dx.doi.org/10.7437/NT2236-7640.2011.01.003
Notas Técnicas, v. 1, n. 1, 2011

Adaptação USB Para Módulo de Aquisição de Dados

Ademarlaudo França Barbosa,* Aridio Schiappacassa,† and Herman Pessoa Lima Júnior‡
Centro Brasileiro de Pesquisas Físicas - CBPF, Rua Dr. Xavier Sigaud, 150 – Urca – Rio de Janeiro – RJ – Brasil

Resumo: Neste trabalho é apresentada uma adaptação para comunicação de dados via interface de Barramento Serial Universal ('Universal Serial Bus', USB), para aplicação em sistemas de aquisição de dados.
São apresentados os circuitos necessários para ativação do dispositivo responsável pela interface e o método de instalação em sistema operacional Scientific Linux. Também são descritos programas simples em linguagem C, para leitura e escrita de dados e teste da interface.

Keywords: Instrumentação; Interface USB; Linguagem C; Linux

1. INTRODUÇÃO

Desenvolvido no Laboratório de Sistemas de Detecção (LSD) do CBPF, o Módulo de Processamento de Dados (MPD, mostrado na Figura 1.1), é utilizado como base na realização da aquisição de dados em vários experimentos físicos do próprio Laboratório e de parceiros do CBPF. O MPD é constituído por quatro conversores de sinal analógico para digital de doze *bits* e um contador de tempo digital binário, com resolução de 120 picosegundos, gerenciados por um dispositivo lógico programável *Field Programmable Gate Array* (FPGA).

Originalmente, a comunicação entre o MPD e um computador dava-se através da porta de comunicação paralela. A partir do momento em que os fabricantes de dispositivos para computadores deixaram de fornecer as tradicionais interfaces de comunicação, principalmente em computadores portáteis, surgiu a necessidade de modificar-se o MPD para uso com a interface USB. Uma boa opção em função da relativa simplicidade, taxa de comunicação, robustez e popularidade.

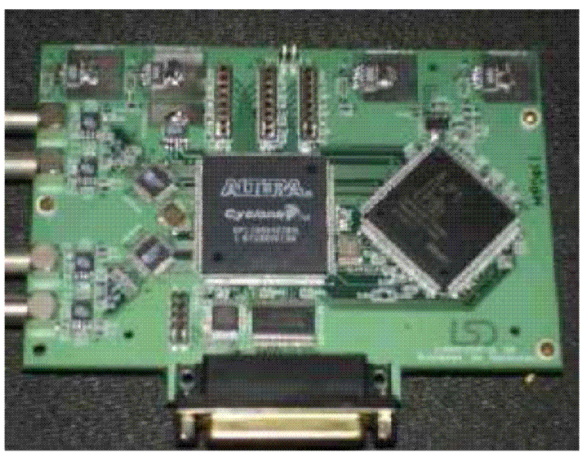

Figura 1.1 - MPD com porta paralela (projeto original)

As primeiras investidas em busca de um controlador para comunicação USB levaram a impasses de direitos comerciais sobre o *software*, e desvantajosos em termos financeiros. Os fabricantes de interfaces USB, geralmente atrelam a utilização dos programas de acionamento do circuito integrado, pelo computador, à compra de conjuntos de desenvolvimento a um custo algumas vezes exorbitante para um simples teste de funcionalidade. Para que se produza um equipamento com direitos exclusivos de divulgação, inclusive do logotipo de compatibilidade USB, é necessário que se faça registro do mesmo na organização normatizadora do protocolo USB [1], a um custo de milhares de dólares. Alguns logotipos difundidos para a interface USB são mostrados na Figura 1.2.

Figura 1.2 – Logotipos de compatibilidade USB

Por essas razões optou-se pelo uso de um circuito integrado que está cada vez mais popular, principalmente por ser de baixo custo e direito de uso livre, o circuito FT245, fabricado pela empresa *Future Technology Devices International Ltd. (FTDI Chip)*. A permissão de uso dos programas de acionamento está condicionada à divulgação do nome do proprietário dos mesmos, neste caso, a FTDI Chip [2].

