USB Interfacing and Real Time Data Plotting with MATLAB

Department of Electrical and Computer Engineering
Real-Time DSP and FPGA Development Lab
Mark S. Manalo and Ashkan Ashrafi

Table of Contents

Introduction

Intro

This tutorial will explain the process of creating a real time plot using MATLAB. Data from a USB port will be read and plotted on a virtual oscilloscope. We will use a device that will write data to a USB port. Then in MATLAB we will write a program to continuously read and display the value from the port.

Required Materials

The following materials are required to follow the tutorial:

1. A device that is continuously writing to the USB port, this can be a microcontroller or a USB based sensor (Note: you need to know what the data stream looks like)
2. MATLAB programming knowledge

MATLAB Programming

Install Required Functions

To easily implement real-time data plotting we will use a pre-made function available in MATLAB File Exchange: http://www.mathworks.com/matlabcentral/fileexchange/

Go to the following website and download the 'Test and Measurement Widgets' function by Scott Hirsch: http://www.mathworks.com/matlabcentral/fileexchange/4722-test-and-measurement-widgets/content/demo_tmwidgets.m

Then set the MATLAB path to point to the files we just downloaded. To do this, in MATLAB click File → Set Path. A new window will come up as shown in Figure 1. Click Add Folder... and browse to the location of the Strip Chart folder. Click Save and close the window.

The complete MATLAB code is located in the Appendix titled 'MATLAB Realtime Plot Code' and 'MATLAB getData function'. What follows are some explanations and code segments that we have used in this project. (These code segments are only for teaching purposes, refer to the appendix for the specific values we used to interface with our USB device).

For our system a device is connected to the USB port that continuously reads ADC data and sends it over to the USB. Our device also needs to be sent a start command to begin transmission. The data stream has this format (your device will vary):

-- Payload

-uint8 token

-uint8 pktcnt

-uint8 msgSize

-uint8 deviceID

-uint16 pktctr

-uint8 adc_chan[ADC_SAMPLES][CHAN_PER_DEV]

11 samples 4 channels

rows columns

-uint8 accel[3]

We need to write MATLAB code that will parse this data stream. But before that is done, we need a way for MATLAB to communicate with our device. The following section will describe how this is accomplished.

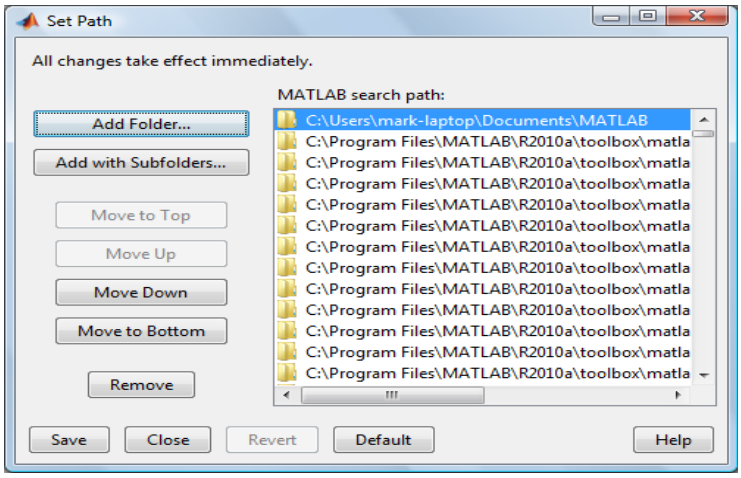

Figure 1: Add the TMWidgets Folder

Open COM Port

The first part in our MATLAB program is to open a COM port so that we can read data from it. Create a new MATLAB script and write the following code:

```
%% Create a serial object
board = serial('COM3', 'BaudRate', 256000, 'DataBits',8);
fopen(board);
```

This code will open and connect to COM3 with a speed of 256000 baud and 8 bits of data (make sure you know which COM port your USB device is connected to, in my case the device is connected to USB COM port 3 with a baud rate of 256000)

Setup the Stripchart

The stripchart will be continuously updated by our timer function. But before we do that we need to initially setup the stripchart figure. The following code accomplishes this task:

```
%% Initialize the stripchart
clf
Fs = 300;
AxesWidth = 2;          % Axes Width (s)
figure(1);  stripchart(Fs,AxesWidth);
```

For more info on the stripchart function, type 'help stripchart' in the MATLAB console. A picture of the stripchart figure is shown in Figure 2.

