com aquisição por interrupção

2.1. Instalando o Driver

Siga as seguintes etapas para instalar o device driver da CAD12/56 no Windows 98:

1. A partir da barra de tarefa, execute **`Iniciar/Programas/Configurações/Painel de controle`** para entrar na janela *Painel de controle* do Windows 98.
2. Na janela *Painel de controle*, clique sobre o ícone *Adicionar novo hardware*. O Windows apresentará a janela do *Assistente para instalar novo hardware*. Clique sobre o botão *Avançar*. Novas janelas serão apresentadas, clique em avançar até que seja apresentada a pergunta: "Deseja que o Windows procure seu novo hardware?". Clique em não e em seguida clique em *Avançar*.
3. O Assistente de instalação do Windows apresentará uma janela para você selecionar o tipo de hardware a ser instalado. *Selecione Outros dispositivos* e clique sobre o botão *Avançar*.
4. Na janela seguinte clique sobre o botão *Com disco*.
5. Insira o disquete do driver da CAD12/56 para Windows 98 na unidade de disquete A.
6. Na janela *Instalar a partir do disco*, selecione a unidade de disquete A: e clique sobre o botão *Procurar*.
7. Selecione *WDM_1256.INF* e clique sobre o botão *OK*. Clique sobre o botão *OK* novamente para retornar na janela do Assistente de adicionar novo hardware.
8. Na janela do *Assistente* será apresentada a lista de modelos, selecione CAD12/56 e clique sobre o botão *Avançar* e siga as instruções do *Assistente*. Ao pressionar o botão *Avançar*, o Windows poderá demorar para avançar para o próximo passo.

2.2. Como Alterar a Configuração de Hardware do Driver

A placa CAD12/56 utiliza endereços de I/O e um canal de interrupção do sistema. Eventualmente esses recursos especificados para uso do device driver podem estar em conflito com outras placas instaladas no microcomputador. Para resolver esse conflito, você precisará modificar a alocação do recurso em conflito alterando a configuração do driver da CAD12/56 ou do outro driver em conflito.

Para alterar os recursos alocados ao driver da CAD12/56, clique sobre o ícone *Sistema* do *Painel de controle* do Windows 98.

1. Na janela do *Propriedade de Sistema*, seleciona a página *Gerenciador de dispositivos*.
2. Selecione o dispostivo CAD12/56 na categoria CAD1256.
3. Clique sobre o botão *Propriedade*. Será apresentada a janela *Propriedade de CAD12/56*.
4. Selecione a página *Recursos*. Na página será apresentada os conflitos existentes na parte inferior da janela.
5. Se for um conflito de *Intervalo de entrada/saída*, selecione uma outra configuração básica. O driver possui quatro opções de configuração básica (0 a 3) que correspondem respectivamente aos endereços base 300, 310, 340 e 350 da placa, que devem ser selecionadas por jumper na placa

CAD12/56 (consulte o manual de referência técnica da placa).

6. Se o conflito for de *Pedido de interrupção*, clique sobre *Pedido de interrupção* e em seguida clique sobre o botão *Alterar configuração*. Na janela apresentada, selecione uma das opções disponíveis que não esteja em conflito com outro dispositivo. A seleção de canal de interrupção na CAD12/56 é realizada por software e não depende de nenhum jumper de configuração.
7. Siga as demais instruções do Windows.

Para informações mais detalhada do Painel de controle do Windows, consulte documentação da Microsoft.

3. INTERFACE COM O LABVIEW

O acesso do programa em LabVIEW às funções do driver da CAD12/56 é realizada através do bloco *Call Library Function* do LabVIEW. Para cada chamada de uma função do driver deve haver um correspondente bloco *Call Library Function*.

Na configuração do *Call Library Function*, deve-se especificar no campo *Library Name or Path* o valor **`WLV_1256.DLL`** e no campo *Calling Conventions* o valor **`C`**. Os demais campos da configuração são dependentes da função. O próximo tópico descreve as funções da biblioteca DLL de acesso ao driver da CAD12/56 para Windows 98.