Os produtos desenvolvidos a partir deste *chip* podem ter seu nome, número de série e grupo que o desenvolveu, gravados. Conectado a uma memória EEPROM 93C46B, o *chip* FT245 permite que se registrem essas informações utilizando o programa FT_PROG [3], fornecido gratuitamente pelo fabricante, ou através da própria interface, com o computador. A importância disto não está somente no fato de ter-se um registro eletrônico de quem o produziu, mas também na identificação do produto pelo computador. Desse modo, o programa que o utilizar poderá fazer referência, para acesso, através do nome do produto e/ou de seu número de série. Uma imagem do dispositivo que implementa a interface para o módulo MPD é mostrada na Figura 1.3, juntamente com uma foto do circuito FT245.

Uma vez desenvolvida a primeira adaptação, o MPD_USB, o Laboratório de Sistemas de Detecção desenvolveu mais dois outros produtos, mostrados na Figura 1.4, utilizando a mesma interface USB:

O SPRO, uma evolução do módulo MPD_USB e o NDAQ. Este último, desenvolvido para o padrão VME, além da comunicação USB, será utilizado na aquisição de dados no projeto Neutrinos Angra, no Brasil, e no experimento Double Chooz [4], sediado na França.

*Electronic address: laudo@cbpf.br
†Electronic address: aridio@cbpf.br
‡Electronic address: hlima@cbpf.br)

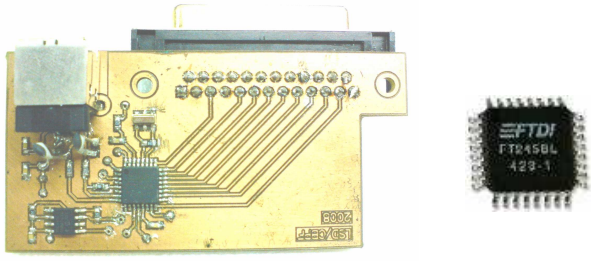

Figura 1.3 – Circuito adaptador USB baseado no FT245

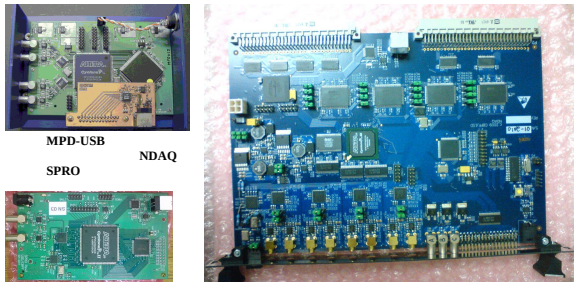

Figura 1.4 – MPD-USB, SPRO e NDAQ

2. A COMUNICAÇÃO VIA PORTA USB

A comunicação de dados via USB se dá por tráfego de dados em série que, classicamente, comparada à comunicação paralela, sempre foi classificada como de baixa velocidade.

A partir da década de 70, com o aperfeiçoamento da comunicação de dados via rede de computadores pela empresa *General Motors*, o uso da tecnologia de circuitos balanceados e a integração de circuitos eletrônicos atingindo velocidades impossíveis com circuitos discretos, levaram a um enorme avanço na comunicação de dados.

Após isso, na década de 80, o uso de protocolo de comunicação no tráfego entre controladores e discos rígidos no padrão *Small Computer System Interface* (SCSI) aumentou bastante a velocidade de comunicação. Além do uso de protocolo na troca de informação, o SCSI utilizava linhas com impedâncias casadas para diminuir a reflexão na transmissão. Na década de 90 o padrão *Wide Ultra* SCSI, com dados trafegando com níveis de tensão de 3,3 Volts em linhas balanceadas, alcançou velocidades ainda maiores, como 40 *megabytes* por segundo, superando o problema da taxa de variação de tensão nos circuitos eletrônicos (*Slew Rate*).

Hoje, o padrão de comunicação USB se beneficia de todos os avanços tecnológicos produzidos nas décadas anteriores, com protocolo de conexão a múltiplos dispositivos, sinais balanceados, linhas com impedâncias casadas e 3,3 Volts de nível de sinal. Possui também a capacidade *hot pluggable*, onde um dispositivo pode ser conectado ao computador, com este já ligado, sendo reconhecido automaticamente.

