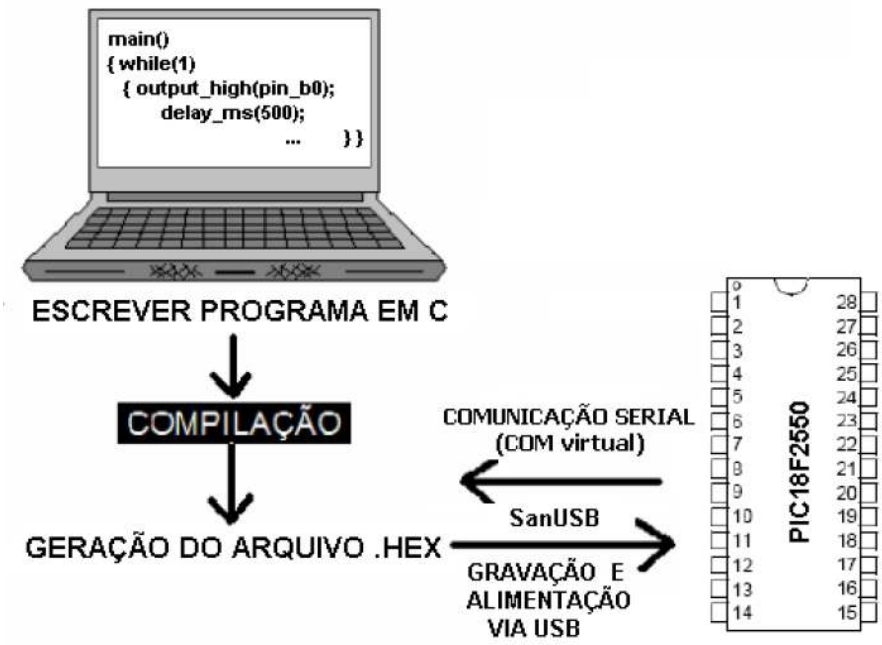
**SanUSB**

O sistema de desenvolvimento *SanUSB* é uma ferramenta composta de *software* e *hardware* básico da família PIC18Fxx5x com interface USB. Esta ferramenta livre se mostra eficiente no desenvolvimento rápido de projetos reais, pois não há necessidade de remover o microcontrolador para a atualização do firmware. Além disso, esta ferramenta se mostra eficaz no ensino e na difusão de microcontroladores, bem como em projetos de eletrônica e informática, pois todos os usuários podem desenvolver projetos reais no ambiente de ensino ou na própria residência sem a necessidade de um equipamento para gravação de microcontroladores. Além disso, o software de gravação de microcontroladores USB é multiplataforma, pois é executável no Windows, Linux e Mac OSX e, também *plug and play*, ou seja, é reconhecido automaticamente pelos sistemas operacionais sem a necessidade de instalar nenhum *driver*. Dessa forma, ela é capaz de suprimir:

1- Um equipamento ou circuito específico para gravação de um programa no microcontrolador;
2- conversor TTL - RS-232 para comunicação serial bidirecional, emulado via USB pelo protocolo CDC, que permite também a depuração do programa através da impressão via USB das variáveis do *firmware*;
*3*- fonte de alimentação, já que a alimentação do PIC provém da porta USB do PC. *É importante salientar que cargas indutivas como motores de passo ou com corrente acima de 400mA devem ser alimentadas por uma fonte de alimentação externa.*
4- Conversor analógico-digital (AD) externo, tendo em vista que ele dispõe internamente de **10** ADs de 10 bits;
5- *software* de simulação, considerando que a simulação do programa e do *hardware* podem ser feitas de forma rápida e eficaz no próprio circuito de desenvolvimento ou com um *protoboard* auxiliar.

Além de todas estas vantagens, os *laptops* e alguns computadores atuais não apresentam mais interface de comunicação paralela e nem serial EIA/RS-232, somente USB.

Como pode ser visto, esta ferramenta possibilita que a compilação, a gravação e a simulação real de um programa, como também a comunicação serial através da emulação de uma porta COM virtual, possam ser feitos de forma rápida e eficaz a partir do momento em o microcontrolador esteja conectado diretamente a um computador via USB.




Utilizando esta ferramenta, estudantes foram três vezes consecutivas campeões da Competição de Robótica do IFCE (2007, 2008 e 2009) na categoria Localização, campeões da Feira Brasileira de Ciências e Engenharia (FEBRACE09) da USP em São Paulo na Categoria Engenharia (2009), como também obtiveram Prêmio de Inovação em Aplicação Tecnológica na *Feria Explora* 2009 em Medelin na Colômbia e foram Campeões na Categoria *Supranivel* do *Foro Internacional de Ciencia e Ingeniería* 2010 no Chile.

## GRAVAÇÃO COM O *SanUSB*

A transferência de programas para os microcontroladores é normalmente efetuada através de um hardware de gravação específico. Através desta ferramenta, é possível efetuar a descarga de programas para o microcontrolador diretamente de uma porta USB de qualquer PC.

Para que todas essas funcionalidades sejam possíveis, é necessário gravar, anteriormente e somente uma vez, com um gravador específico para PIC, o gerenciador de gravação pela USB Gerenciador.hex disponível na pasta completa da ferramenta no link abaixo, onde também é possível baixar periodicamente as atualizações dessa ferramenta e a inclusão de novos programas:

http://www.4shared.com/file/sIZwBP4r/100727SanUSB.html

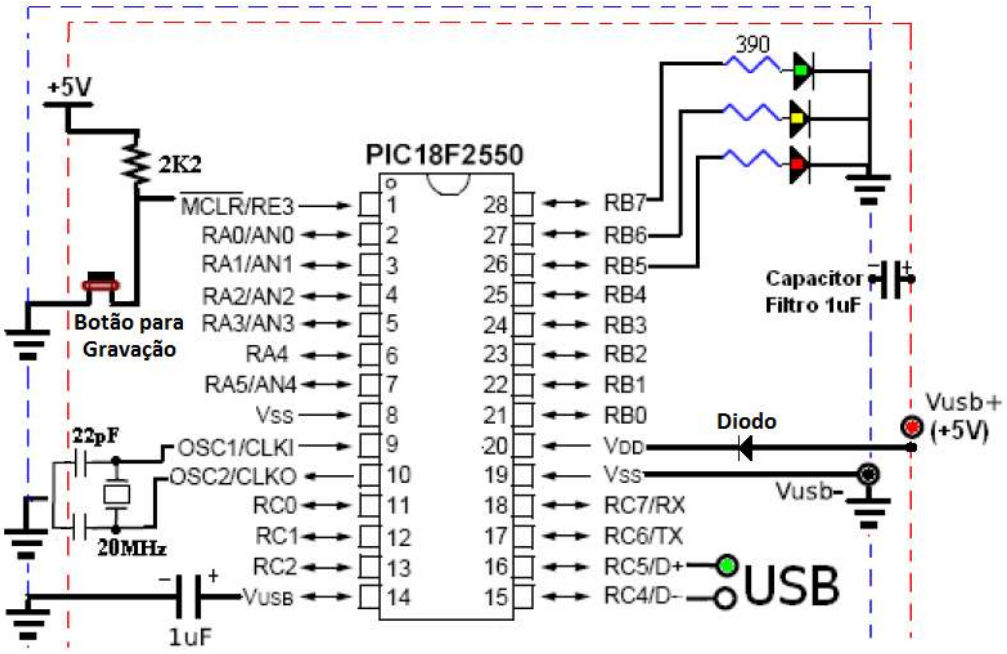
Caso o computador ainda não o tenha o aplicativo Java JRE ou SDK instalado para suporte a programas executáveis desenvolvidos em Java, baixe a Versão Windows disponível em: http://www.4shared.com/file/WKDhQwZK/jre-6u21-windows-i586-s.html ou através do link: http://www.java.com/pt_BR/download/manual.jsp.