3.1. Estrutura de Dados do Driver

O driver mantém uma estrutura de dados interna com todas as informações necessárias para a utilização da CAD12/56. Algumas dessas informações (endereço base padrão ecanal de interrupção) são passadas para o driver através do Registry do Windows NT.

O driver possui uma memória de canais onde são armazenados a ordem em que os canais serão aquisitados e os seus respectivos ganhos. Com essa estrutura você pode, por exemplo, informar ao driver para aquisitar 4 canais na seguinte ordem e ganhos: canal 0 em ±5 volts, canal 3 em ±10 volts, canal 12 em ±2.0 volts e o canal 15 em 0-10 volts).
O driver possui um buffer circular de 64K amostras para a aquisição de sinais. Por exemplo, para a aquisição de 4 canais a 1000 Hz por canal, o buffer teria capacidade de armazenar até 16 segundos sem que o programa aplicativo remova os dados do buffer circular.

3.2. Constantes

A tabela seguinte apresenta as constantes retornadas pela primitiva *WLV_GetAcquiredData*.

Constante	Valor	Descrição
ieNoError	0	Não ocorreu nenhum erro desde a última chamada da primitiva WLV_GetAcquiredData
ieIntOverrun	1	Indica que houve interrupt overrun. Ou seja, antes do término do tratamento da interrupção, houve um novo pedido de interrupção da placa A/D.
ieAD_Error	2	Indica que houve erro no conversor A/D da placa.
ieBufOverflow	3	Overflow no buffer de aquisição do driver
ieDriver	100	Erro de acesso ao driver
ieAcqStop	101	Aquisição não está ativa
ieLabView	102	Dimensão errada no buffer do LabView passado na função WLV_GetSamples. Número de colunas é diferente do número de sinais habilitados para aquisição.

A tabela seguinte apresenta as constantes retornadas pela primitiva *WLV_AcquisitionSetup*.

Constante	Valor	Descrição
seNoError	0	Nenhum erro
seNoSignal	1	Não foi programado nenhum sinal para a aquisição.

4. DESCRIÇÃO DAS FUNÇÕES DO WLV_1256.DLL

Neste tópico são descritas as funções da biblioteca DLL de acesso ao driver da CAD12/56 especialmente implementada para interfaceamento com o LabVIEW. A tabela abaixo lista as funções do WLV_1256.DLL.

Primitiva	**Descrição**
WLV_Open	Inicia o acesso ao driver
WLV_Close	Finaliza o acesso ao driver
WLV_GetVersion	Informa a versão e revisão do driver
WLV_GetAIO_Caps	Informa os recursos de entrada e saída analógica disponíveis no driver
WLV_GetDIO_Caps	Informa os recursos de entrada e saída digital disponíveis no driver
WLV_SetAiRange	Seleciona faixa de entrada do conversor A/D
WLV_ReadAi	Leitura de canal de entrada analógica
WLV_WriteAo	Escreve em canal de saída analógica
WLV_ReadDI	Leitura dos ports de entrada digital
WLV_WriteDO	Escrita nos ports de saída digital
WLV_Clear_CM	Limpa a memória de canais da aquisição por interrupção
WLV_Insert_AI_CM	Insere um canal de entrada analógica para aquisição
WLV_Insert_DI_CM	Insere os ports de saída digital para aquisição
WLV_AcquisitionSetup	Programa aquisição por interrupção
WLV_StartAcquisition	Inicia a aquisição por interrupção
WLV_StopAcquisition	Finaliza a aquisição por interrupção
WLV_GetSamples	Leitura das amostras do buffer de aquisição por interrupção

4.1. Primitiva WLV_Open

```
uInt8 WLV_Open (void);
```

Esta primitiva abre o acesso ao driver e deve ser a primeira primitiva a ser executada pelo programa aplicativo. A primitiva retorna o valor *zero* quando a primitiva foi executada com sucesso. Umas das causas possíveis são:

Causa	**Verificação**	**Solução**
Driver não instalado	No Painel de Controle do Windows 98, clique sobre Sistema. Se o driver da CAD12/56 estiver instalado, ele deve estar na lista de dispositivos apresentada.	Instale o driver
Driver não iniciado ou erro na parametrização	No Painel de Controle do Windows 98, clique sobre Sistema. Verifique em propriedades da CAD12/56 se há algum erro reportado pelo Windows.	Proceda como descrito em 2.2.
Placa não instalada ou falha na iniciação da placa	Verifique se o endereço base da placa e o canal de interrupção configurado por jumpers na placa estão coerentes com o especificado na parametrização do driver. Para mudar a parametrização do driver, consulte 2.2.	Instale a placa CAD12/56 e/ou altere a parametrização do driver.

Veja também a primitiva *WLV_Close*.

4.2. Primitiva WLV_Close

```
void WLV_Close (void);
```

Esta primitiva finaliza o acesso ao driver e deve ser a última primitiva a ser executada pelo programa aplicativo. Para cada chamada bem sucedida da primitiva *WLV_Open* deve haver uma correspondente chamada da *WLV_Close*.

Veja também a primitiva *WLV_Open*.

4.3. Primitiva WLV_GetVersion

```
uInt8 WLV_GetVersion (uInt8 *VersionHigh, uInt8 *VersionLow);
```

Estando o driver instalado, esta primitiva retorna 1 (*true*) e devolve nos parâmetros *VersionHigh* e *VersionLow*, respectivamente o byte mais significativo e e o byte menos significativo da versão do device driver.

4.4. Primitiva WLV_GetAIO_Caps

```
uInt8 WLV_GetAIO_Caps (uInt16 *nAiChannels, uInt16 *nAoChannels,
                       uInt16 *nAiRange, Array1DSingle **AiRange);
```

Estando o device driver instalado e operando, esta primitiva retorna (1) *true* e informa nos parâmetros de saída os recursos de entrada e saída analógica disponibilizados pelo device driver.

Nos parâmetros *nAiChannels* e *nAoChannels* são retornados respectivamente, o número de entradas analógicas e o número de saídas analógicas da placa.

O valor retornado em *nAiRange* indica o número de faixas de entrada do conversor A/D. No parâmetro *AiRange* é retornado um vetor de 16 posições (apenas 6 posições são usadas) com os valores das faixas de entrada do conversor A/D em volts. Valores negativos indicam faixa de entrada bipolar, por exemplo o valor -10.0 indica que a faixa de entrada correspondente é de -10 a 10 volts. Os valores positivos indicam que a faixa de entrada é unipolar, por exemplo, o valor 5.0 indica que a faixa de entrada é de 0 a 5 volts. O conteúdo do vetor é apenas informativo para o programa aplicativo. Para selecionar uma faixa de entrada, o programa deverá informar o índice correspondente ao vetor. Para a placa CAD12/56 são disponíveis as faixas de entrada do A/D apresentadas na tabela.

Índice	Valor do AiRange	Faixa de Entrada
0	-10.0	-10.0 a 10.0 volts
1	-5.0	-5.0 a 5.0 volts
2	-2.0	-2.0 a 2.0 volts
3	-1.0	-1.0 a 1.0 volts
4	10.0	0 a 10.0 volts
5	5.0	0 a 5.0 volts
6	2.0	0 a 2.0 volts
7	1.0	0 a 1.0 volts

Veja também a primitiva *WLV_GetDIO_Caps*.

4.5. Primitiva WLV_GetDIO_Caps

```
uInt8 WLV_GetDIO_Caps (uInt16 *nDiPorts, uInt16 *nBitsDI,
                       uInt16 *nDoPorts, uInt16 *nBitsDO);
```

Estando o device driver instalado e operando, esta primitiva retorna 1 (*true*) e informa nos parâmetros de saída os recursos de entrada e saída digital disponibilizados pelo device driver.

