NT4_1232.SYS

DRIVER DA CAD12/*32* PARA WINDOWS NT 4.0

1. INTRODUÇÃO

NT4_1232.SYS é um kernel mode driver que gerencia o acesso à placa A/D CAD12/32 no sistema operacional Windows NT4.0 Workstation.

1.1. Escopo

O objetivo deste documento é apresentar a interface com o driver realizada por módulo em Borland Delphi para Windows NT.

1.2. Aplicabilidade

O driver NT4_1232 deve ser utilizado somente no sistema operacional Windows NT 4.0 ou compatível e com a placa A/D CAD12/32.

2. INSTALANDO O DRIVER NT4_1232

Os arquivos fornecidos para utilização do driver NT4_1232:

NT4_1232.SYS	kernel mode driver para Windows NT 4.0
InstDrv.exe	programa de instalação do driver no registry do Windows NT 4.0
RegIni.exe	programa de registro dos parâmetros do driver no registry do Windows NT 4.0
NT4_1232.ini	arquivo de registro do driver
NT4_1232.DLL	biblioteca DLL de acesso ao driver
NT4api.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

2.1. Instalando o Driver

Siga as seguintes etapas para instalar o driver NT4_1232:

1. A partir da barra de tarefa, execute **Iniciar/Programas/Prompt de comando** para entrar na janela *Prompt de comando* (Janela DOS).
2. Supondo que o Windows NT esteja instalado no diretório C:\WINNT, execute o seguinte comando para copiar o driver para o diretório do Windows.

   ```
   COPY A:\NT4_1232.SYS C:\WINNT\SYSTEM32\DRIVERS
   ```

3. Mude o diretório corrente para A:\:
   ```
   CD A:\
   A:
   ```

4. Execute o seguinte comando para registrar a configuração padrão do driver no *Registry do Windows NT*:
   ```
   REGINI NT4_1232.INI
   ```

5. Execute o seguinte comando para instalar o driver no *Registry do Windows NT*:

   ```
   INSTDRV NT4_1232 C:\WINNT\SYSTEM32\DRIVERS\NT4_1232.SYS
   ```

2.2. Como Alterar a Configuração Padrão do Driver

A configuração padrão do driver *NT4_1232* pode ser alterada através do *Registry do Windows NT*. Os parâmetros configuráveis se referem a:

- Endereço base de I/O da CAD12/*32*
- Canal de interrupção utilizado

Para modificar esses parâmetros, você deverá utilizar o programa *Editor do Registro* do Windows NT. Esse programa se encontra no sub-diretório SYSTEM32 do diretório de instalação do Windows NT. O nome do arquivo executável do *Editor do Registro* é REGEDT32.EXE.

Após entrar no *Editor do Registro*, selecione a janela **HKEY_LOCAL_MACHINE** e abra a pasta **SYSTEM\CurrentControlSet\Services\NT4_1232\Parameters** . Esta pasta possui os seguintes valores correspondentes aos parâmetros de configuração do driver:

Nome do Parâmetro	Parâmetro	Valor padrão
IoBaseAddr	Endereço base de I/O da placa CAD12/32	0x300
Interrupt	Canal de interrupção utilizado pela CAD12/32	5

Para consultar os valores de endereços de I/O que estão sendo utilizados por outros dispositivos instalados no Windows NT, utilize o programa *Diagnóstico do Windows NT*. Para isso execute a partir da barra de tarefas **Iniciar/Ferramentas administrativas/Diagnóstico do Windows NT** . No *Diagnóstico do Windows NT*, consulte a página *Recursos*.

Para informações mais detalhadas dos programas *Editor de Registro* e *Diagnóstico do Windows NT*, consulte documentação da Microsoft.