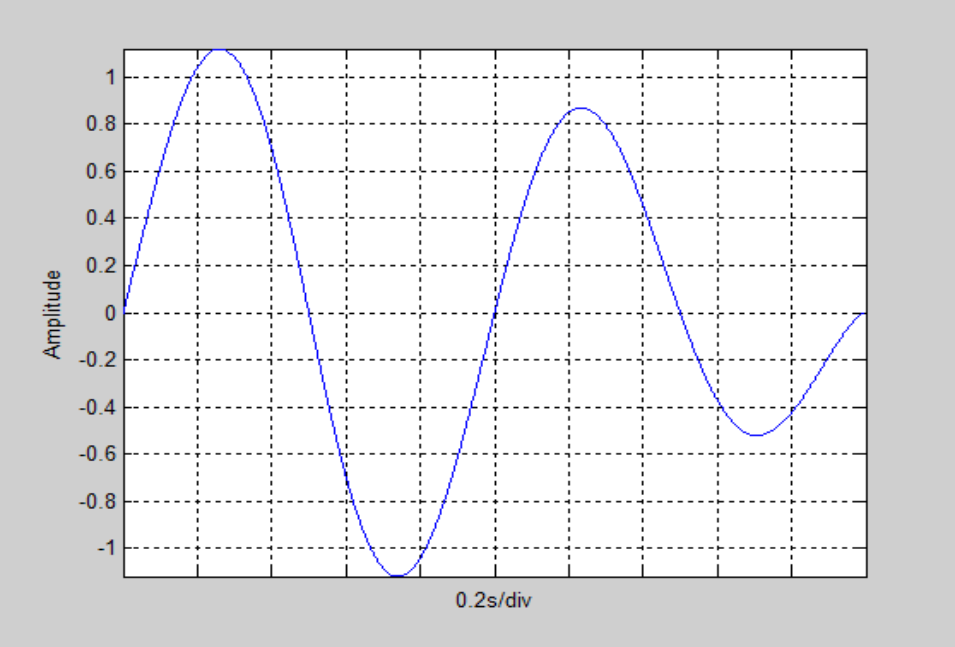

Figure 2: Example Stripchart figure

Timer Function

Next we need a way to continuously read the COM port at fixed time intervals. For this case we will use a timer function to accomplish this task. The following code will create a timer object.

```
%% Setup a timer
% The timer has a callback that reads the serial port and updates the
% stripchart
t = timer('TimerFcn', @(x,y)getData(board), 'Period', 0.1);
set(t,'ExecutionMode','fixedRate');
```

```
start(t);
```

We created and started a timer called t that will execute every 0.1sec. When the timer is called it will also run the callback function called 'getData' which will be explained in the next section.

Callback Function

Every time the timer function finishes its countdown we will tell it to read the data from the COM port and update the strip chart with that data. To do this we will create a new function called 'getData'. In MATLAB click File → New → Script and copy the following code:

```
function  getData(board)
    data = fread(board,10);                    % read 10 bytes
    fread(board, board.BytesAvailable);        % clear the buffer

    % update the stripchart
    figure(1); stripchart(data(1:end));
```

The function 'getData' will read 10 bytes from the COM port, clear the COM port buffer and update the stripchart. Note: (normally it is not a good idea to clear the buffer because we will lose some data. The UBS device we are using has a control byte called TOKEN that we will use to synchronize with it.)

This completes the MATLAB code. After clicking the play button you should know be able to read and plot data from a device connected to your USB port.

