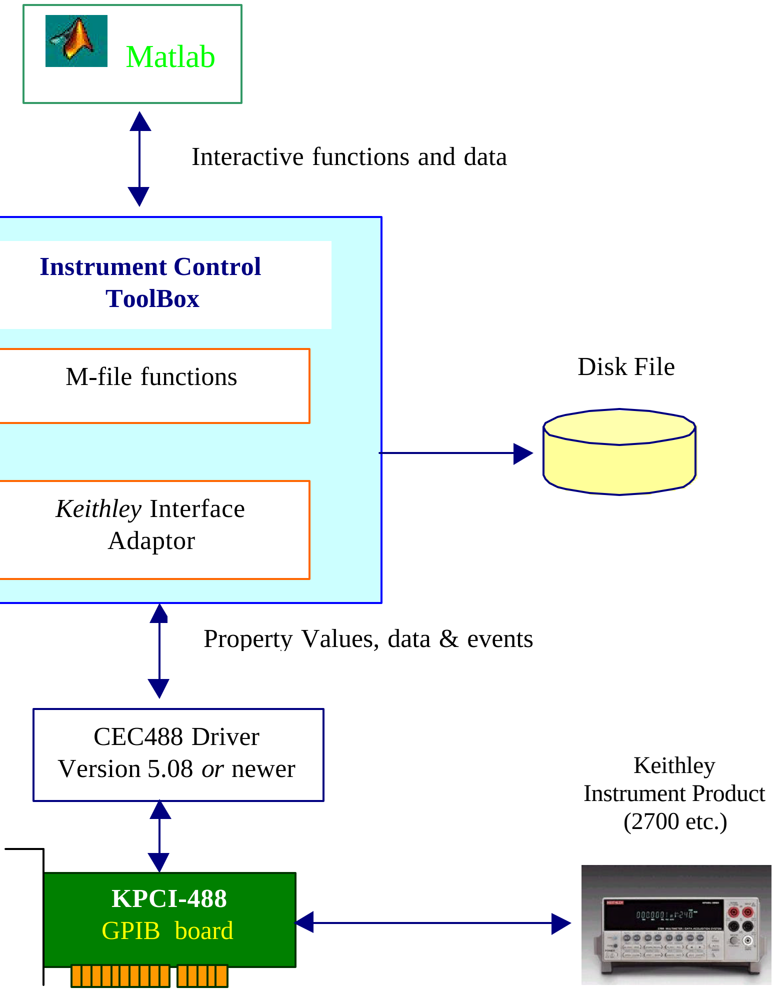
Using MATLAB® with KEITHLEY Instrument Devices through GPIB

Preface

This application note is intended to help the user in setup and configuration of Keithley Instrument products through Keithley GPIB cards in Matlab. It does not cover extensive programming issues.

Background

The Instrument Control Toolbox is a collection of M-file functions built on the MATLAB® technical computing environment. The toolbox provides you with a framework for communicating with instruments that support the GPIB interface (IEEE-488, HPIB), the VISA standard, and the serial port interface (RS-232, RS-422, and RS-485). Currently, the Instrument Control Toolbox supports interface for Keithley's Instrument products through GPIB and serial port.


Matlab
Interactive functions and data
Instrument Control ToolBox
M-file functions
Keithley Interface Adaptor
Disk File
Property Values, data & events
CEC488 Driver Version 5.08 or newer
Keithley Instrument Product (2700 etc.)
KPCI-488 GPIB board

Using Matlab with Instrument Control ToolBox & Keithley Instrument Device.

MATLAB® is a registered trade mark of The MathWorks Inc.

Requirements

Unlike the Data Acquisition Toolbox, you do not need to download a specific adaptor. But you must have the Instrument Control Toolbox in addition to the Matlab base package. You also need 'Keithley-488' or 'CEC-488' driver version 5.08 or newer for your Keithley IEEE-488 card.

Using the Instrument Control Toolbox

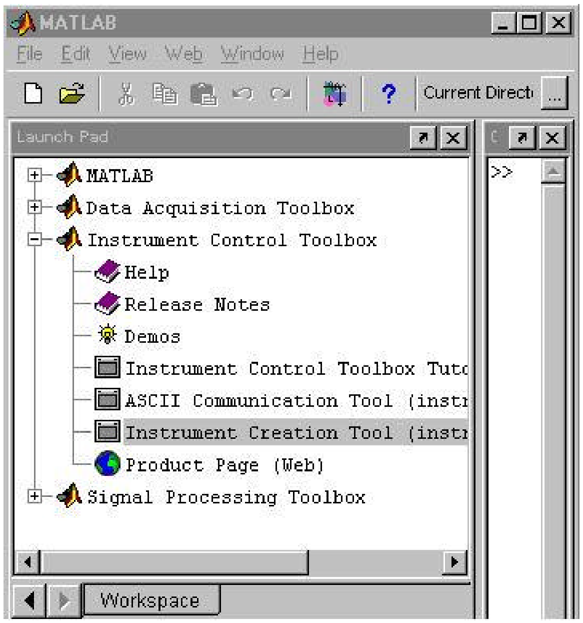
In most cases, you should follow these steps.