2.1. Características gerais

Existem duas formas de acesso ao *chip* FT245, que utiliza o padrão USB 1.1. A forma mais frequentemente utilizada para acesso à porta cria um enumerador de portas de comunicação seriais virtuais. Para cada tomada de conexão USB presente no computador o sistema operacional define um nome diferente para designar a porta de comunicação a ela associada. Observa-se que o sistema reserva as portas séries de nome COM1 a COM4 para circuitos reais, iniciando a designação das portas virtuais a partir de COM5. O inconveniente deste método de acesso é o limite de velocidade de dados de um e meio *megabit* por segundo (1,5Mbps), padrão compatível com *Low Speed*. A outra forma de acesso ao FT245, na qual este trabalho está baseado, acessa a porta USB através de um programa (*driver*), para acionamento do *chip*, com velocidade de comunicação de até doze *megabits* por segundo (12Mbps), padrão denominado *Full Speed*. Devido ao avanço tecnológico de velocidade, temos agora a nova nomenclatura *Hi Speed*, para o padrão USB 2.0, até quatrocentos *megabits* por segundo, e USB 3.0, *Super Speed*, até quatro *gigabits* por segundo. Na Tabela 2.1 estão listados os padrões de velocidade [5][6] na comunicação de dados via USB.

Padrão	Nome	Velocidade
USB 1	LOW SPEED	Até 1,5 Mbps
USB1.1	FULL SPEED	Até 12 Mbps
USB 2	HIGH SPEED	Até 480 Mbps
USB 3	SUPER SPEED	Até 4800 Mbps

Tabela 2.1 – Padrões de velocidade USB

2.2. Características elétricas da porta USB.

Atualmente, o cabo de conexão do computador até a interface USB tem enorme relevância, pois além do tráfego de dados é responsável pelo fornecimento de energia para pequenos dispositivos eletrônicos. Partindo dessa necessidade, há agora a preocupação quanto ao comprimento do cabo devida à queda de tensão provocada em sua linha de +5 Volts. Paralelamente, o aumento de velocidade no tráfego de dados, nos padrões mais modernos, acarreta maior sensibilidade a ruídos induzidos eletromagneticamente, exigindo mais cuidado na blindagem do cabo e com os fios de dados trançados.

2.2.1. Os conectores USB.

À parte das características elétricas, os conectores e tomadas de dados USB têm características físicas que os distingue por faixa de velocidade além de distinguir a conexão na tomada de dados do dispositivo ou do controlador. Desta forma, as tomadas de dados dos computadores são compatíveis com o padrão USB 1.0/2.0 usando conectores modelo "A", enquanto a tomada de dados, no painel de uma impressora, por exemplo, cumpre o padrão USB 1.0 com conector B, conforme mostra a Figura 2.1. Nota-se também no caso dos dispositivos no padrão USB 2.0, como armazenadores de dados em massa, ainda que compatíveis com o padrão USB 1.0, o conector do tipo "A".

Figura 2.1 – Conectores e cabos USB

2.2.2. Cabo de conexões USB

Ao se conectar um dispositivo USB ao computador, este reconhece a conexão pelo fato de existir, no dispositivo, um resistor conectado a uma das linhas de dados. Para cada tipo de conexão, em função da velocidade do dispositivo, há uma forma de conectar o resistor [7]. O modo *Full Speed* é mostrado na figura 2.2.

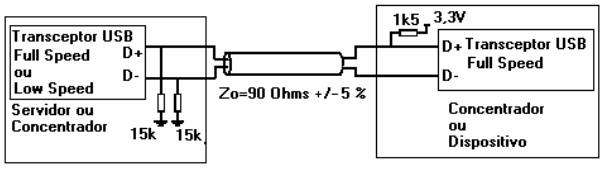

Figura 2.2 - Conexão USB *Full Speed*

As cores dos fios são padronizadas para cada sinal e tensão, conforme a Tabela 2.2.

Pino	**Sinal**	**Cor do fio**
1	VCC (+ 5V)	Vermelho
2	D-	Branco
3	D+	Verde
4	GND (Massa)	Preto

Tabela 2.2 – Padrões de cores nos condutores dos cabos USB

2.2.3. Alimentação e capacidade de corrente

O servidor com conexão USB fornece tensão de alimentação de +5 Volts ao dispositivo USB, com limitação de corrente a 100 mA em princípio, podendo ser configurado por *software* para fornecer até 500 mA. O sinal +VCC tem padrão da família lógica TTL, + 4,75 Volts a +5,25 Volts, com valor nominal de +5 Volts.