Após modificar os parâmetros do driver, você deverá reiniciar o Windows NT para que os novos valores passem a ter efeito. Ou se preferir, dê um duplo clique sobre o ícone *Dispositivos* no *Painel de Controle*. Na janela do programa *Dispositivos*, selecione a linha correspondente ao *NT4_1232* e clique sobre o botão *Finalizar* e em seguida sobre o botão *Iniciar*, para reiniciar o driver com os novos parâmetros.

3. INTERFACE DO DRIVER COM O PROGRAMA APLICATIVO

O acesso do programa aplicativo ao driver é disponibilizado através do módulo em Pascal NT4api. Esse módulo deve ser incorporado ao projeto do programa aplicativo em Borland Delphi. Se você estiver utilizando outra linguagem de programação, basta criar um módulo equivalente ao NT4api.pas com as referências à biblioteca NT4_12*32*.DLL. A biblioteca DLL deverá ser copiada para o mesmo diretório do programa aplicativo.

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/*32*. Algumas dessas informações (endereço base padrão e canal 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 ±2.5 volts, canal 12 em ±1.0 volts e o canal 15 em 0-5 volts).

O driver possui um buffer circular de 64K amostras para a aquisição de sinais. Por exemplo, para a aquisição de 4 canais a 200 Hz por canal, o buffer teria capacidade de armazenar até 80 segundos sem que o programa aplicativo remova os dados do buffer circular.

3.2. Constantes

A tabela seguinte apresenta as constantes retornadas pela primitiva *NT4_GetAcquiredData*.

Constante	Valor	Descrição
ieNoError	0	Não ocorreu nenhum erro desde a última chamada da primitiva NT4_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
ieBufSize	102	Buffer passado para a NT4_GetAcquiredData é muito pequeno

A tabela seguinte apresenta as constantes retornadas pela primitiva *NT4_AcquisitionSetup*.

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

4. DESCRIÇÃO DAS PRIMITIVAS DO NT4_12*32*

Neste tópico são descritas as primitivas de acesso ao driver NT4_12*32*. A tabela abaixo lista as primitivas disponibilizadas para o programa aplicativo.

Primitiva	**Descrição**
NT4_Open	Inicia o acesso ao driver
NT4_Close	Finaliza o acesso ao driver
NT4_DriverInfo	Informações gerais do driver
NT4_GetVersion	Informa a versão e revisão do driver
NT4_GetAIO_Caps	Informa os recursos de entrada e saída analógica disponíveis no driver
NT4_GetDIO_Caps	Informa os recursos de entrada e saída digital disponíveis no driver
NT4_SetAiRange	Seleciona faixa de entrada do conversor A/D
NT4_ReadAi	Leitura de canal de entrada analógica
NT4_WriteAo	Não disponível na placa CAD12/32
NT4_ReadDI	Leitura dos ports de entrada digital P0 e P1
NT4_WriteDO	Escrita nos ports de saída digital P0 e P1
NT4_Clear_CM	Limpa a memória de canais da aquisição por interrupção
NT4_Insert_AI_CM	Insere um canal de entrada analógica para aquisição
NT4_Insert_DI_CM	Insere os ports de saída digital para aquisição (P0 e P1)
NT4_AcquisitionSetup	Programa aquisição por interrupção
NT4_StartAcquisition	Inicia a aquisição por interrupção
NT4_StopAcquisition	Finaliza a aquisição por interrupção
NT4_GetAcquiredData	Leitura das amostras do buffer de aquisição por interrupção
NT4_GetAcquiredDataVB	Leitura das amostras do buffer de aquisição por interrupção

As funções da bilioteca DLL são descritas com a sintaxe do Object Pascal e do Visual Basic. A convenção de chamada utilizada pela DLL é **stdcall**.

4.1. Primitiva NT4_Open

Object Pascal:
```
Function NT4_Open: dword;
```

Visual Basic:
```
Declare Function NT4_Open Lib "NT4_1232.DLL" () As Long
```

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 NT, clique sobre Dispositivos. Se o driver NT4_1232 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 NT, clique sobre Dispositivos. Verifique se o driver foi iniciado.	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 CAD12/32 e/ou altere a parametrização do driver.