1. Create an instrument object ie. name = gpib('keithley', '*index*', '*gpib address*')
2. Configure properties using 'set' commands
3. Connect to the device ie. fopen('name')
4. Write and read data ie. fprintf(*gpib object*, '*SCPI COMMANDS*') etc.
5. Disconnect and clean up ie. fclose('name') etc.

Building an M-file

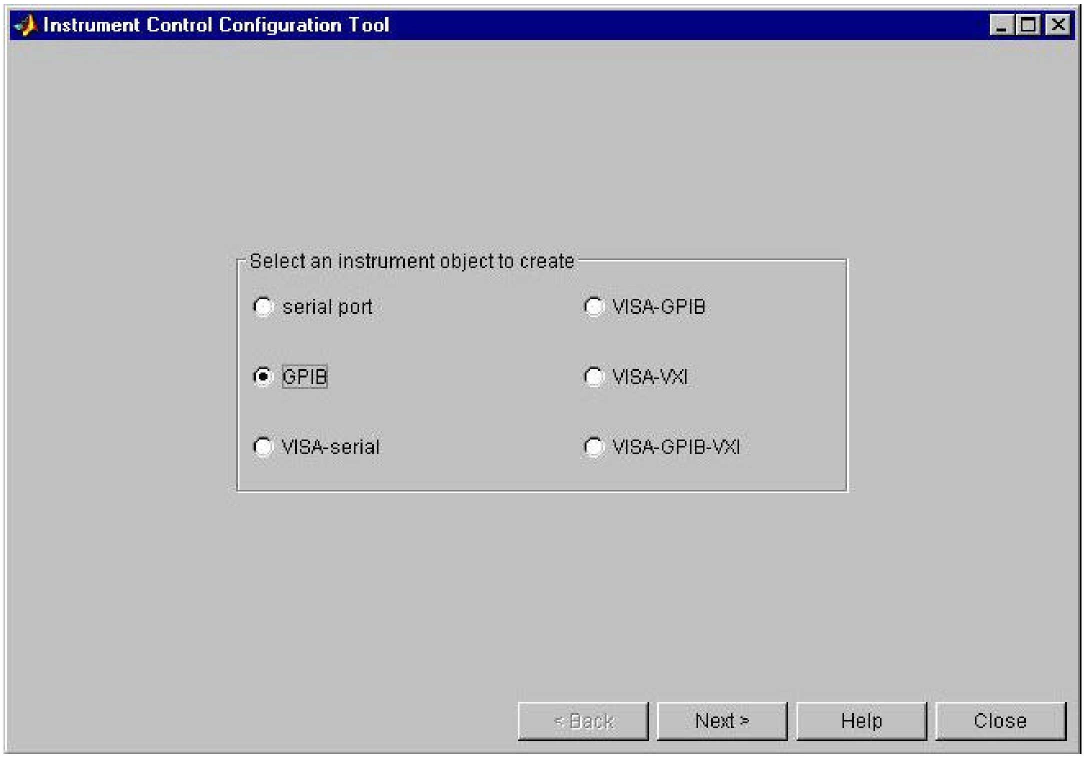
To create an M-file, you can either manually configure all the properties or use "Instrument Creation Tool" to generate part of the code. We recommend that you take advantage of this useful tool.

You can launch the Instrument Creation Tool either by double clicking the icon on the launch pad or by typing 'instrcreate' command in the Matlab command prompt.


MATLAB
File Edit View Web Window Help
Current Directo
Launch Pad
MATLAB
Data Acquisition Toolbox
Instrument Control Toolbox
Help
Release Notes
Demos
Instrument Control Toolbox Tuto
ASCII Communication Tool (instr
Instrument Creation Tool (instr
Product Page (Web)
Signal Processing Toolbox
>>
Workspace

Launching the Instrument Creation Tool from the Launch Pad.

After launching, you simply select the configuration you want step by step.