Para que os programas em C possam ser gravados no microcontrolador via USB, é necessário compilá-los, ou seja, transformá-los em linguagem de máquina hexadecimal. Existem diversos compiladores que podem ser utilizados por esta ferramenta, entre eles o SDCC, o C18, o Hi-Tech e o CCS. Devido à didática das funções e bibliotecas USB disponíveis para emulação serial, diversos periféricos e *multitasking*, um dos compiladores utilizados com bom rendimento, além do C18, com exemplos de aplicação disponíveis na pasta de desenvolvimento, é o CCS na versão 3.245. Esta versão funcional com bibliotecas de suporte a USB pode ser obtida através do link:

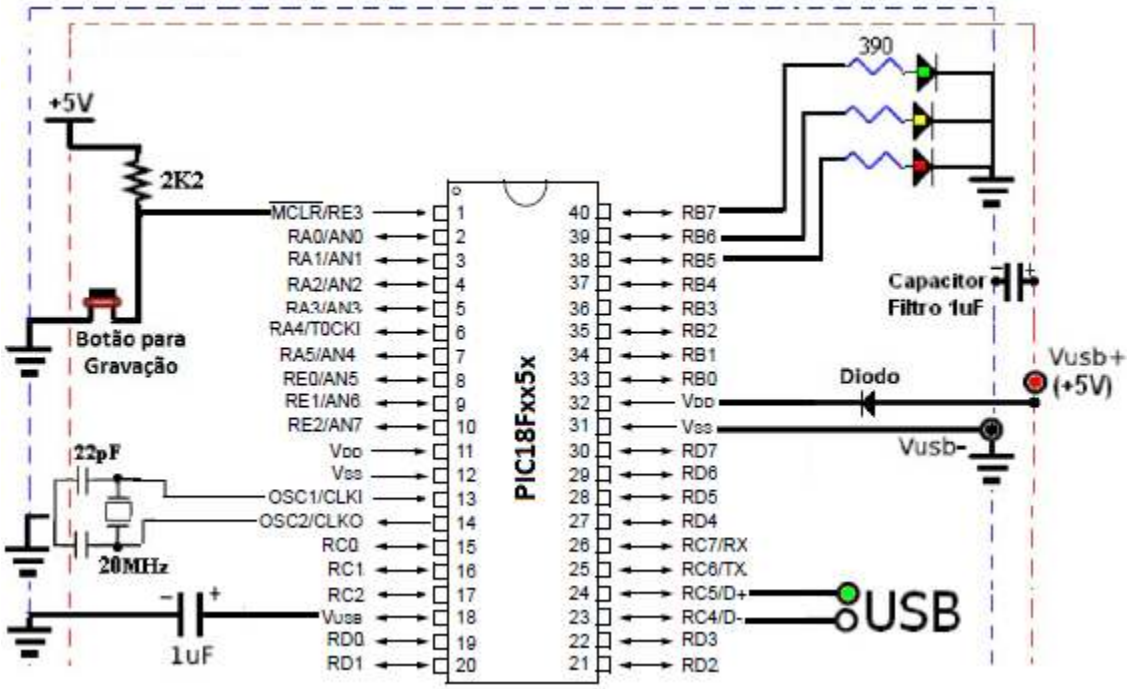
http://www.4shared.com/file/Mo6sQJs2/100511Compilador.html .

As versões 4 deste compilador apresentam *bugs* em funções e aplicações, embora tenham sido testadas algumas versões e funcionaram satisfatoriamente até a versão 4.084. Neste caso, é recomendado criar, para cada firmware (programa a ser compilado), um novo *source file*.

Caso grave no microcontrolador o novo gerenciador de gravação pela USB Gerenciador.hex, não esqueça de colar o novo arquivo cabeçalho SanUSB.h dentro da pasta ExemploseBibliotecasCCS localizada na pasta instalada do compilador (C:\Arquivos de programas\PICC\Drivers ). A representação básica do circuito *SanUSB* montado em *protoboard* é mostrada a seguir:




Para um microcontrolador de 40 pinos, o circuito é mostrado abaixo:




Os componentes básicos do circuito são:

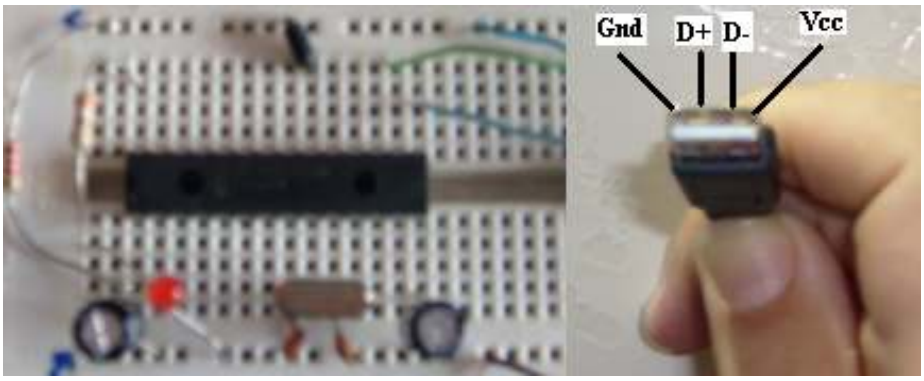
- 1 microcontrolador da família PIC USB (18F2550, 18F2455, 18F4550, etc.);
- 1 cristal de 20MHz;
- 2 capacitores de 22pF;
- 2 capacitores de 1uF (um no pino 14 Vusb e outro entre o +5V e o Gnd ) ;
- 3 leds e 3 resistores de 390 (só é necessário um led com resistor no pino B7);
- 1 resistor de 2k2 e um botão ou fio para gravação no pino 1;

- 1 diodo qualquer entre o +5V e o o pino Vdd;
- 1 Cabo USB qualquer.

Note que, ***este sistema multiplataforma pode ser implementado também em qualquer placa de desenvolvimento de microcontroladores PIC com interface USB***, pois utiliza o botão de *reset*, no pino 1, como botão de gravação via USB. Ao conectar o cabo USB e alimentar o microcontrolador, com o pino 1 no Gnd (0V), através do botão ou de um simples fio, o microcontrolador entra em Estado para Gravação via USB (*led* no pino B7 aceso) e que, após o *reset* com o pino 1 no Vcc (+5V através do resistor fixo de 2K2 sem o *jump*), entra em Estado para Operação do programa aplicativo (*firmware*) que foi compilado.

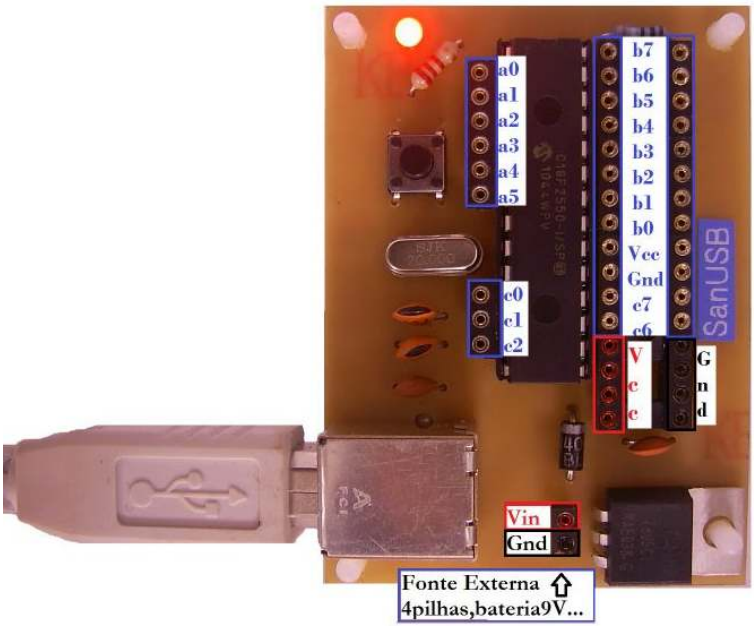
O cabo USB apresenta normalmente quatro fios, que são conectados ao circuito do microcontrolador nos pontos mostrados na figura acima, onde normalmente, o fio Vcc (+5V) do cabo USB é vermelho, o Gnd (Vusb-) é marrom ou preto, o D+ é azul ou verde e o D- é amarelo ou branco. Note que a fonte de alimentação do microcontrolador nos pinos 19 e 20 e dos barramentos vermelho (+5V) e azul (Gnd) do circuito provem da própria porta USB do computador. Para ligar o cabo USB no circuito é possível cortá-lo e conectá-lo direto no *protoboard*, com fios rígidos soldados, como também é possível conectar sem cortá-lo, em um *protoboard* ou numa placa de circuito impresso, utilizando um conector USB fêmea. O diodo de proteção colocado no pino 20 entre o Vcc da USB e a alimentação do microcontrolador serve para proteger contra corrente reversa caso a tensão da porta USB esteja polarizada de forma inversa.

