TECH NOTE :: ClipX Matlab Einbindung

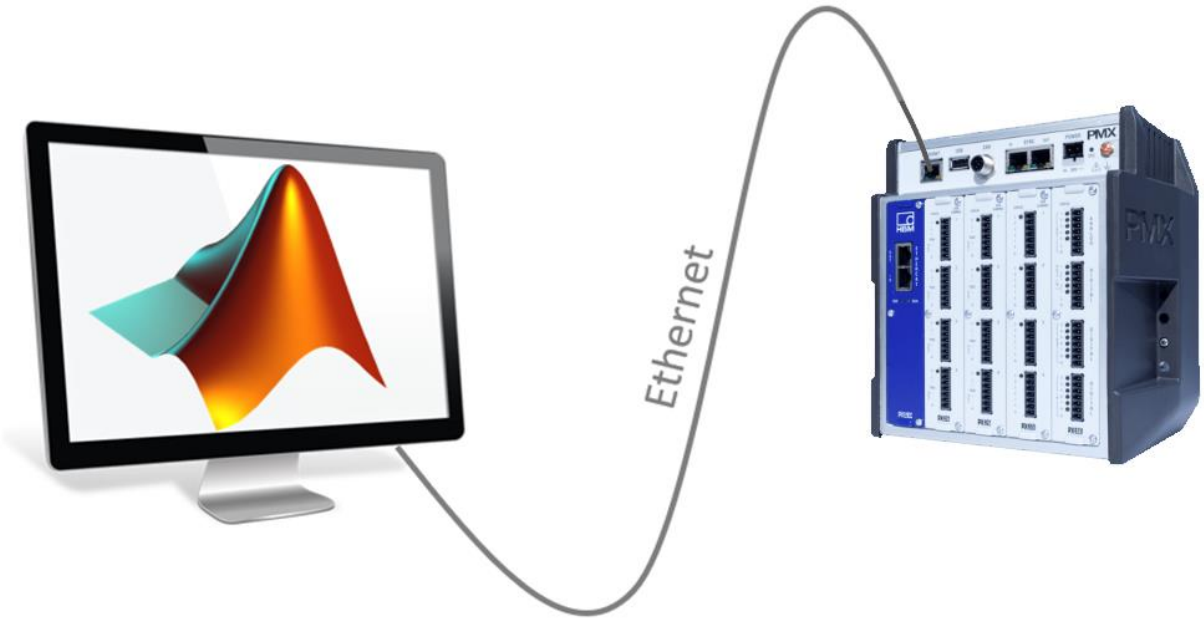
Version: 2019-09-12
Autor: Michael Guckes, Roland Siepmann
Status: HBM: Public

Kurzbeschreibung

Dies ist eine Anleitung zur Einbindung von PMX in Matlab. Zur Kommunikation mit PMX wird die Befehlsschnittstelle des PMX verwendet. Die hierzu benötigten Befehle werden über eine Socketverbindung an PMX gesendet und müssen für die gewünschte Messung in den Matlabdateien adaptiert werden.
Im Folgenden wird vorausgesetzt, dass Matlab bereits installiert wurde.

Wichtig: Für eine korrekte Darstellung der Signale von PMX müssen diese, wie in diesem Beispiel, über dessen NTP-Zeitkanal geplottet werden. Theoretisch ist eine Messrate von bis zu 19,2kHz möglich. Diese hohe Messrate kann allerdings zu Performanceproblemen führen, abhängig von der Leistungsstärke des verwendeten Systems.