O número de ports de entrada digital é retornado no parâmetro *nDiPorts* e o número de pontos por port de entrada digital é retornado em *nBitsDI*.

O número de ports de saída digital é retornado no parâmetro *nDoPorts* e o número de pontos por port de saída digital é retornado em *nBitsDO*.

Veja também a primitiva *WLV_GetAIO_Caps*.

4.6. Primitiva WLV_SetAiRange

```
uInt8 WLV_SetAiRange (uInt8 iRange);
```

Esta primitiva permite selecionar a faixa de entrada da placa A/D. Ao ser iniciada a CAD12/56 o driver programa o ganho para a faixa de entrada de ±10 volts. Se desejar uma outra faixa de entrada, o aplicativo deve chamar esta primitiva passando no parâmetro *iRange* o número da faixa de entrada desejada (veja 4.4. Primitiva *WLV_GetAIO_Caps*). A primitiva retorna 1 (true) se foi executada com sucesso. Os erros mais comuns na execução da primitiva são faixa de entrada inválida e driver não instalado.

Veja também as primitivas *WLV_ReadAi* e *WLV_GetAIO_Caps*.

4.7. Primitiva WLV_ReadAi

```
uInt8 WLV_ReadAi (uInt8 Channel, Int16 *Value);
```

Esta primitiva realiza a leitura de um canal de entrada analógica da CAD12/56. O programa aplicativo deve passar no parâmetro *Channel* o número do canal A/D a ser lido. A faixa de entrada a ser utilizada na conversão do canal A/D deve ser previamente programada através da primitiva *WLV_SetAiRange*, caso seja uam faixa diferente da última faixa programada.

No parâmetro de saída *Value* é retornado o valor lido do conversor A/D. O valor lido é representado em complemento de 2 e pode assumir valores de -32768 a 32767. A primitiva retorna 1 (true) se foi executada com sucesso. Os erros mais comuns na execução da primitiva são canal de entrada analógica inválido, erro no conversor A/D e driver não instalado.

Esta primitiva não pode ser chamada durante a aquisição por interrupção.

Veja também as primitivas *WLV_SeAiRange* e *WLV_GetAIO_Caps*.

4.8. Primitiva WLV_WriteAo

```
uInt8 WLV_WriteAo (uInt8 Channel, Int16 Value);
```

Esta primitiva realiza a escrita em canal de saída analógica da CAD12/56. O programa aplicativo deve passar no parâmetro *Channel* o número do canal D/A a ser atualizado. O valor a ser escrito no D/A deve ser passado no parâmetro *Value*, que deve estar representado em complemento de 2 com valores de -32768 a

32767. Esses valores correspondem, respectivamente, aos limites inferior e superior da saída analógica. A primitiva retorna 1 (true) se foi executada com sucesso. Os erros mais comuns na execução da primitiva são canal de saída analógica inválido e driver não instalado.

Os canais D/A 1 e 3 quando utilizados como referência para respectivamente os canais D/A 0 e 2, devem ser programados com o sinal trocado. Por exemplo, para colocar a referência do canal D/A 0 em 5 volts, deve-se programar o valor -16384 no canal D/A 1.

Esta primitiva não pode ser chamada durante a aquisição por interrupção.

Veja também a primitiva *WLV_GetAIO_Caps*.

4.9. Primitiva WLV_ReadDI

```
uInt8 WLV_ReadDI (uInt8 Group, uInt16 *Value);
```

Esta primitiva realiza a leitura de um port de entrada digital da CAD12/56. O programa aplicativo deve passar no parâmetro *Group* o número do port de entrada digital a ser lido. O número de ports de entrada digital disponíveis na placa pode ser consultado através da primitiva *WLV_GetDIO_Caps*. No caso da CAD12/56 é disponível 1 port de entrada digital de 16 bits.

No parâmetro de saída *Value* é retornado o valor lido do port de entrada digital. A primitiva retorna 0 (zero) se foi executada com sucesso. Os erros mais comuns na execução da primitiva são número do port de entrada digital inválido e driver não instalado.