A figura abaixo mostra a ferramenta SanUSB montada em *protoboard* seguindo o circuito anterior e a posição de cada terminal no conector USB a ser ligado no PC. Cada terminal é conectado diretamente nos pinos do microcontrolador pelos quatro fios correspondentes do cado USB.

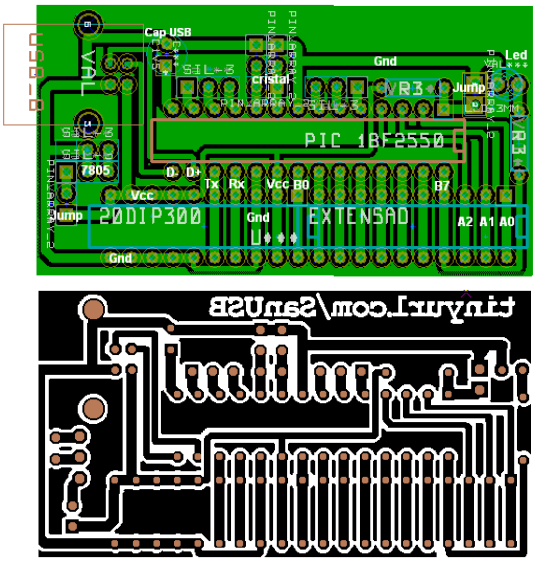



É importante salientar que, para o perfeito funcionamento da gravação via USB, o circuito desta ferramenta deve conter um capacitor de filtro entre 0,1uf e 1uF na alimentação que vem da USB, ou seja, colocado entre os pinos 20 (+5V) e 19 (Gnd).

Caso o sistema microcontrolado seja embarcado como, por exemplo, um robô, um sistema de aquisição de dados ou um controle de acesso, ele necessita de uma fonte de alimentação externa, que pode ser uma bateria comum de 9V ou um carregador de celular. A figura abaixo mostra o PCB, disponível nos Arquivos do Grupo SanUSB, e o circuito para esta ferramenta com entrada para fonte de alimentação externa. Para quem deseja obter o sistema pronto para um aprendizado mais rápido, é possível também encomendar placas de circuito impresso da ferramenta SanUSB, como a foto da placa abaixo, entrando em contato com o grupo SanUSB através do e-mail: sanusb_laese@yahoo.com.br .



Se preferir confeccionar a placa, é possível também imprimir o PCB (em preto) abaixo, em folha apropriada, corroer, furar (pontos marrons) e soldar os componentes. Mais detalhes em: http://www.4shared.com/get/ithqLbiq/FazendoPCBtermico.html ou através do video disponível em: http://www.youtube.com/watch?v=8NhNsNw5BfU.




Para obter vários programas-fonte e vídeos deste sistema livre de gravação, comunicação e alimentação via USB, basta se cadastrar no grupo de acesso livre www.tinyurl.com/SanUSB e clicar no item Arquivos.

Durante a programação do microcontrolador basta inserir, no inicio do programa em C, a biblioteca cabeçalho SanUSB (#include <SanUSB.h>) contida dentro da pasta ExemploseBibliotecasCCS e que você já adicionou dentro da *Drivers* localizada na pasta instalada do compilador ( C:\Arquivos de programas\PICC\Drivers ). Essa biblioteca contém instruções do PIC18F2550 para o sistema operacional, configurações de fusíveis e habilitação do sistema *Dual Clock*, ou seja, oscilador RC interno de 4 MHz para CPU e cristal oscilador externo de 20 MHz para gerar a frequência de 48MHz da comunicação USB, através de *prescaler* multiplicador de freqüência.

Como a freqüência do oscilador interno é de 4 MHz, cada incremento dos temporizadores corresponde a um microssegundo. O programa exemplo1 abaixo comuta um *led* conectado no pino B7 a cada 0,5 segundo.

```
#include <SanUSB.h>

void main()
{
clock_int_4MHz();//Função necessária para habilitar o dual clock (48MHz para USB e 4MHz para CPU)

while (1)
  {
  output_toggle(pin_B7); // comuta Led na função principal
  delay_ms(500);
  }
}
```

O programa pisca3 abaixo pisca três *leds* conectados nos pinos B5, B6 e B7.

```
#include <SanUSB.h>

main(){
clock_int_4MHz();//Função necessária para habilitar o dual clock (48MHz para USB e 4MHz para CPU)

while (1)
        {
        output_high(pin_B5); // Pisca Led na função principal
        delay_ms(500);
        output_low(pin_B5);
        output_high(pin_B6);
        delay_ms(500);
        output_low(pin_B6);
        output_high(pin_B7);
        delay_ms(500);
        output_low(pin_B7);
        }
}
```

Os arquivos compilados .hex assim como os firmwares estão disponíveis em http://www.4shared.com/file/sIZwBP4r/100727SanUSB.html.

## GRAVANDO O MICROCONTROLADOR VIA USB NO WINDOWS

Para executar a gravação com a ferramenta *SanUSB*, é importante seguir os seguintes passos:

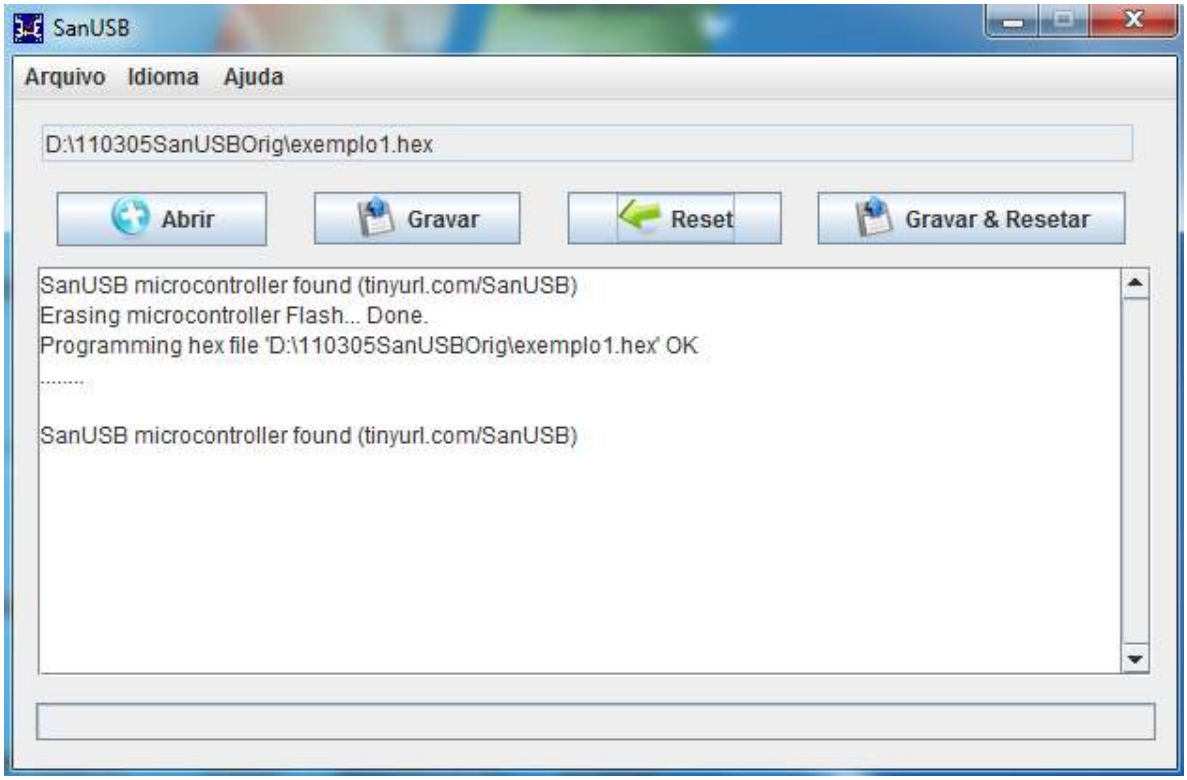
1. Baixe o a pasta da ferramenta de desenvolvimento SanUSB, para um diretório raiz C ou D, obtida no link http://www.4shared.com/file/sIZwBP4r/100727SanUSB.html.