Os dados que trafegam entre o servidor (*host*) e o dispositivo, sinais D- e D+, trabalham com tensão de 3,3 Volts e são balanceados à massa. O balanceamento garante alta imunidade a ruídos introduzidos no cabo de comunicação durante o tráfego de dados.

2.2.4. Blindagem do cabo de dados

Os cabos de dados devem ter a blindagem conectada apenas ao controlador. O pino de massa do cabo USB é ligado à massa do dispositivo.

Os cabos blindados apresentam condutor de dados com bitola 28 AWG **[NOTA]** trançados e os cabos não blindados apresentam fios não trançados para a transmissão de dados.

Os fios de alimentação têm bitolas de 28 a 20 AWG, conforme o comprimento do cabo de conexão.

2.2.5. Comprimento do cabo de dados

Em face à preocupação quanto à queda de tensão ao longo dos fios, já que, dependendo do dispositivo, podem circular correntes de até 500 miliamperes, para cada comprimento é necessário alterar a bitola dos fios de alimentação. Deve-se ter em mente também que, por ter as linhas de dados balanceadas em relação à massa, não circula corrente de sinal no fio de massa, exceto por algum efeito de reflexão na impedância de 90 ohms, normalmente desprezível.

A Tabela 2.3 relaciona as distâncias do controlador ao dispositivo em função da bitola do condutor de alimentação usado no cabo de comunicação:

Diâmetro AWG	28	26	24	22	20
Comprimento máximo	0,8 m	1,3 m	2 m	3,3 m	5 m

Tabela 2.3 – Comprimento de cabos USB em função da bitola do fio de alimentação

[NOTA] *American Wire Gauge*– AWG –Bitola de Fio Americana - padrão utilizada nos EUA e Canadá desde 1857, não deve ser confundido com o padrão *Brown & Sharpe* - B&S. embora tenham calibres muito próximos.

3. O *CHIP* DE INTERFACE USB FT245

Situado entre o computador e o circuito eletrônico, entre os quais se dará a transferência dos dados, o circuito integrado FT245 é responsável por todo o tratamento protocolar durante a comunicação. Uma vez que o cabo de conexão é ligado da interface ao computador, é ele quem providencia o envio da informação que diz ao computador quem está sendo conectado. No momento inicial da conexão o circuito integrado envia sua identificação ao computador para que este coloque em funcionamento o programa de acionamento correspondente. Também é ele que ordena o tráfego entre o computador e o circuito eletrônico. Basicamente ele sinaliza, através de seus terminais RxF# e TxE#, o envio de um dado pelo computador ou quando o computador está pronto para receber um dado.

3.1. Diagrama

A Figura 3.1 apresenta o diagrama do circuito da interface eletrônica usando o circuito integrado FT245. No circuito está identificado o barramento bidirecional paralelo dos dados para conexão ao circuito eletrônico que se comunicará com o computador.

O tráfego de dados é sinalizado pelos sinais TxE# e RxF#, e o comando de leitura ou escrita é feito pelos sinais RD# e WR#.

Note-se o uso de capacitores de 47 picofarads e resistores de 27 ohms nas linhas de entrada balanceada do sinal USB

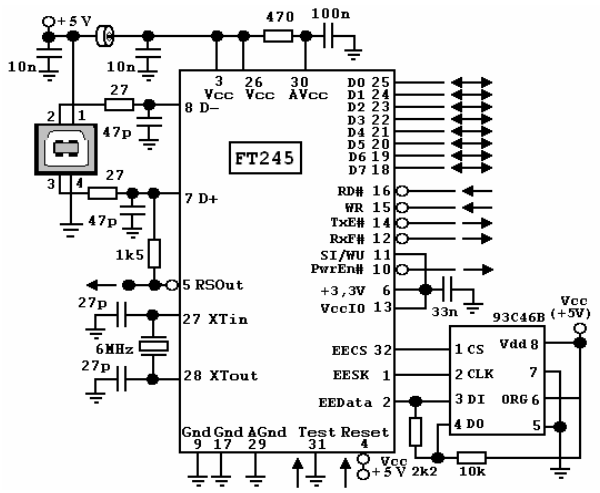

Figura 3.1 – Circuito da Interface USB usando o FT245

para redução da sensibilidade ao ruído apresentada por algumas conexões USB 2.0. Seu uso está documentado na nota de aplicação [8] AN232B-06_11, seção 3.1.