Veja também as primitivas *WLV_WriteDO* e *WLV_GetDIO_Caps*.

4.10. Primitiva WLV_WriteDO

```
uInt8 WLV_WriteDO (uInt8 Group, uInt16 Value);
```

Esta primitiva realiza a escrita em port de saída digital da CAD12/56. O programa aplicativo deve passar no parâmetro *Group* o número do port de saída digital a ser atualizado. O número de ports de saída digital disponíveis na placa pode ser consultado através da primitiva *WLV_GetDIO_Caps*. No caso da CAD12/56 é disponível 1 port de saída digital de 16 bits.

O valor a ser escrito no port de saída digital deve ser passado no parâmetro *Value*. A primitiva retorna 1 (true) se foi executada com sucesso. Os erros mais comuns na execução da primitiva são número do port de saída digital inválido e driver não instalado.

Veja também as primitivas *WLV_ReadDI* e *WLV_GetDIO_Caps*.

4.11. Primitiva WLV_Clear_CM

```
void WLV_Clear_CM (void);
```

Esta primitiva limpa a memória de canais utilizada na aquisição por interrupção. Através da memória de canais, o aplicativo informa ao driver a relação dos canais a serem aquisitados durante a aquisição por interrupção. Para cada canal analógico a ser aquisitado, o aplicativo deverá realizar uma chamada da primitiva *WLV_Insert_AI_CM*. Analogamente, deve-se executar a primitiva *WLV_Insert_DI_CM* para os ports de entrada digital. Antes, porém, o aplicativo deverá limpar a memória de canais através da chamada da primitiva *WLV_Clear_CM*.

Veja também as primitivas *WLV_Insert_AI_CM* e *WLV_Insert_DI_CM*.

4.12. Primitiva WLV_Insert_AI_CM

```
uInt8 WLV_Insert_AI_CM (uInt8 Channel, uInt8 iRange);
```

O programa aplicativo deve realizar chamadas sucessivas desta primitiva para informar os canais de entrada analógica a serem lidos na aquisição por interrupção. A ordem de chamada desta primitiva determina a ordem em que os canais serão lidos. Antes da chamada desta primitiva para especificar o primeiro canal a ser aquisitado, deve-se chamar a primitiva *WLV_Clear_CM* para limpar a memória de canais.

Os parâmetros de entrada *Channel* e *iRange* correspondem respectivamente ao número do canal A/D e o índice da faixa de entrada (veja 4.4. Primitiva *WLV_GetAIO_Caps*). A primitiva retorna 1 (true) se foi executada com sucesso. Os erros mais comuns na execução da primitiva são canal de entrada analógica inválido, faixa de entrada inválida, excedeu a capacidade da memória de canais e driver não instalado.

Veja também as primitivas *WLV_Insert_DI_CM* e *WLV_Clear_CM*.

4.13. Primitiva WLV_Insert_DI_CM

```
uInt8 WLV_Insert_DI_CM (uInt8 Group);
```

O programa aplicativo deve realizar chamadas sucessivas desta primitiva para informar os ports de entrada digital a serem lidos na aquisição por interrupção. A ordem de chamada desta primitiva determina a ordem em que os ports serão lidos.

O parâmetro *Group* corresponde ao número do port de entrada de entrada digital. Os ports de entrada digital são lidos em grupo de 16 bits e no caso da placa CAD12/56, deve-se passar o valor 0 (zero) neste parâmetro para que o driver leia os ports P0 e P1. A primitiva retorna 0 (zero) se foi executada com sucesso. Os erros mais comuns na execução da primitiva são número do port de entrada digital inválido, excedeu a capacidade da memória de canais e driver não instalado.

Veja também as primitivas *WLV_Insert_AI_CM* e *WLV_Clear_CM*.

4.14. Primitiva WLV_AcquisitionSetup

```
uInt8 WLV_AcquisitionSetup (single SampleFreq, uInt32 *ErrorCode);
```

Esta primitiva prepara a aquisição de sinais por interrupção e deve ser executada depois da programação da memória de canais através das funções *WLV_Clear_CM*, *WLV_Insert_AI_CM* e *WLV_Insert_DI_CM*.