2. Grave no microcontrolador, somente uma vez, com um gravador específico para PIC ou com o circuito simples de gravação descrito no final deste tutorial, o novo gerenciador de gravação pela USB Gerenciador.hex disponível na pasta Gerenciador, compatível com os sistemas operacionais Windows e Linux.

3. Pressione o botão ou conecte o *jump* de gravação do pino 1 no Gnd para a transferência de programa do PC para o microcontrolador.

4. Conecte o cabo USB, entre o PIC e o PC, e solte o botão ou retire o *jump*. Se o circuito SanUSB estiver correto acenderá o *led* do pino B7.

5. Caso o computador ainda não o tenha o aplicativo Java JRE ou SDK instalado para suporte a programas executáveis desenvolvidos em Java, baixe a Versão Windows disponível em: http://www.4shared.com/file/WKDhQwZK/jre-6u21-windows-i586-s.html ou através do link: http://www.java.com/pt_BR/download/manual.jsp e execute o aplicativo **SanUSB** da pasta SanUSBwinPlugandPlay. Surgirá a seguinte tela:

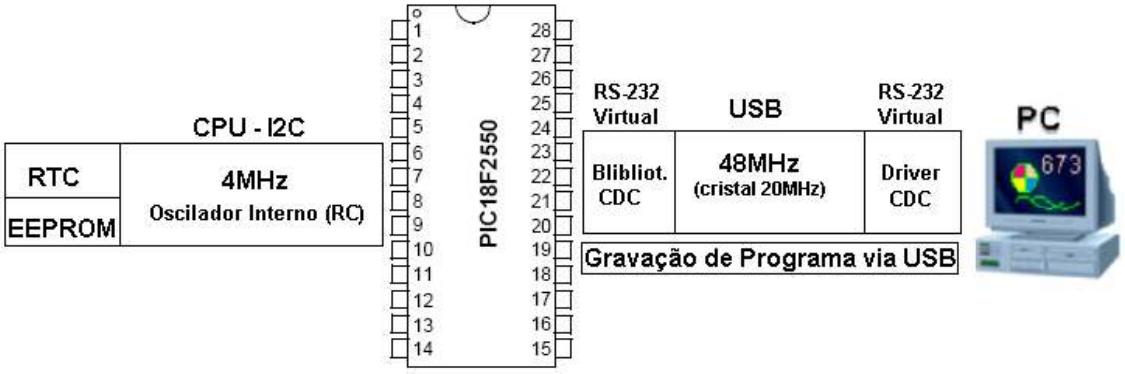


6. Clique em *Abrir* e escolha o programa *.hex* que deseja gravar, como por exemplo, o programa compilado *exemplo1.hex* da pasta ExemploseBibliotecasSanUSB e clique em *Gravar*. Este programa pisca o *led* conectado no pino B7;

7. Após a gravação do programa, lembre-se de soltar o botão ou retirar o *jump* do pino de gravação e clique em Resetar. Pronto o programa estará em operação. Para programar novamente, repita os passos anteriores a partir do passo 3.

## SISTEMA DUAL CLOCK

Devido à incompatibilidade entre as freqüências necessárias para a gravação e emulação serial via USB e a freqüência padrão utilizada pela CPU, temporizadores e interface $I^2C$, esta ferramenta adota o princípio *Dual Clock*, ou seja, utiliza duas fontes de *clock*, uma para o canal USB de 48MHz, proveniente do cristal oscilador externo de 20MHz multiplicada por um *prescaler* interno, e outra para o CPU de 4 MHz, proveniente do oscilador RC interno de 4 MHz, como é ilustrado na figura abaixo.




Esse princípio de *clock* paralelo, realizado pela instrução *clock_int_4MHz()*, permite que um dado digitado no teclado do computador, trafegue para o microcontrolador em 48 MHz via USB, depois para periféricos como um relógio RTC ou para a memória EEPROM em 4 MHz via $I^2C$ e vice-versa.

## SanUSB CDC – EMULAÇÃO DE COMUNICAÇÃO SERIAL NO WINDOWS

Neste tópico é mostrado um método de comunicação serial bidirecional através do canal USB do PIC18F2550. Uma das formas mais simples, é através do protocolo

*Communications Devices Class* (CDC), que emula uma porta COM RS-232 virtual, através do canal USB 2.0. Dessa forma, é possível se comunicar com caracteres ASCII via USB através de qualquer software monitor serial RS-232 como o HyperTerminal, o SIOW do CCS® Compiler ou o ambiente de programação Delphi®. O driver CDC instalado no PC e o programa aplicativo gravado no PIC, com a biblioteca CDC (#include <usb_san_cdc.h>), são os responsáveis por esta emulação da porta RS-232 virtual através da USB.

A biblioteca CDC para o programa.c do microcontrolador está dentro da pasta de exemplos, a qual deve estar na mesma pasta onde está o programa.c a ser compilado para a emulação da comunicação serial RS-232. Além disso, o programa.c deve inserir a biblioteca usb_san_cdc.h, como mostra a o exemplo de leitura e escrita em um buffer da EEPROM interna do microcontrolador. As funções CDC mais utilizadas contidas na biblioteca usb_san_cdc.h para comunicação com a COM virtual são:

- **usb_cdc_putc() –** o microcontrolador envia caracteres ASCII emulados via USB.

  Ex.: printf(**usb_cdc_putc**, "\r\nEndereco para escrever: ");

- **usb_cdc_getc()** – retém um caractere ASCII emulado pela USB.

  Ex.: dado = **usb_cdc_getc();** //retém um caractere na variável dado

- **gethex_usb()** – retém um número hexadecimal digitado no teclado.

  Ex.: valor = **gethex_usb();**//retém um número hexadecimal na variável valor

- **usb_cdc_kbhit( )** – Avisa com TRUE (1) se acabou de chegar um novo caractere no buffer de recepção USB do PIC.

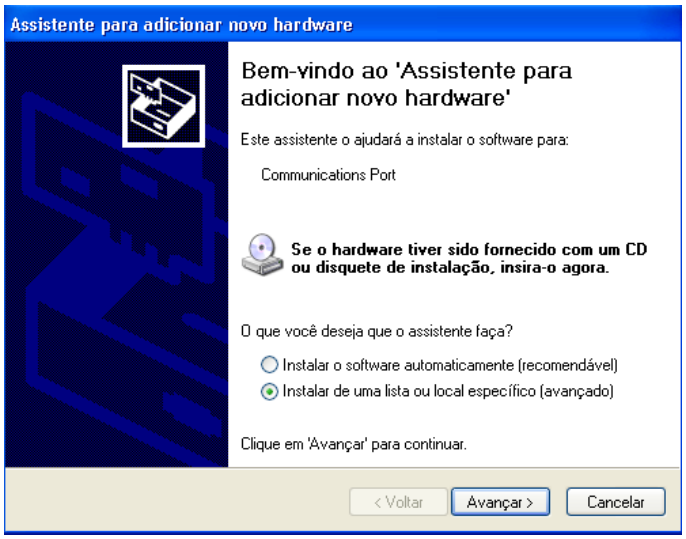
  Ex.: **if (usb_cdc_kbhit()) {dado = usb_cdc_getc();}**

O exemplo abaixo mostra a leitura e escrita em um buffer da EEPROM interna do microcontrolador com emulação da serial através da USB:

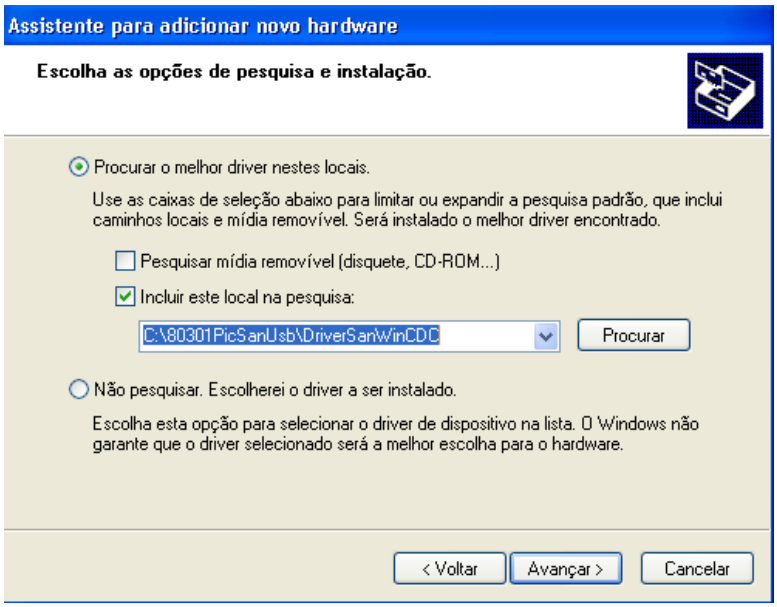
```
#include <SanUSB.h>

#include <usb_san_cdc.h>// Biblioteca para comunicação serial

BYTE i, j, endereco, valor;
boolean led;

main() {
clock_int_4MHz();
  usb_cdc_init(); // Inicializa o protocolo CDC
  usb_init();    // Inicializa o protocolo USB
  usb_task();  // Une o periférico com a usb do PC

  output_high(pin_b7);  // Sinaliza comunicação USB Ok

while (1) {
      printf(usb_cdc_putc, "\r\n\nEEPROM:\r\n"); // Display contém os primeiros 64 bytes em hex
      for(i=0; i<=3; ++i) {
        for(j=0; j<=15; ++j) {
          printf(usb_cdc_putc, "%2x ", read_eeprom( i*16+j ) );
        }
        printf(usb_cdc_putc, "\n\r");
      }

      printf(usb_cdc_putc, "\r\nEndereco para escrever: ");
      endereco = gethex_usb();
      printf(usb_cdc_putc, "\r\nNovo valor: ");
      valor = gethex_usb();

      write_eeprom( endereco, valor );
led = !led; // inverte o led de teste
output_bit (pin_b7,led);
    }
}
```

Após gravação de um programa que utilize comunicação serial CDC no microcontrolador pelo SanUSB e resetar o microcontrolador, vá, se for o **Windows 7**, em propriedades do sistema -> Configurações avançadas do sistema -> Hardware -> Gerenciador de dispositivos e clique com botão direito no driver CDC do microcontrolador e atualizar Driver, apontando para a pasta DriverCDCwinSerial.

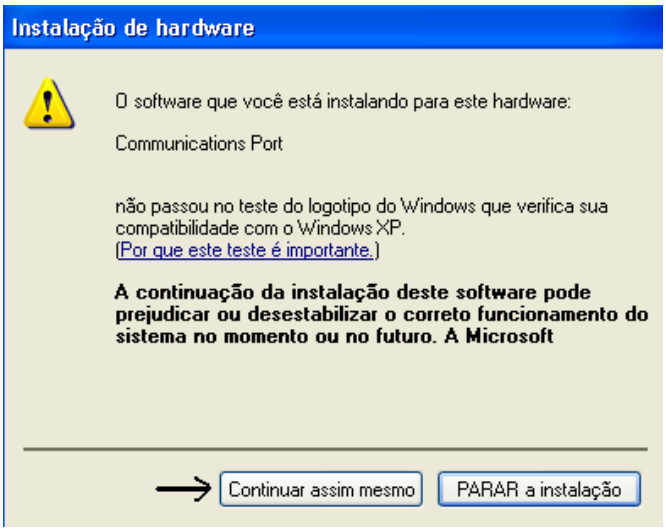
No Windows XP, após a gravação de um programa que utilize comunicação serial CDC no microcontrolador pelo SanUSB e resetar o microcontrolador, o sistema vai pedir a instalação do driver CDC (somente na primeira vez).



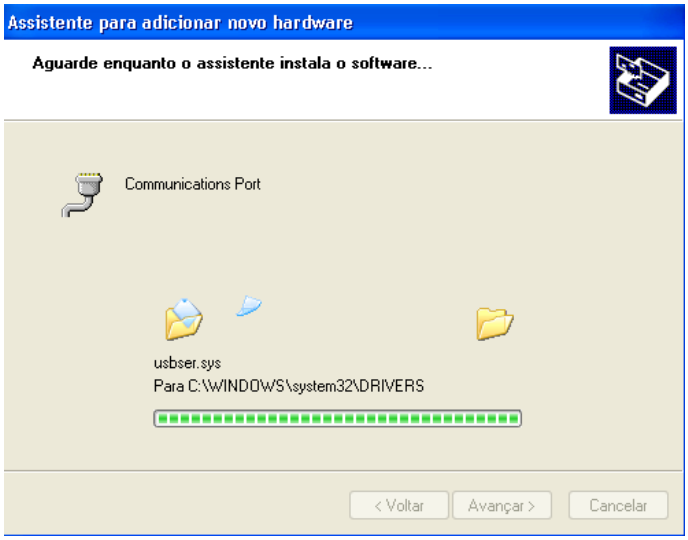

Escolha a opção *Instalar de uma lista ou local especifico (avançado)*. Após Avançar, selecione a *opção Incluir este local na pesquisa* e selecione a pasta DriverSanWinCDC, onde está o driver CDC.



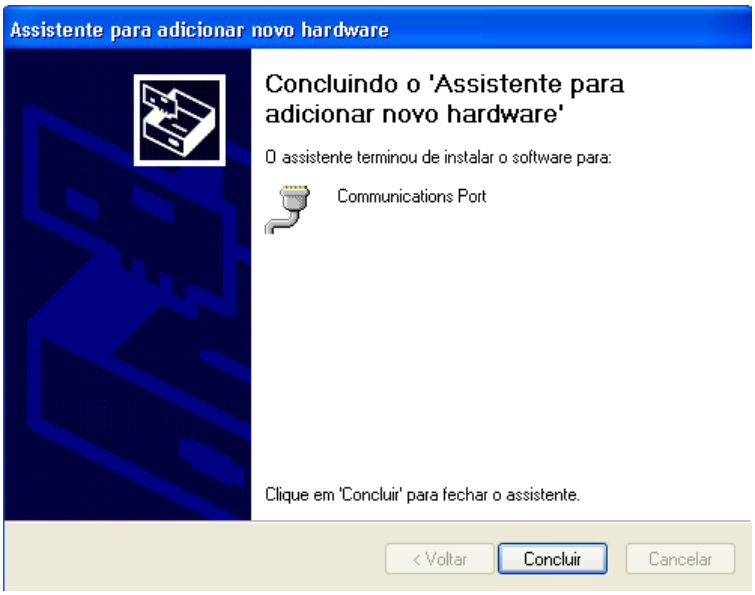

Após Avançar, clique em *Continuar assim mesmo*.



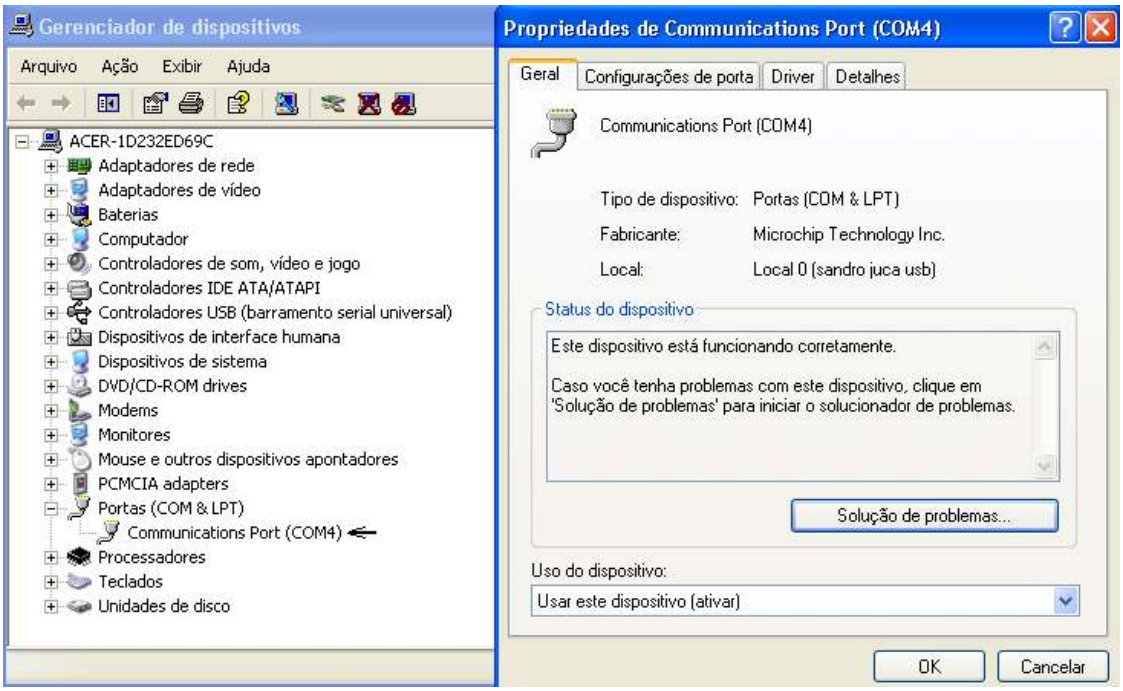

Aguarde enquanto o Driver CDC é instalado no Windows.