A geração de *clock* de 48 MHz para sincronismo da comunicação é conseguida a partir de um ressoador piezoelétrico de 6 MHz, aplicado internamente a um *Phase Lock Loop – PLL*.

O circuito integrado contém um regulador de tensão de 3,3 volts interno para uso no barramento, dispensando o uso de reguladores externos, para aplicações até o limite de 5 miliamperes.

A memória 93C46 é responsável por armazenar a identificação do produto e é gravada através do programa FT_PROG [3], fornecido gratuitamente pelo fabricante.

3.2. Principais sinais do FT245

Para acessar os dados da interface FT245, dois sinais, RxF# e TxE#, são disponibilizados para que o circuito eletrônico proceda à leitura do dado vindo do computador ou envie o dado para o computador, acionando respectivamente, RD e WR. A sequência a ser obedecida é descrita resumidamente abaixo, estando documentada em notas de aplicação, disponíveis via *internet* [9].

Os sinais disponibilizados são:

(pino 12) / RXF# (Saída):

Quando em nível baixo, pelo menos um *byte* está presente na memória *fifo* de 128 *bytes* e está pronta para ser lida ao comando de RD#. RXF# vai para nível alto quando o *buffer* de recepção está vazio.

(pino 14) / TXE# (Saída):

Quando em nível alto, a fifo de 384 *bytes* do *buffer* de transmissão está cheia ou ocupada armazenando o último *byte* escrito. Não tentar escrever dados no *buffer* de transmissão quando TXE# estiver em nível alto.

(pino 15) / WR (Entrada):

Sinal aplicado ao *chip*, em cuja borda de descida o *byte* presente no barramento de dados é colocado na memória *fifo* do *buffer* de transmissão. O dado escrito é enviado ao computador controlador (*host*) dentro do limite de intervalo de tempo (*time-out*), normalmente de 16 milisegundos, podendo ser programado entre 1 e 255 milisegundos.

(pino 16) / RD# (Entrada):

Quando colocado em nível baixo, as linhas de dados têm o *byte* corrente da memória *fifo* do *buffer* de recepção. Quando volta ao nível alto, as linhas de dados voltam ao estado de alta impedância (*three state*).

O circuito integrado contém internamente uma memória do tipo *first in first out* (FIFO) de 128 *bytes* para recepção dos dados vindos do computador e outra de 384 *bytes* para os dados a serem enviados ao computador.

A presença do circuito da interface USB é percebida pelo computador no momento em que é conectada à porta USB. Um resistor de *pull-up* com valor de 1,5 k ohms conectado ao pino de dados D+ e a +Vcc, fornecido pelo pino RSOut, informa ao computador a presença da interface.

4. INSTALAÇÃO DOS PROGRAMAS DE ACIONAMENTO

O procedimento a seguir foi testado no sistema operacional *Scientif Linux 5 – Red Hat*.

Conforme orientação do fabricante, a instalação dos programas é feita mediante cópia do arquivo obtido pela *internet*, além da configuração do sistema LINUX.

O arquivo deve ser copiado para a pasta */usr/local/lib*. Desta forma, se o arquivo usado for, por exemplo, o *libftd2xx.so.1.0.4*, será criado o *link* virtual com nome genérico *libftd2xx.so* apontando para o arquivo */usr/local/lib/libftd2xx.so.1.0.4*. Também na pasta */usr//lib* será criado um *link* virtual apontando para o arquivo */usr/local/lib/ libftd2xx.so.1.0.4*. O objetivo é fazer com que atualizações futuras do arquivo de acionamento sejam simplificadas.

A seguir são apresentadas orientações sobre como proceder. Para tanto, pode ser usado o terminal de texto ou a interface gráfica que estiver ativada: ***gnome***, ***kde*** ou ***ice***.