Hinweis: Stellen Sie sicher, dass Sie die aktuellste Version Beispiels verwenden.
https://www.hbm.com/de/2981/pmx-modulares-messverstaerkersystem-fuer-industrie-4-0/



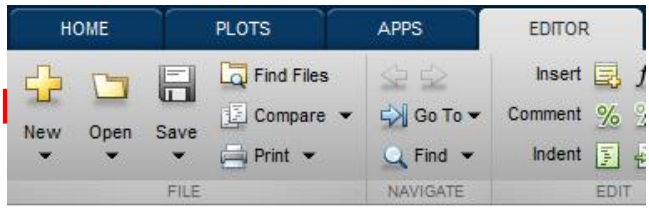
Inhalt

Das zur Verfügung gestellte Beispiel besteht aus den folgenden Dateien:

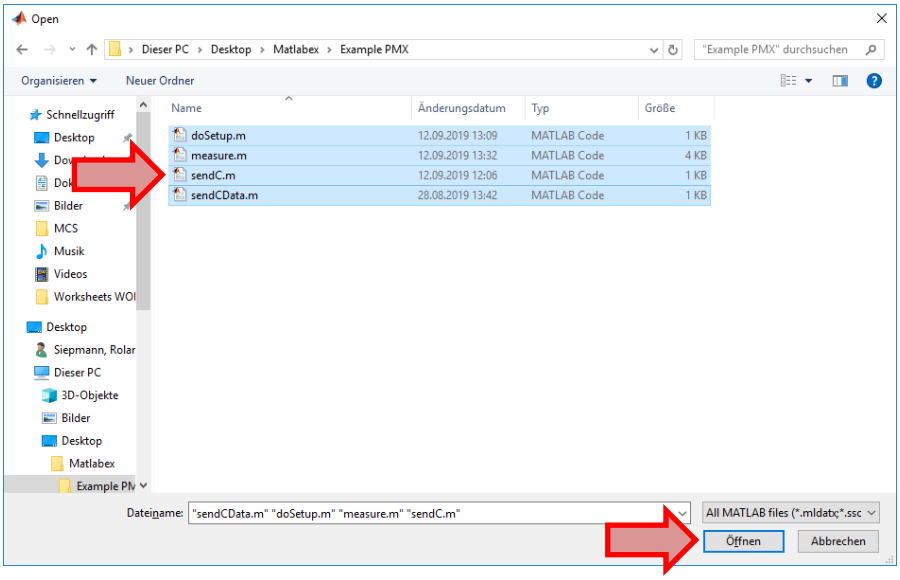
- Matlabskript „measure.m“ zur Messung mit anschließendem Plot
- Matlabsfunktion „doSetup.m“ initialisiert die Messung und die Messparameter
- Matlabfunktion „sendC.m“ sendet Befehle zu PMX
- Matlabfunktion „sendCData.m“ liest den berechneten Umfang an Messdaten vom Socket

Zur Durchführung einer Messung wird Matlab gestartet.

- In der Menüleiste „Öffnen“ wählen



- Anschließend die beiden oben genannten Dateien öffnen



doSetup.m

Diese Funktion initialisiert die Messparameter. Hier werden die zu übertragenden Messkanäle, Messraten etc. eingestellt. In diesem Beispiel wird der Berechnungskanal 9.1 mit einer Messrate von 19,2kHz aufgezeichnet.

Hinweis: Die Parameter und Befehle sind der PMX Bedienungsanleitung (Kapitel: Befehlssatz des PMX) zu entnehmen.

```
function [sock] = doSetup(ip)
sock = tcpip(ip, 55000, 'NetworkRole', 'client');
sock.InputBufferSize = 90000000;
fopen(sock);

%%Set up measurement group 0
%Select measuring card
sendC(sock, 'PCS9');
%Select measuring channel
sendC(sock, 'SPS1');
%Select desired signal (gross, net,...)
sendC(sock, 'MSS214');
%Assign to measuring group (0,1,2)
sendC(sock, 'MRG0');

%Set up a card for measuring
sendC(sock, 'PCS9');
%Select channel
sendC(sock, 'SMS1');


%Select measuring channel
sendC(sock, 'MCS9,17');

%Set buffer format
sendC(sock, 'MBF1257,0');

%Set measuring rate for group 0
sendC(sock, 'ICR6345,0');

%Set time format (2 = milliseconds + seconds)
sendC(sock, 'STF2');

end
```

measure.m

Mit diesem Matlabskript wird eine Messung ausgeführt und die Messwerte anschließend geplottet. Nach der Messung stehen die Messwerte zu Weiterverarbeitung in Arrays zur Verfügung.