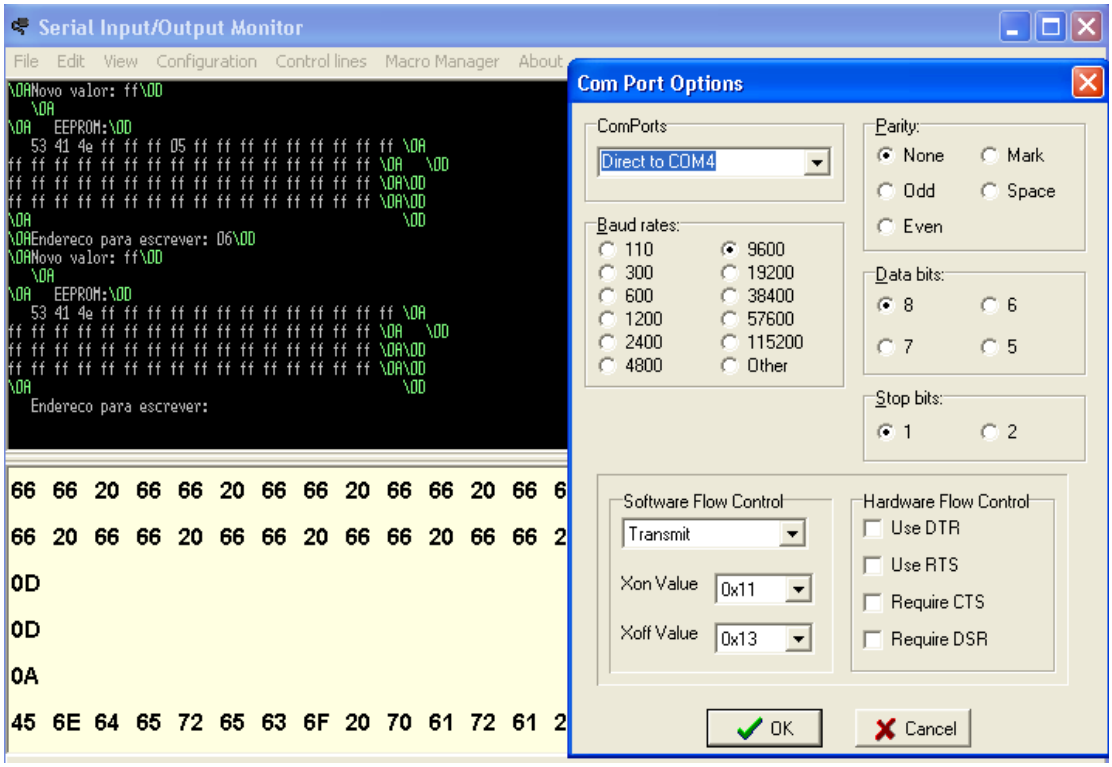

Clique em *Concluir* para terminar a instalação.




Vá em painel de controle -> sistema -> Hardware -> Gerenciador de dispositivos -> Portas (COM & LPT) e confira qual é a porta COM virtual instalada.



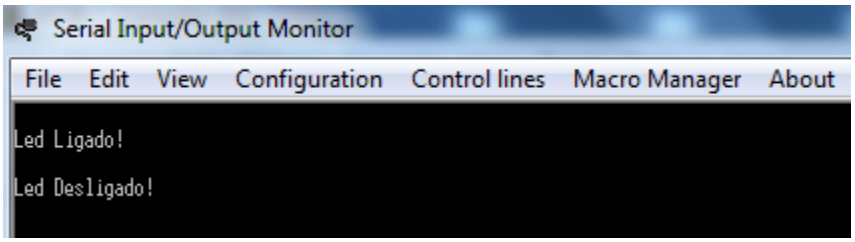

Abrindo qualquer programa monitor de porta serial RS-232, como o SIOW do CCS ou o Java-SanUSB, direcionando para a COM virtual instalada (COM3,COM4,COM5,etc.). No CCS clique em *Tools -> Serial port Monitor -> configuration -> set port options* para que o computador entre em contato com o PIC através da emulação serial via USB.




Para utilizar uma função que necessite de atendimento imediato quando um caractere for digitado como, por exemplo o caractere L ou D, é necessário inserir no firmware do microcontrolador a condição para verificar de forma constante e reter o caractere emulado que chegou pela USB ***if (usb_cdc_kbhit( ))*** **{dado=usb_cdc_getc();}** no laço infinito da função principal. O comando ***(usb_cdc_kbhit( ))*** evita que o programa fique parado no **usb_cdc_getc** (que fica esperando um caractere para prosseguir o programa). Veja o programa abaixo, que pisca um led na função principal (pino B6) e comanda o estado de outro led (pino B7) pelo teclado de um PC via USB:

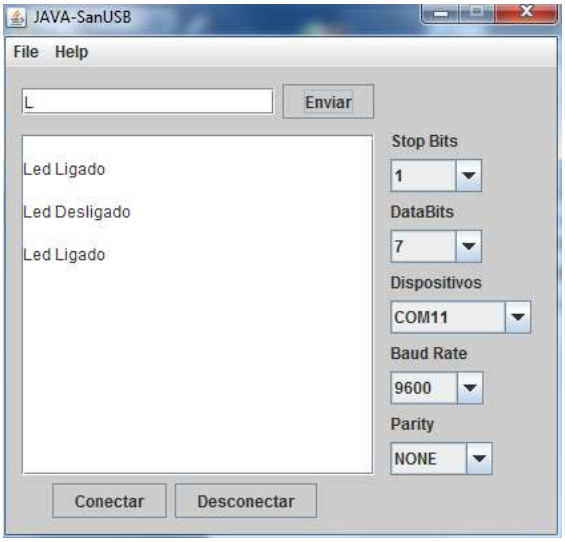
```
#include <SanUSB.h>
#include <usb_san_cdc.h>// Biblioteca para comunicação serial virtual

BYTE comando;

void main() {
clock_int_4MHz();//Função necessária para habilitar o dual clock (48MHz para USB e 4MHz para CPU)
usb_cdc_init(); // Inicializa o protocolo CDC
usb_init(); // Inicializa o protocolo USB
usb_task(); // Une o periférico com USB do PC

while (TRUE)
{
```

```
if (usb_cdc_kbhit( )) //avisa se chegou dados do PC
{ //verifica se tem um novo byte no buffer de recepção, depois o kbhit é zerado para próximo byte
comando=usb_cdc_getc(); //se chegou, retém o caractere e compara com 'L' ou 'D' em ASCII

if (comando=='L') {output_high(pin_b7); printf(usb_cdc_putc, "\r\nLed Ligado\r\n");}
if (comando=='D') {output_low(pin_b7); printf(usb_cdc_putc, "\r\nLed Desligado\r\n");}
}
output_high(pin_B6); // Pisca Led na função principal
delay_ms(500);
output_low(pin_B6);
delay_ms(500);
} }
```



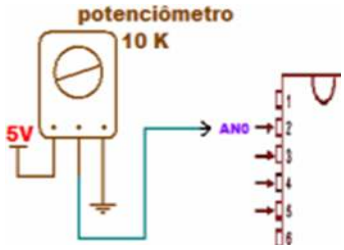

Para utilizar o programa de comunicação Java-SanUSB para emulação serial virtual entre o computador e o microcontrolador, é necessário baixá-lo através do link disponível em http://www.4shared.com/file/1itVIv9s/101009SoftwareComSerial_Window.html .