4.1. Instalação do arquivo *libftd2xx*

Acesse o Linux como usuário *root*, ou insira o comando *su* e a senha. Copie o arquivo *libftd2xx.so.1.0.4* para a pasta */usr/local/lib*.

[cp libftd2xx.so.1.0.4 /usr/local/lib]

Mude para a pasta */usr/local/lib*.

[cd /usr/local/lib]

Faça um *link* simbólico para esse arquivo dentro da própria pasta.

[ln -s libftd2xx.so.1.0.4 libftd2xx.so]

Mude para a pasta /usr/lib e faça um *link* simbólico para o arquivo *libftd2xx.so.1.0.4* da pasta */usr/local/lib*.

[cd /usr/lib]

[ln -s /usr/local/lib/libftd2xx.so.1.0.4 libftd2xx.so]

Carregue as novas configurações de bibliotecas com o comando:

[ldconfig -p |grep libftdi]

Este comando atualiza o *cache* de bibliotecas do Linux, sem necessidade de reiniciar o computador, baseado no arquivo /ect/ld.so.conf.

Para verificar se o circuito da interface está sendo detectado pelo computador, use o comando de listagem dos dispositivos USB.
[lsusb].

Caso ocorram problemas de reconhecimento do *chip* FT245 durante a execução de programas, é possível que outras bibliotecas estejam instaladas, como **usbsio** e **ftdi_sio**, que podem ser instaladas automaticamente pelo Linux na presença do circuito conectado.

Para a remoção permanente dos módulos deve ser editado o arquivo */etc/modprobe.d/blacklist*, acrescentando as seguintes linhas:

blacklist ftdi_sio

e

blacklist usbserial

Para remoção de **ftdi_sio** e **usbserial** através do terminal de texto do Linux, que devem ser descarregados se estiverem associados ao seu dispositivo, use os seguintes comandos:

Lista os módulos atualmente carregados, [lsmod]:
[lsmod |grep usbserial]
[lsmod |grep ftdi_sio]

Para descarregar módulos executados no Linux, [rmmod]:
[http://linux.about.com/cs/linux101/g/rmmod.htm acessado em 09Mai2011]
[rmmod ftdi_sio]
[rmmod usbserial]

Para retirada dos *drivers*, como super usuário, faça:
[rmmod ftdi_sio e rmmod usbserial]
modprobe é usado para carregar um módulo
[http://linux.about.com/od/commands/l/blcmdl8_modprob.htm acessado em 09Mai2011]
[modprobe usbserial]
[modprobe ftdi_sio]

Também é possível haver problemas de reconhecimento do circuito por diferença no PID –*Product Identifier* e VID – *Vendor Identifier*.

Neste caso, recomenda-se que sejam usados PID de 0x6006 e VID de 0x0403.
Para gravar no circuito integrado faz-se uso do programa FT_PROG [3], fornecido gratuitamente pelo fabricante FTDI Chip.

5. TESTE DE FUNCIONAMENTO

A seguir são fornecidas listagens de programas para teste da interface FT245, baseados em linguagem C, compilados em *gcc* no terminal de texto do Linux.

Para compilação e execução basta seguir as orientações dadas nos comentários no inicio da listagem.

5.1. Programa que lista quantos dispositivos FTDI estão conectados

```
##############################################
//CBPF 2011 Aridio Schiappacassa
//gcc sob sistema Scientific LINUX (CERN)
//com libftd2xx.so.0.4.16 instalado
//FTDInumdevices.cxx
//manter na mesma pasta os arquivos ftd2xx.h e Wintypes.h
//para compilar digite:
//g++ -o FTDInumdevices.exe FTDInumdevices.cxx -lftd2xx -Wno-deprecated
//para executar digite:
//./FTDInumdevices.exe
##############################################

#include "ftd2xx.h"

#include <stdio.h>
#include <stdlib.h>
#include <iostream.h>

int main()
 {

   DWORD numdevices;

   FT_ListDevices

   (&numdevices,NULL,FT_LIST_NUMBER_ONLY);

cout<<"\n\nNumero de dispositivos conectados e: "<<numdevices<<"\n\n"<<endl;

 return 0;
 }

//##############################################
```