Parametereinstellungen

Die folgenden Parameter sind in der Datei ClipXMatlabStore.m anzupassen:

- ip: Die IP-Adresse des gewünschten ClipX Gerätes
- maxchannels: Hier die Anzahl an übertragenen Messkanälen angeben (ohne Zeitkanäle)
- timechannels: Hier die Anzahl an übertragenen Zeitkanälen angeben
- Wertearrays: An dieser Stelle muss für jeden übertragenen Messkanal ein Array initialisiert werden

```
1   %Edit IP address here
2   ip = '172.21.64.40';
3
4   %transmitted measuring channels
5   maxchannels = 1;
6
7   %transmitted time channels
8   timechannels = 1;
9
10  %Sets up the measurement
11  sock = doSetup(ip);
12
13  %Start measurement for group 0
14  data1 = sendC(sock, 'TSV0');
15
16  %clock for measurement - change endtime for measurement duration
17  acttime = clock;
18  endtime = acttime(6) + 10;
19
20  %Calculation of memory requirement of a line
21  datalineb = maxchannels*4 + timechannels*8;
22
23  %Value and time arrays (need to be added, if more signals are transmitted)
24  recValues1 = [];
25  recValues2 = [];
26  recNTP = [];
```

Die While-Schleife führt die Messwertaufnahme durch. Zu Beginn werden immer die Anzahl der verfügbaren Messwertzeilen und der Messmodus abgefragt. Wenn Messwertzeilen verfügbar sind und gemessen wird, wird die Datenmenge hierfür berechnet und vom Socket gelesen. In der For-Schleife werden die Daten den Arrays zugewiesen. Abhängig davon, wieviele Kanäle gemessen werden, muss der rot umrahmte Block entsprechend oft kopiert und einem neuen Array zugewiesen werden (wie unten gezeigt).

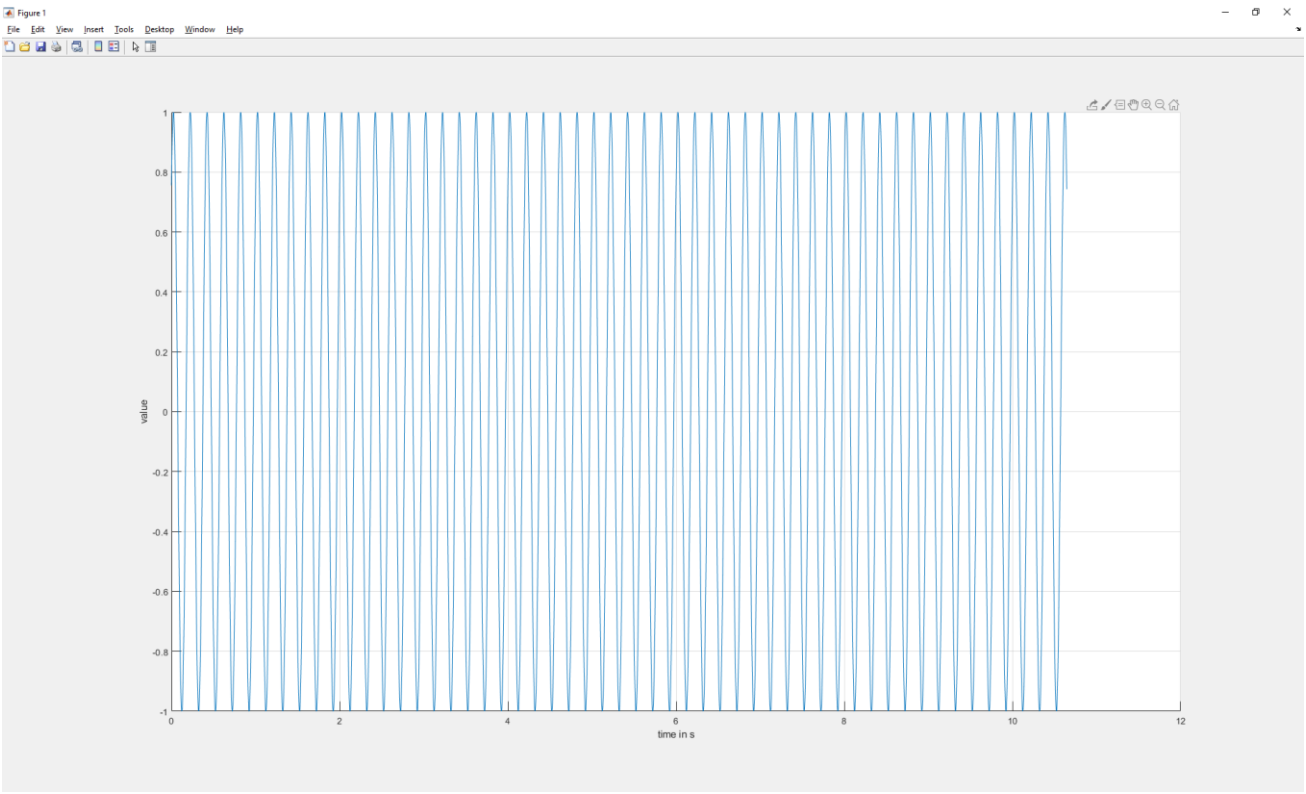
```
while acttime(6) < endtime
    datal = sendC(sock, 'OMP? 0');
    datal = split(datal, ',');
    lines = datal(1);
    msrmode = str2double(datal(2));
    if msrmode == 1
       if str2double(lines) > 0
            am = 4+(str2double(lines{1,1})*datalineb);
            data = sendCData(sock, char(strcat('RMB?', lines, ',6409,0')), am);
            noOfDataLines = (length(data)-4) / datalineb;
            bytePos = 3;

            for dataline = 1:1:(noOfDataLines)
                %{Read Measurement Values}%
                %Read Measurement Value1
                valueBuff = data(bytePos:(bytePos + 3));
                a = uint8(valueBuff);
                val = typecast(a, 'single');
                recValues1 = [recValues1 val];
                bytePos = bytePos + 4;

                %Read Measurement Value2
                %valueBuff = data(bytePos:(bytePos + 3));
                %a = uint8(valueBuff);
                %val = typecast(a, 'single');
                %recValues2 = [recValues2 val];
                %bytePos = bytePos + 4;

                %Add more measurement values here
```