Após executar o programa de comunicação serial Java-SanUSB, verifique a porta COM virtual gerada (COM3,COM4,COM11,etc.) no Windows, em Painel de Controle\Todos os Itens do Painel de Controle\Sistema e altere no programa serial Java-SanUSB em Dispositivos e depois clique em Conectar, como mostra a figura abaixo.



## OBTENÇÃO DE UM VOLTÍMETRO ATRAVÉS DO CONVERSOR AD COM A VARIAÇÃO DE UM POTENCIÔMETRO

```
#include <SanUSB.h> //Leitura de tensão em mV com variação de um potenciômetro
#include <usb_san_cdc.h>// Biblioteca para comunicação serial virtual

int32 tensao;

void main() {
clock_int_4MHz();

usb_cdc_init(); // Inicializa o protocolo CDC
usb_init(); // Inicializa o protocolo USB
usb_task(); // Une o periférico com a usb do PC

setup_adc_ports(AN0);  //Habilita entrada analógica - A0
setup_adc(ADC_CLOCK_INTERNAL);

while(1){                       //ANALÓGICO        DIGITAL(10 bits)
set_adc_channel(0);               // 5000 mV             1023
delay_ms(10);                     // tensao              read_adc()
tensao= (5000*(int32)read_adc())/1023;
printf (usb_cdc_putc,"\r\nA tensao e' = %lu mV\r\n",tensao); // Imprime pela serial virtual

output_high(pin_b7);
delay_ms(500);
output_low(pin_b7);
delay_ms(500);              }}
```




## PROGRAMA COM INTERRUPÇÃO EXTERNA POR BOTÃO E DO TIMER 1

```
#include <SanUSB.h>

BYTE comando;

short int led;
int x;

#int_timer1
void trata_t1 ()
{
  led = !led; // inverte o led - pisca a cada 0,5 seg.
  output_bit (pin_b7,led);
  set_timer1(3036 + get_timer1());
}

#int_ext
void bot_ext()
{
  for(x=0;x<5;x++) // pisca 5 vezes após o pino ser aterrado (botão pressionado)
```

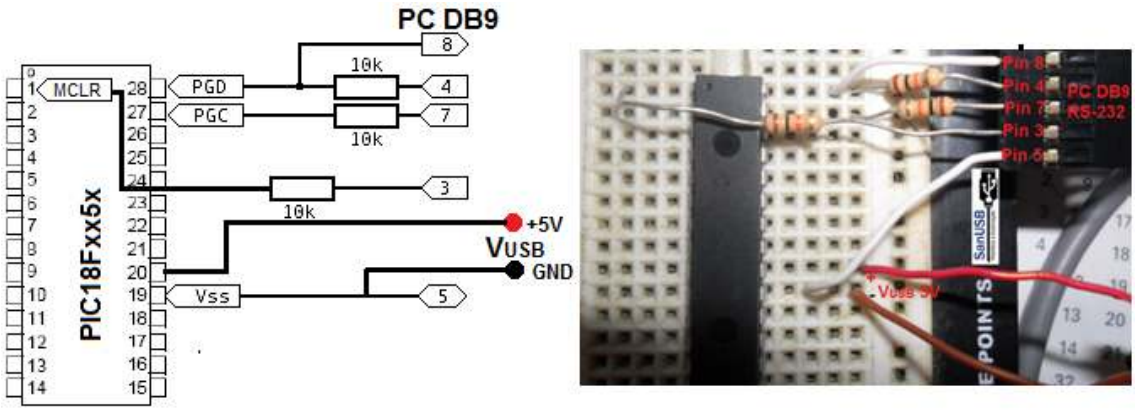
```
  {
  output_high(pin_B5); // Pisca Led em B5
  delay_ms(1000);
  output_low(pin_B5);
  delay_ms(1000);
  }
}

void main() {
clock_int_4MHz();
enable_interrupts (global); // Possibilita todas interrupcoes
enable_interrupts (int_timer1); // Habilita interrupcao do timer 1
//enable_interrupts (int_ext); // Habilita interrupcao externa 0 no pino B0
setup_timer_1 ( T1_INTERNAL | T1_DIV_BY_8);// configura o timer 1 em 8 x 62500 = 0,5s
set_timer1(3036);                // Conta 62.500us x 8 para estourar= 0,5s

while (1);  //Loop infinito (parado aqui)
                                        }
```

Para obter novos programas e projetos, basta acessar os arquivos do grupo SanUSB em www.tinyurl.com/SanUSB como também baixar a apostila completa disponível em http://www.4shared.com/document/Qst_pem-/100923Apostila_CPIC.html .

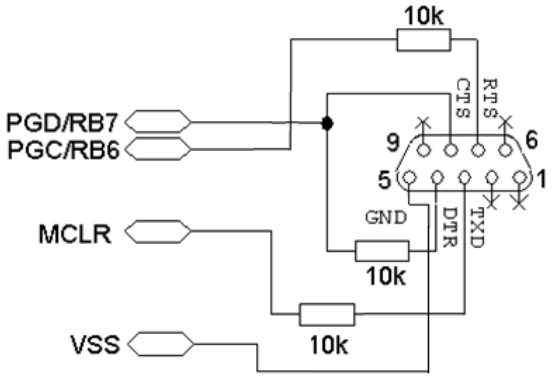
Divirta-se!
sandro_juca@yahoo.com.br

## CIRCUITO SIMPLES PARA GRAVAÇÃO DO *gerenciador.hex*

Para este circuito de gravação só é necessário 3 resistores de 10k, um cabo serial DB9 (RS-232) e uma fonte externa de 5V, que pode ser obtida da porta USB. O circuito e a foto abaixo mostram o esquema simples de ligação dos pinos.

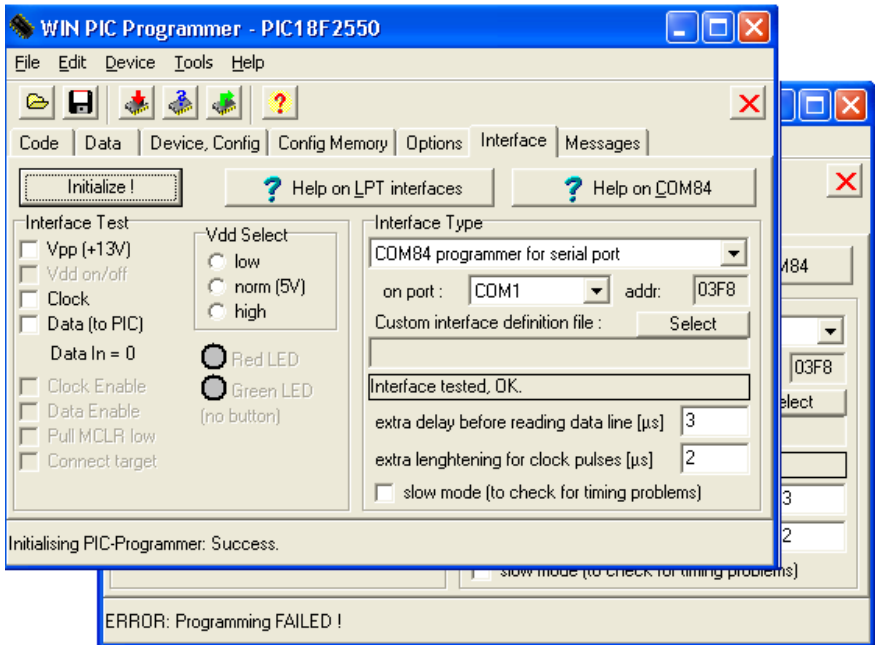



Este circuito a partir da porta COM DB9 pode ser visualizado na figura abaixo.