5.2. Programa que lista o número de identificação dos dispositivos FTDI que estão conectados

```
##############################################
//CBPF 2011 Aridio Schiappacassa
//gcc sob sistema Scientific LINUX (CERN)
//com libftd2xx.so.0.4.16 instalado
//FTDIserialnumber.cxx
//manter na mesma pasta os arquivos ftd2xx.h e wintypes.h
```

```
//para compilar digite:
//g++ -o FTDIserialnumber.exe FTDIserialnumber.cxx
// -lftd2xx –Wno -deprecated
//para executar digite:
//./FTDIserialnumber.exe
//MOSTRA O NUMERO DE SERIE DO DISPOSITIVO FTDI NUMERO 0
//#############################################

#include ”ftd2xx.h”

#include <stdio.h>
#include <stdlib.h>
#include <iostream.h>

int main()
 {

 DWORD devicenum=0;//numero do dispositivo conectado

 DWORD numdevices;//numero de dispositivos conectados
 FT_ListDevices
(&numdevices,NULL,FT_LIST_NUMBER_ONLY);


 if(numdevices!=0)//existe dispositivo conectado
 {
   char buffer[6];

   DWORD numserie;//caracteres do numero de serie

   FT_ListDevices
((PVOID)devicenum,buffer,FT_LIST_BY_INDEX);

  cout<<”\n\nNumero de serie do dispositivo 0 e: ”<<buffer<<”\n\n”<<endl;
 }

 else
 {
cout<<”\n\nenhum dispositivo conectado
\n\n”<<endl;
 }

 return 0;
}
//#############################################
```

6. EXEMPLO DE APLICAÇÃO

A interface USB usando o circuito integrado FT245 foi aplicada ao Módulo de Processamento de Dados do LSD-CBPF, controlado por uma FPGA EP1C6Q240C8 [10], programada em linguagem VHDL, e configurada através da ferramenta QUARTUS 9 [11].

Os programas de acesso aos dados foram compilados em *gcc*, usando como interface gráfica a plataforma ROOT CERN [12], em sistema *Scientific Linux*.

A Figura 6.1 mostra o diagrama simplificado do conjunto de testes de um Tubo Fotomultiplicador (*Photomultiplier Tube*, PMT), para aquisição de dados de seu sinal de saída. O resultado de uma aquisição de sinal é mostrado na Figura 6.2, capturada do monitor de vídeo do computador, com o programa em modo pausa.