Veja também a primitiva *NT4_Close*.

4.2. Primitiva NT4_Close

Object Pascal:
```
Procedure NT4_Close;
```

Visual Basic:
```
Declare Sub NT4_Close Lib "NT4_1232.DLL" ()
```

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 *NT4_Open* deve haver uma correspondente chamada da *NT4_Close*.

Veja também a primitiva *NT4_Open*.

4.3. Primitiva NT4_DriverInfo

Object Pascal:
```
Function NT4_DriverInfo (Var DriverInfo: TpDriverInfo): boolean;
```

Visual Basic:
```
Não disponível para Visual Basic. Utilize as primitivas NT4_GetVersion,
NT4_GetAIO_Caps e NT4_GetDIO_Caps.
```

Esta primitiva devolve no parâmetro *DriverInfo* o número da versão do driver, o endereço base de I/O da placa, canal de interrupção utilizado, a frequência de clock do timer para programação da frequência de amostragem, número de entradas analógicas, número de faixas de entrada do conversor A/D e respectivas faixas de entrada, número e ports de entrada digital e número de ports de saída digital. A primitiva retorna *true* se a execução foi realizada com sucesso ou *false* (zero) caso ocorra algum erro na sua execução. A estrutura *TpDriverInfo* possui a seguinte declaração em Object Pascal.

```
Type {--------------------------- Parâmetro de entrada da NT4_DriverInfo --}
  TpAiRange = array [0..15] of single;
  TpDriverInfo = record
    VersionHigh : byte;       { Versão do driver }
    VersionLow  : byte;       { Versão do driver }
    CAD_IoBase  : longint;    { Endereço base da CAD12/32 }
    CAD_IRQ     : longint;    { Canal de interrupção da CAD12/32 }
    CAD_DMA     : longint;    { Não utilizado na CAD12/32 }
    TmrClock    : single;     { Clock do timer (Hz) }
    nAiChannels : smallint;   { Número de entradas analógicas }
    nAiRange    : smallint;   { Número de faixas de entrada do A/D em volts }
    AiRange     : TpAiRange;
                              { Tabela com as faixas de entrada do A/D }
    nAoChannels : smallint;   { Número de saídas analógicas }
    nDiPorts    : smallint;   { Número de ports de entrada digital }
    nBitsDI     : smallint;   { Número de bits por port de entrada digital }
    nDoPorts    : smallint;   { Número de ports de saída digital }
    nBitsDO     : smallint;   { Número de bits por port de saída digital }
  end;
```

O campo *AiRange* da estrutura *TpDriverInfo* é um vetor com os valores das faixas de entrada do conversor A/D em volts. Valores negativos indicam faixa de entrada bipolar, por exemplo o valor -5.0 indica que a faixa de entrada correspondente é de -5 a 5 volts. Os valores positivos indicam que a faixa de entrada é unipolar, por exemplo, o valor 2.5 indica que a faixa de entrada é de 0 a 2.5 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/*32* são disponíveis as faixas de entrada do

A/D apresentadas na tabela.

Índice	Valor do AiRange	Faixa de Entrada
0	-5.0	-5.0 a 5.0 volts
1	-2.5	-2.5 a 2.5 volts
2	-1.0	-1.0 a 1.0 volts
3	-0.5	-0.5 a 0.5 volts
4	5.0	0 a 5.0 volts
5	2.5	0 a 2.5 volts
6	1.0	0 a 1.0 volts
7	0.5	0 a 0.5 volts

4.4. Primitiva NT4_GetVersion

Object Pascal:
```
Function NT4_GetVersion (Var VersionHigh, VersionLow: byte): boolean;
```

Visual Basic:
```
Declare Function NT4_GetVersion Lib "NT4_1232.DLL" _
        (VersionHigh As Byte, VersionLow As Byte) As Byte
```

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

Veja também a primitiva *NT4_DriverInfo*.