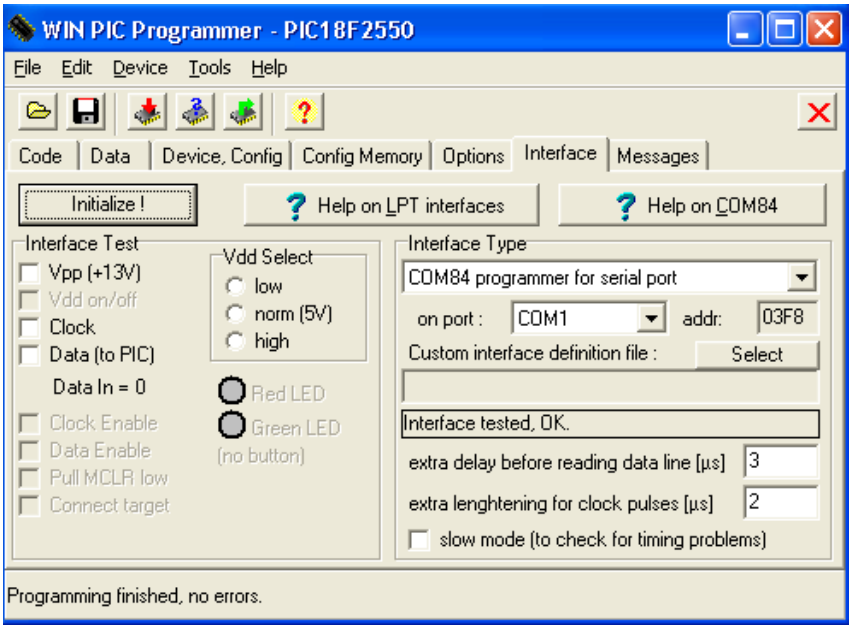

Este circuito de gravação funciona com o software PICPgm (detectado como JDM Programmer) ou com WinPic (detectado como COM84 Programmer). Este último se mostra mais estável, pois após a detecção do microcontrolador, é possível gravar o microcontrolador, e mesmo indicando *ERROR: Programming failed*, o arquivo gerenciador.hex mostrou-se gravado corretamente para gerenciar gravações no microcontrolador pela porta USB nos sistemas operacionais Windows®, Linux e Mac OSX.

O software Winpic pode ser baixado do site www.qsl.net/dl4yhf/winpicpr.html ou em http://www.4shared.com/get/1uP85Xru/winpicprCOM84.html.




Após a instalação, execute o programa. Na guia "*Device, Config*", escolha o microcontrolador. Uma vez que o microcontrolador é conectado à porta COM RS-232 de 9

pinos do PC, vá para "*Interface*", selecione " COM84 programmer for serial port", e pressione "Initialize". Se o software disser que a inicialização foi um êxito "*Success*", então o programa está pronto para gravar o gerenciador.hex no microcontrolador. Para a gravação, selecione em *File Load & ProgramDevice* e depois selecione o arquivo gerenciador.hex. Como citado anteriormente, mesmo que, após a gravação e verificação apareça "*Programmed Failed*", é provável que o gerenciador.hex tenha sido gravado corretamente.




## Firmware Header file for CCS C Compiler

SanUSB.h header File:

```
#include <18F4550.h> //This library 18F4550.h is valid for the whole family USB PIC18Fx5xx

#device ADC=10

#fuses HSPLL,PLL5, USBDIV,CPUDIV1,VREGEN,NOWDT,NOPROTECT,NOLVP,NODEBUG

#byte OSCCON=0XFD3

#use delay(clock=48000000)// USB standard frequency (cpu and timers 12 MIPS = 4/48MHz)
```

```
//#use delay(clock=4000000) // internal Oscillator Clock of 4MHz

#use rs232(baud=9600, xmit=pin_c6, rcv=pin_c7)

//SanUSB program memory allocation

#define CODE_START 0x1000

#build(reset=CODE_START, interrupt=CODE_START+0x08)

#org 0, CODE_START-1 {}



void clock_int_4MHz(void)

{

 //OSCCON=0B01100110; //with dual clock -> cpu and timers  #use delay(clock=4000000)

 while(read_eeprom(0xfd));

                }
```

## Firmware Modification for Microchip C18 compiler

```
/*  www.tinyurl.com/SanUSB    */



#include "p18F4550.h"



void low_isr(void);

void high_isr(void);



#pragma code low_vector=0x1018
```

```
void interrupt_at_low_vector(void)

{

  _asm GOTO low_isr _endasm

}

#pragma code

#pragma code high_vector=0x1008

void interrupt_at_high_vector(void)

{

  _asm GOTO high_isr _endasm

}

#pragma code



#pragma interruptlow low_isr

void low_isr (void)

{

   return;

}



#pragma interrupt high_isr

void high_isr (void)

{

 return;

}
```

```
void main( void )

{

     ...;

 }
```

## Firmware Modification for SDCC

Example Format

```
/*  www.tinyurl.com/SanUSB    */

#include <pic18f4550.h>

#pragma code _reset 0x001000

void _reset( void ) __naked

{

    __asm

    EXTERN __startup

    goto __startup

    __endasm;

}

  #pragma code _high_ISR  0x001008

  void _high_ISR( void ) __naked

  {

      __asm

      retfie
```

```
      __endasm;
  }
  #pragma code _low_ISR  0x001018
  void _low_ISR( void ) __naked
  {
      __asm
      retfie
      __endasm;
  }
void main() {  }
```

## Firmware Modification for MikroC

Example Format for Bootloader

```
/*  www.tinyurl.com/SanUSB    */


#pragma orgall 0x1000
void interrupt(void) org 0x1008
{
;
}
void interrupt_low(void) org 0x1018
{
```

```
;

}

void main()

{

  ......;

}
```

## Hi-Tech C Compiler

step1:goto Build option

step2:linker tap

step3:set offset : 1000

## Firmware Modification for Microchip ASM compiler

```
/*  www.tinyurl.com/SanUSB    */

        processor PIC18F4550

        #include"p18f4550.inc"



        org 0x1000

        goto init

        org 0x1020

        goto int_isr



init
```

```
        ...             ; initialization
loop
        ...             ; code
        goto loop
int_isr
        ...             ; interrupt code
        retfie
        end
```

## REFERÊNCIAS BIBLIOGRÁFICAS


Grupo SanUSB (2011). *Arquivos do Grupo SanUSB*. Retirado em 05/01/11, no World Wide Web: www.tinyurl.com/SanUSB/.

Jornal O Povo (2011). *Da escola pública para o mundo*. Retirado em 05/01/11, no World Wide Web: http://www.opovo.com.br/app/opovo/cienciaesaude/2011/01/08/noticiacienciaesaudejornal,2086691/da-escola-publica-para-o-mundo.shtml.

Jucá, S. *et al.*(2011). *A low cost concept for data acquisition systems applied to decentralized renewable energy plants*. Retirado em 05/01/11, no World Wide Web: http://www.mdpi.com/1424-8220/11/1/743 .

Jucá, S. *et al.*(2011). *Gravação de microcontroladores PIC via USB pelo terminal do Linux*. Retirado em 05/03/11, no World Wide Web: http://www.vivaolinux.com.br/artigo/Gravacao-de-microcontroladores-PIC-via-USB-pelo-terminal-do-Linux/.

Jornal O Povo (2010). *De Maracanaú para Eslováquia*. Retirado em 05/01/11, no World Wide Web: http://publica.hom.opovo.com.br/page,489,109.html?i=2051467 .

Diário do Nordeste (2010). *Robô cearense*. Retirado em 05/01/11, no World Wide Web: http://diariodonordeste.globo.com/materia.asp?codigo=861891.

TV Diário (2010). *Feira do Empreendedorismo SEBRAE*. Retirado em 05/01/11, no World Wide Web: http://www.youtube.com/watch?v=8Y7gOPd_zN4.

TV Verdes Mares (2009). *Estudantes competem com robôs*. Retirado em 05/01/11, no World Wide Web: http://tvverdesmares.com.br/bomdiaceara/estudantes-competem-com-robos/.

TV Cidade (2009). *Projetos Comsolid/Setapi IFCE*. Retirado em 05/01/11, no World Wide Web: http://www.youtube.com/watch?v=i_waT0_201o.

Jucá, S. *et al.*(2009). *SanUSB: software educacional para o ensino da tecnologia de microcontroladores*. Retirado em 05/01/11, no World Wide Web: http://www.cienciasecognicao.org/pdf/v14_3/m254.pdf .

Diário do Nordeste (2007). *Alunos estimulados a construir robôs*. Retirado em 05/01/11, no World Wide Web: http://diariodonordeste.globo.com/materia.asp?codigo=491710.