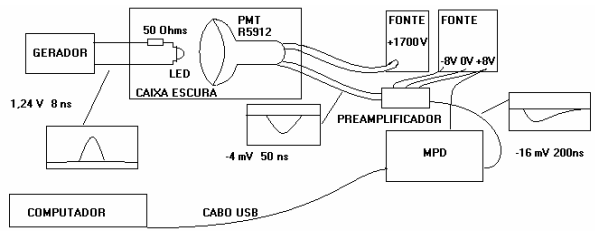

Figura 6.1 Conjunto de testes do PMT usando a interface USB

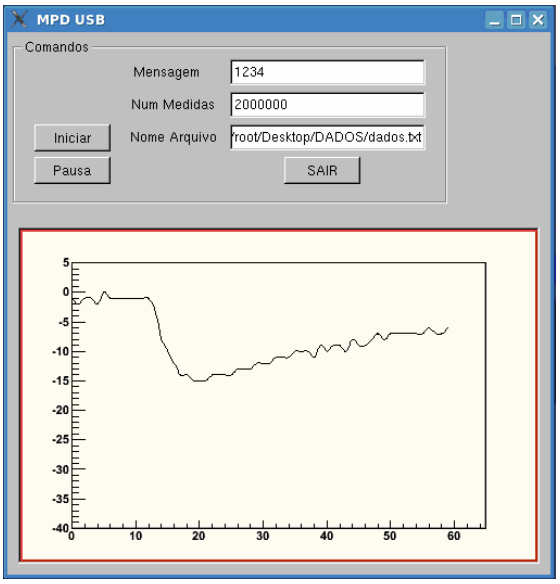

Figura 6.2 Imagem de uma aquisição com o programa em modo pausa

Neste conjunto, um diodo emissor de luz (*Light Emitting Diode*, LED), associado em série com um resistor de 50 ohms, recebe pulsos elétricos com duração de 8 ns, provenientes de um gerador. Os pulsos de curta duração fazem o LED acender com quantidade de luz extremamente baixa. Esta mínima quantidade de luz tem o propósito de excitar o fotocatodo do PMT, fazendo-o emitir, na maioria das vezes, somente um elétron. Os elétrons emitidos são acelerados pelo dinodos do PMT até chegarem ao anodo principal, energizado por um potencial de aproximadamente 2000 volts. Durante o trajeto, as cargas elétricas chocam-se com os dinodos, provocando o deslocamento de mais cargas elétricas, resultando em uma amplificação. O pulso elétrico obtido ao final é proporcional à quantidade de luz excitando o PMT, que, em níveis discretos, corresponde às cargas elétricas extraídas do fotocatodo. O sinal proveniente do PMT passa por

um preamplificador integrador, que produz uma tensão proporcional à carga total do pulso gerado pelo PMT. O conversor analógico-digital do MPD converte o sinal em números binários. Comandados pela FPGA, os números binários são levados à interface FT245 e finalmente enviados ao computador. No computador, através do programa de captura, as medidas das amplitudes de carga do PMT são armazenadas em arquivo para análise posterior. Como resultado, o histograma das aquisições mostra que o ponto médio da estatística dos dados corresponde ao ganho do PMT.

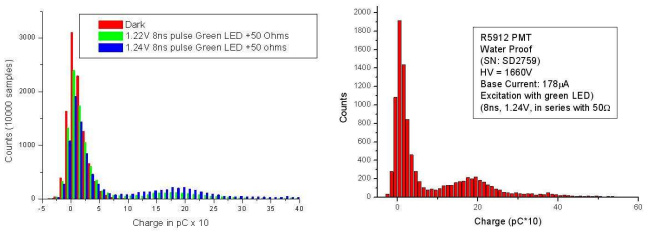

Figura 7.1 Histograma de 10.000 eventos do PMT R5912 ASSY WP s.n. SD2759

7. CONCLUSÃO

Na Figura 7.1 é mostrado o resultado da aplicação do MPD_USB, na forma do histograma de cargas, processado no programa QtiPlot [13], de 10.000 aquisições dos sinais do PMT R5912 WP s.n. SD2759. Com valor médio de contagens de 1,85pC, temos como resultado o ganho correspondente a 1,16x10^7.

As informações e os resultados apresentados neste trabalho contextualizam um procedimento relativamente simples para implementação de interfaces que utilizem o padrão USB. O procedimento foi efetivamente adotado na realização de módulos de processamento de dados baseados em FPGAs, e testado em aplicações práticas, em particular na observação de fotoelétrons individuais em um PMT.

[1] http://www.usb.org
Acessado em 10 de Maio de 2011.
[2] http://www.ftdichip.com
Acessado em 13 de Maio de 2011.
[3] http://www.ftdichip.com/Support/Utilities.htm#FT_Prog
Acessado em 13 de Maio de 2011.
[4] http://doublechooz.in2p3.fr
Acessado em 13 de Maio de 2011.
[5] http://www.usb.org/developers/devclass_docs/CabConn20.pdf
(*Low Speed, Full Speed e High Speed.* Pag. 3)
Acessado em 09 de Maio de 2011.
[6] http://www.usb.org/press
/USB_CES_2010_Media_Alert_FINAL.pdf
(*Super Speed Press release*)
Acessado em 09 de Maio de 2011.
[7] http://www.usb.org/developers/docs/resistor_ecn.pdf
Acessado em 09 de Maio de 2011.
[8] http://www.ftdichip.com/Support/Documents/AppNotes
/AN232B-06_11.pdf
Seção 3.1, acessada em 09 de Maio de 2011.
[9] http:// www.ftdichip.com/Documents/datasheets/modules
/ds_um245r.pdf
Acessado em 09 de Maio de 2011.
[10] http://www.altera.com
Acessado em 09 de Maio de 2011.
[11] https://www.altera.com/download/dnl-index.jsp
Acessado em 09 de Maio de 2011.
[12] http://root.cern.ch
Acessado em 09 de Maio de 201
[13] http://soft.proindependent.com/qtiplot.html
Acessado em 09 de Maio de 2011.