Select an instrument object to create

○ serial port ○ VISA-GPIB

● GPIB ○ VISA-VXI

○ VISA-serial ○ VISA-GPIB-VXI

The Instrument Control Configuration Tool.

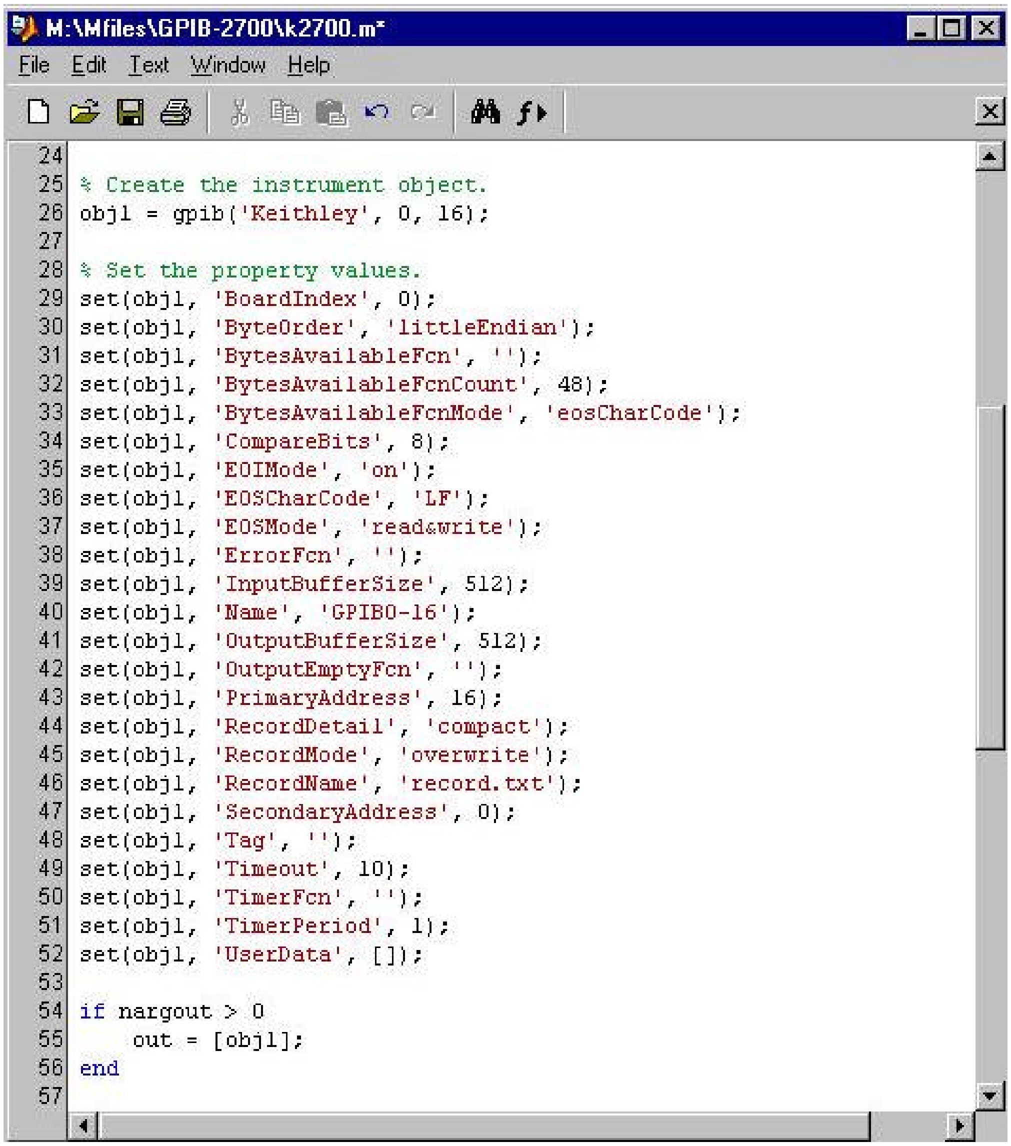
Choose:

- Instrument object to create → GPIB
- You need to select several properties on the next screen
  - Select Adaptor → Keithley
  - Enter the GPIB board index (*Normally 0 is fine*)
  - Enter GPIB Address of your instrument (*16 etc.*)
  - Choose EOSMode → *Choose* read & write
  - EOIMode → *will not matter: see the Limitations*
  - CompareBits → Match All 8 Bits
  - *Note: You can set other properties such as the name of file to log data, under "General" tab.*

- On the last screen, enter:
  - Matlab work space with GPIB object name (*The object name will default to **obj1** no matter what you choose*).
  - M-file & name
  - .mat file & name (***Note: You may not be able read your m-file without this .mat file so you should choose to create it!***)

- Click *Create* to generate *.m* and *.mat* files in your current directory.