4.5. Primitiva NT4_GetAIO_Caps

Object Pascal:
```
Function NT4_GetAIO_Caps (Var TmrClock: single;
                          Var nAiChannels, nAoChannels, nAiRange: smallint;
                          Var AiRange: TpAiRange): boolean;
```

Visual Basic:
```
Declare Function NT4_GetAIO_Caps Lib "NT4_1232.DLL" _
        (TmrClock As Single, nAiChannels As Integer, nAoChannels As Integer, _
         nAiRange As Integer, AiRange As TpAiRange) As Byte
```

Estando o device driver instalado e operando, esta primitiva retorna *true* e informa nos parâmetros de saída os recursos de entrada e saída analógica disponibilizados pelo device driver. Os parâmetros retornados pela primitiva correspondem aos campos de mesmo nome da estrutura *TpDriverInfo* descrita na primitiva *NT4_DriverInfo*.

Veja também as primitivas *NT4_GetDIO_Caps* e *NT4_DriverInfo*.

4.6. Primitiva NT4_GetDIO_Caps

Object Pascal:
```
Function NT4_GetDIO_Caps (Var nDiPorts, nBitsDI: smallint;
                          Var nDoPorts, nBitsDO: smallint): boolean; export;
```
Visual Basic:
```
Declare Function NT4_GetDIO_Caps Lib "NT4_1232.DLL" _
        (nDiPorts As Integer, nBitsDI As Integer, _
         nDoPorts As Integer, nBitsDO As Integer) As Byte
```
Estando o device driver instalado e operando, esta primitiva retorna *true* e informa nos parâmetros de saída os recursos de entrada e saída digital disponibilizados pelo device driver. Os parâmetros retornados pela primitiva correspondem aos campos de mesmo nome da estrutura *TpDriverInfo* descrita na primitiva *NT4_DriverInfo*.

Veja também as primitivas *NT4_GetAIO_Caps* e *NT4_DriverInfo*.

4.7. Primitiva NT4_SetAiRange

Object Pascal:
```
Function NT4_SetAiRange (iRange: byte): dword;
```

Visual Basic:
```
Declare Function NT4_SetAiRange Lib "NT4_1232.DLL" _
        (ByVal iRange As Byte) As Long
```

Esta primitiva permite selecionar a faixa de entrada da placa A/D. Ao ser iniciada a CAD12/*32*, o driver programa o ganho para a faixa de entrada de ±5 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 4.3. Primitiva NT4_DriverInfo). A primitiva retorna 0 (zero) 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 *NT4_ReadAi* e *NT4_DriverInfo*.

4.8. Primitiva NT4_ReadAi

Object Pascal:
```
Function NT4_ReadAi (Channel: byte; Var Value: smallint): dword;
```

Visual Basic:
```
Declare Function NT4_ReadAi Lib "NT4_1232.DLL" _
        (ByVal Channel As Byte, Value As Integer) As Long
```

Esta primitiva realiza a leitura de um canal de entrada analógica da CAD12/*32*. 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 *NT4_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 0 (zero) 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 *NT4_SeAiRange* e *NT4_DriverInfo*.

4.9. Primitiva NT4_WriteAo

Object Pascal:
```
Function NT4_WriteAo (Channel: byte; Value: smallint): dword;
```

Visual Basic:
```
Declare Function NT4_WriteAo Lib "NT4_1232.DLL" _
        (ByVal Channel As Byte, ByVal Value As Integer) As Long
```

Esta primitiva não é disponível para a placa CAD12/*32*.

4.10. Primitiva NT4_ReadDI

Object Pascal:
```
Function NT4_ReadDI (Group: byte; Var Value: word): dword;
```

Visual Basic:
```
Declare Function NT4_ReadDI Lib "NT4_1232.DLL" _
        (ByVal Group As Byte, Value As Integer) As Long
```

Esta primitiva realiza a leitura de um port de entrada digital da CAD12/*32*. 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 no campo *nDiPorts* da estrutura *TpDriverInfo*. No caso da CAD12/*32* 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 *NT4_WriteDO* e *NT4_DriverInfo*.