Appendix

MATLAB Realtime Plot Code

```
close all;          % close all figures
clear all;          % clear all workspace variables
clc;                % clear the command line
fclose('all');      % close all open files
delete(instrfindall);   % Reset Com Port
delete(timerfindall);   % Delete Timers

% ************************************************
% Constants
% ************************************************
TOKEN       = 238;
BAUDERATE   = 256000;
INPUTBUFFER = 512;
START       = [255 7 3];  % 0XFF, 0X07, 0X03
STOP        = [255 9 3];  % 0XFF, 0X09, 0X03
TMR_PERIOD  = 0.003;
```

```
%% Initialize the stripchart

clf
Fs = 300;
AxesWidth = 2;          % Axes Width (s)
figure(1);  stripchart(Fs,AxesWidth);
% figure(2);  stripchart(Fs,AxesWidth);
% figure(3);  stripchart(Fs,AxesWidth);
% figure(4);  stripchart(Fs,AxesWidth);


%% Create a serial object
board = serial('COM3', 'BaudRate', 256000, 'DataBits',8);

% Set serial port buffer
set(board,'InputBufferSize', INPUTBUFFER);

fopen(board);

% Send the Start command to the radio
fwrite(board,START,'uint8');

%% Setup a timer
% The timer has a callback that reads the serial port and updates the
% stripchart
% Construct a timer object with a timer callback funciton handle,
% getData
t = timer('TimerFcn', @(x,y)getData(board), 'Period', TMR_PERIOD);
set(t,'ExecutionMode','fixedRate');
start(t);
```

MATLAB getData function

```
function getData(board)
      %% Receive the Data
      % After sending the start command to the radio, the receiver will start
      % receiving values and writing them to the serial port
      % Payload
      %   -uint8  token
      %   -uint8  pktcnt
      %   -uint8  msgSize
      %   -uint8  deviceID
      %   -uint16 pktctr
      %   -uint8  adc_chan[ADC_SAMPLES][CHAN_PER_DEV]
      %                   11 samples   4 channels
      %                   rows         columns
      %   -uint8  accel[3]

      % ************************************************
      % Constants
      % ************************************************
      TOKEN       = 238;
      PACKETS     = 2;        % #Of packets to store, 1 packet = 11 samples
```

```
payload = zeros(100,1);

%% Parse the Data

start = 1;
for j = 1:1:PACKETS

    % Wait for token to arrive for start of payload
    while (fread(board,1) ~= TOKEN)
    end

    payload(1:100,1) = fread(board, 100);
    %fread(board, board.BytesAvailable);
    disp('Reading data');

    m = 6;
    for i=start:1:start+10
        data1(i) = payload(m);          % Channel 1 sample i
        data2(i) = payload(m+1);        % Channel 2 sample i
        data3(i) = payload(m+2);        % Channel 3 sample i
        data4(i) = payload(m+3);        % Channel 4 sample i

        m=m+4;
    end
    start = start + 11;
end

%% Update the stripchart for data1
% we can also update for data2-data4 but more than 1 would be laggy

figure(1); stripchart(data1(1:end)); title('Channel 1'); drawnow;
```

References

"File Exchange Pick of the Week." *Advanced MATLAB: Timer Objects*. Web. 30 Apr. 2012. <http://blogs.mathworks.com/pick/2008/05/05/advanced-matlab-timer-objects/>.

"Test and Measurement Widgets: Demo_tmwidgets - File Exchange - MATLAB Central."*Test and Measurement Widgets: Demo_tmwidgets - File Exchange - MATLAB Central*. Web. 08 May 2012. <http://www.mathworks.com/matlabcentral/fileexchange/4722-test-and-measurement-widgets/content/demo_tmwidgets.m>.

"Thread Subject: Serial Port at 115200 Baud Rate for Real Time Plot." *Serial Port at 115200 Baud Rate for Real Time Plot*. Web. 30 Apr. 2012. <http://www.mathworks.com/matlabcentral/newsreader/view_thread/240440>.