A frequência em que os sinais serão amostrados deve ser especificada em hertz no parâmetro *SampleFreq*. No parâmetro de saída *ErrorCode* é retornado o código de erro na programação da aquisição (veja 3.2 Constantes). A primitiva retorna 1 (true) se foi executada com sucesso.

Veja também as primitivas *WLV_Clear_CM*, *WLV_Insert_AI_CM*, *WLV_Insert_DI_CM*, *WLV_StartAcquisition*, *WLV_GetSamples* e *WLV_StopAcquisition*.

4.15. Primitiva WLV_StartAcquisition

```
void WLV_StartAcquisition (void);
```

Esta primitiva inicia a aquisição de sinais por interrupção com os parâmetros programados anteriormente. Ela deve ser executada depois da programação da memória de canais e da preparação da aquisição através das primitivas *WLV_Clear_CM*, *WLV_Insert_AI_CM*, *WLV_Insert_DI_CM* e *WLV_AcquisitionSetup*.

Após a chamada da *WLV_StartAcquisition*, o programa aplicativo tem acesso ao andamento da aquisição através da primitiva *WLC_GetSamples*.

Veja também as primitivas *WLC_Clear_CM*, *WLV_Insert_AI_CM*, *WLV_Insert_DI_CM*, *WLV_AcquisitionSetup*, *WLV_GetSamples* e *WLV_StopAcquisition*.

4.16. Primitiva WLV_StopAcquisition

```
void WLV_StopAcquisition (void);
```

Esta primitiva encerra a aquisição de sinais por interrupção.

Veja também a primitiva *WLV_StartAcquisition*.

4.17. Primitiva WLV_GetSamples

```
uInt8 WLV_GetSamples (uInt32 *ieStatus, uInt32 *iSample, uInt32 *nSampRead,
                      Array2DInt16 *Samples);
```

Através desta função o programa aplicativo tem acesso às amostras aquisitadas pelo driver durante a aquisição de sinais por interrupção.

O driver possui um buffer circular de 64K amostras para a aquisição de sinais. Por exemplo, para a aquisição de 16 canais a 140 Hz por canal, o buffer teria capacidade de armazenar até 29 segundos sem que o programa aplicativo remova os dados do buffer circular.

No entanto, o programa aplicativo normalmente possui um loop de processamento onde é realizada periódicamente a chamada da função *WLV_GetSamples* para a leitura dos dados amostrados. O tamanho do buffer de aquisição do driver é utilizado apenas para que não haja perdas de dados durante processamentos demorados do programa aplicativo. No exemplo, o programa aplicativo pode ficar até 29 segundos sem retirar dados do buffer de aquisição. Depois deste tempo ocorre *overflow* no buffer.

A função retorna no parâmetro de saída *ieStatus* o código de ocorrência de erro durante a aquisição (veja 3.2 Constantes).

O parâmetro de saída *Samples* é um array do LabVIEW com duas dimensões. A primeira dimensão representa o índice da amostragem e a segunda dimensão o índice do canal (ordem na memória de canais). Ou seja, *Samples* é uma tabela em que as linhas são os “instantes” de amostragem e as colunas os canais.

A função remove as amostras do buffer de aquisição do driver e as transfere para o *Samples*. O número de amostras transferidas para o array é limitado pelo tamanho do array e pelo número de amostras disponíveis no buffer de aquisição do driver. O número de amostras por canal transferidos para o array é retornado no parâmetro de saída *nSampRead*. No parâmetro *iSample* é retornado o número de ordem da última amostra transferida para o array. O número de ordem da amostra se refere ao início da aquisição. Em cada chamada da função *WLV_GetSamples* é transferido no máximo um total de 16K amostras.

A função retorna o valor 1 (true) se foi transferido dados para o array.

Veja também a função *WLV_StartAcquisition* e *WLV_StopAcquisition*.