4.11. Primitiva NT4_WriteDO

Object Pascal:
```
Function NT4_WriteDO Group: byte; Value: word): dword;
```

Visual Basic:
```
Declare Function NT4_WriteDO Lib "NT4_1232.DLL" _
        (ByVal Group As Byte, ByVal Value As Integer) As Long
```

Esta primitiva realiza a escrita em port de saída digital da CAD12/*32*. 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 no campo *nDoPorts* da estrutura *TpDriverInfo*. No caso da CAD12/*32* 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 0 (zero) 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 *NT4_ReadDI* e *NT4_DriverInfo*.

4.12. Primitiva NT4_Clear_CM

Object Pascal:
```
Procedure NT4_Clear_CM;
```

Visual Basic:
```
Declare Sub NT4_Clear_CM Lib "NT4_1232.DLL" ()
```

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 *NT4_Insert_AI_CM*. Analogamente, deve-se executar a primitiva *NT4_Insert_DI_CM* para os ports de entrada digital. Antes, porém, o aplicativo deverá limpar a memória de canais através da chamada da primitiva *NT4_Clear_CM*.

Veja também as primitivas *NT4_Insert_AI_CM* e *NT4_Insert_DI_CM*.

4.13. Primitiva NT4_Insert_AI_CM

Object Pascal:
```
Function NT4_Insert_AI_CM (Channel, iRange: byte): dword;
```

Visual Basic:
```
Declare Function NT4_Insert_AI_CM Lib "NT4_1232.DLL" _
        (ByVal Channel As Byte, ByVal iRange As Byte) As Long
```

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 *NT4_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 4.3. Primitiva NT4_DriverInfo). A primitiva retorna 0 (zero) 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 *NT4_Insert_DI_CM* e *NT4_Clear_CM*.

4.14. Primitiva NT4_Insert_DI_CM

Object Pascal:
```
Function NT4_Insert_DI_CM (Group: byte): dword;
```

Visual Basic:
```
Declare Function NT4_Insert_DI_CM Lib "NT4_1232.DLL" _
        (ByVal Group As Byte) As Long
```

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/*32*, 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 *NT4_Insert_AI_CM* e *NT4_Clear_CM*.

4.15. Primitiva NT4_AcquisitionSetup

Object Pascal:
```
Function NT4_AcquisitionSetup (SampleFreq: single;
                               Var ErrorCode: dword): dword;
```

Visual Basic:
```
Declare Function NT4_AcquisitionSetup Lib "NT4_1232.DLL" _
        (ByVal SampleFreq As Single, ErrorCode As Long) As Long
```

Esta primitiva prepara a aquisição de sinais por interrupção e deve ser executada depois da programação da memória de canais através das funções *NT4_Clear_CM*, *NT4_Insert_AI_CM* e *NT4_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 0 (zero) se foi executada com sucesso.

Veja também as primitivas *NT4_Clear_CM*, *NT4_Insert_AI_CM*, *NT4_Insert_DI_CM*, *NT4_StartAcquisition*, *NT4_GetAcquiredData* e *NT4_StopAcquisition*.

4.16. Primitiva NT4__StartAcquisition

Object Pascal:
```
Procedure NT4_StartAcquisition;
```

Visual Basic:
```
Declare Sub NT4_StartAcquisition Lib "NT4_1232.DLL" ()
```

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 *NT4_Clear_CM*, *NT4_Insert_AI_CM*, *NT4_Insert_DI_CM* e *NT4_AcquisitionSetup*.

Após a chamada da *NT4_StartAcquisition*, o programa aplicativo tem acesso ao andamento da aquisição através da primitiva *NT4_GetAcquiredData*.