Im Folgenden wird der Zeitstempel zusammengesetzt. In diesem Beispiel ist das Zeitformat so gewählt, dass Sekunden und Mikrosekunden übertragen werden. Diese Einstellung kann nach belieben in der doSetup.m Funktion adaptiert werden. Anschließend wird der Wert dem Zeitarray hinzugefügt.

```
58                  %{Read Time}%
59
60                  %%Optional Block to get the time in ticks (STF needs to be
61                  %%set to 0)
62                  %Get 64bit time
63                  %valueBuff = data(bytePos:(bytePos + 7));
64                  %a = uint8(valueBuff);
65                  %time = typecast(a, 'uint64');
66                  %time = double(time);
67                  %time = time*10^(-5);
68                  %bytePos = bytePos + 8;
69
70                  %Get ms time
71 -                valueBuff = data(bytePos:(bytePos + 3));
72 -                a = uint8(valueBuff);
73 -                timems = typecast(a, 'uint32');
74 -                bytePos = bytePos + 4;
75
76                  %Get s time
77 -                valueBuff = data(bytePos:(bytePos + 3));
78 -                a = uint8(valueBuff);
79 -                times = typecast(a, 'uint32');
80 -                bytePos = bytePos + 4;
81
82                  %Get doubles
83 -                timems = double(timems);
84 -                times = double(times);
85
86                  %Get exact time in s
87 -                timems = timems*10^(-6);
88 -                time = timems + times;
89
90                  %Add time to timearray
91 -                recNTP = [recNTP time];
```

Messung

Zur Durchführung der Messung muss nur das Skript gestartet werden. Die Messung eines 5Hz Sinus als Berechnungskanal sieht bspw. wie folgt aus:



Rechtlicher Hinweis

Diese Beispiele dienen lediglich der Veranschaulichung. Sie unterliegen keinen Gewährleistung oder Haftungsansprüchen.