Veja também as primitivas *NT4_Clear_CM*, *NT4_Insert_AI_CM*, *NT4_Insert_DI_CM*, *NT4_AcquisitionSetup*, *NT4_GetAcquiredData* e *NT4_StopAcquisition*.

4.17. Primitiva NT4_StopAcquisition

Object Pascal:
```
Procedure NT4_StopAcquisition;
```

Visual Basic:
```
Declare Sub NT4_StopAcquisition Lib "NT4_1232.DLL" ()
```

Esta primitiva encerra a aquisição de sinais por interrupção.

Veja também a primitiva *NT4_StartAcquisition*.

4.18. Primitiva NT4_GetAcquiredData

Object Pascal:
```
Function NT4_GetAcquiredData (Var AcquiredData: TpAcquiredData): dword;
```

Visual Basic:
```
Não disponível para Visual Basic. Utilize a primitiva NT4_GetAcquiredDaraVB.
```

A primitiva *NT4_GetAcquiredData* verifica o status do andamento da aquisição por interrupção e obtém as últimas amostras aquisitadas pelo driver. O status de erro da aquisição é retornado pela primitiva conforme codificação descrita na tabela do item 3.2.

```
Type {-------------------------- Parâmetro de saída da NT4_GetAquiredData --}
  TpAcquiredData = record
    ieStatus: byte;    { Status da aquisição }
    nSamples: dword;   { Número de amostras aquisitadas por canal }
    nSampGot: dword;   { Número de amostras transferidas por canal }
    iLast   : dword;   { Número da última amostra transferida }
    UserBuf : array [0..16383] of smallint;
  end;
```

A primitiva retorna no parâmetro *AcquiredData* o status do andamento da aquisição. O campo *ieStatus* do type *TpAcquiredData* tem o mesmo significado do valor retornado pela primitiva. O campo *nSamples* informa o número de amostras aquisitadas por canal desde o início da aquisição.

Através desta primitiva 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 4 canais a 200 Hz por canal, o buffer teria capacidade de armazenar até 80 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 primitiva *NT4_GetAcquiredData* 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 80 segundos sem retirar dados do buffer de aquisição. Depois deste tempo ocorre *overflow* no buffer.

O campo *UserBuf* de *AcquiredData* é um array que deve ser interpretado como uma matriz. As linhas da matriz representam o índice da amostragem e as colunas os canais (na ordem da memória de canais e por tipo, entradas analógica e depois as entrada digitais).

A primitiva remove as amostras do buffer de aquisição do driver e as transfere para o *UserBuf*. O número de amostras transferidas para o array é limitado pelo tamanho do array (16K amostras) e pelo número de

amostras disponíveis no buffer de aquisição do driver. O número de amostras por canal transferidos para o *UserBuf* é retornado no campo *nSampGot*. No campo *iLast* é retornado o número de ordem da última amostra transferida para o *UserBuf*. O número de ordem da amostra se refere ao início da aquisição.

Veja também as primitivas *NT4_StartAcquisition* e *NT4_StopAcquisition*.

4.19. Primitiva NT4_GetAcquiredDataVB

Object Pascal:
```
Function  NT4_GetAcquiredDataVB (Var ieStatus, nSamples, nSampGot,
                                  iLast: dword; Var UserBuf: TpUserBuf): dword;
```

Visual Basic:
```
Declare Function NT4_GetAcquiredDataVB Lib "NT4_1232.DLL" _
        (ieStatus As Long, nSamples As Long, nSampGot As Long, _
         iLast As Long, UserBuf As TpUserBuf) As Long
```

Esta primitiva é análoga à primitiva *NT4_GetAcquiredData*. Os parâmetros de saída da primitiva correspondem aos campos de mesmo nome da estrutura *TpAcquiredData* descrita na primitiva *NT4_GetAcquiredData*.

Veja também as primitivas *NT4_StartAcquisition*, *NT4_StopAcquisition* e *NT4_GetAcquiredData*.