When you open your M-file, you will see that the *gpib* object is created and several properties are set for the object.



```

% Create the instrument object.
obj1 = gpib('Keithley', 0, 16);

% Set the property values.
set(obj1, 'BoardIndex', 0);
set(obj1, 'ByteOrder', 'littleEndian');
set(obj1, 'BytesAvailableFcn', '');
set(obj1, 'BytesAvailableFcnCount', 48);
set(obj1, 'BytesAvailableFcnMode', 'eosCharCode');
set(obj1, 'CompareBits', 8);
set(obj1, 'EOIMode', 'on');
set(obj1, 'EOSCharCode', 'LF');
set(obj1, 'EOSMode', 'read&write');
set(obj1, 'ErrorFcn', '');
set(obj1, 'InputBufferSize', 512);
set(obj1, 'Name', 'GPIB0-16');
set(obj1, 'OutputBufferSize', 512);
set(obj1, 'OutputEmptyFcn', '');
set(obj1, 'PrimaryAddress', 16);
set(obj1, 'RecordDetail', 'compact');
set(obj1, 'RecordMode', 'overwrite');
set(obj1, 'RecordName', 'record.txt');
set(obj1, 'SecondaryAddress', 0);
set(obj1, 'Tag', '');
set(obj1, 'Timeout', 10);
set(obj1, 'TimerFcn', '');
set(obj1, 'TimerPeriod', 1);
set(obj1, 'UserData', []);

if nargout > 0
    out = [obj1];
end

```

The code generated by Instrument Control Configuration Tool.

You can add your code at the end of the generated code. You should start with 'fopen' to connect to the device. Since you have the properties already set, you can now send SCPI commands to the device using 'fprintf' and 'fscanf' commands. At the end of your code, you must have 'fclose' command to disconnect the device. Note that delete('*gpib object*') command will not disconnect the object.

Example

This simple example acquires one temperature reading from 2700 with a 7700 module in the slot 1.

```
% open the gpib object
fopen(obj1)
% reset and identify
fprintf(obj1,'*RST')
fprintf(obj1,'*IDN?')
% display the id
idn = fscanf(obj1)

% configure the 2700 for 1 temperature reading
% from 7700 card
fprintf(obj1,'SENS:FUNC "TEMP"')
fprintf(obj1,'UNIT:TEMP C')
fprintf(obj1,'TEMP:TRAN TC')
fprintf(obj1,'TEMP:TC:TYPE J')
fprintf(obj1,'TEMP:TC:RJUN:RSEL INT')
fprintf(obj1,'ROUT:CLOS (@101)')
fprintf(obj1,'FORM:ELEM READ')
fprintf(obj1,'TRAC:CLE')
fprintf(obj1,'TRAC:POIN 1')
fprintf(obj1,'INIT')
fprintf(obj1,'TRACE:FEED:CONT NEXT')
fprintf(obj1,'TRAC:DATA')

fprintf(obj1,'READ?')
% Scan and display the reading
TCtemp = fscanf(obj1)
% IMPORTANT: disconnect the device
fclose(obj1)
% clear the memory
clear all
```

Note: If you want to insert time delays, you can use `pause(x)` command where `x` is in seconds.

Limitations *(excerpt from Mathlab Help)*

The Keithley GPIB driver has these limitations:

- Asynchronous read and write operations are not supported. Therefore, Keithley GPIB objects do not support the following toolbox functionality:
  - The readasync function
  - The async flag for the fprintf and fwrite functions
  - The BytesAvailableAction and OutputEmptyAction properties
- The Handshake and Bus Management line values are not provided. The BusManagementStatus and HandshakeStatus properties always return the line value as on.
- The EOI line is not asserted when the EOS character is written to the hardware. Therefore, when the EOSMode property is configured to write and the EOIMode property is configured to on, the EOI line will not be asserted when the EOSCharCode property value is written to the hardware.
- All eight bits are used for the EOS comparison. Therefore, the only value supported by the CompareBits property is 8.
- You should not simultaneously use a GPIB controller address of 0 and an instrument primary address of 0.

Conclusion

The Instrument Creation Tool enables you to set up the GPIB properties quickly and to start writing the code in SCPI commands. Once you get your data, all of Matlab's analysis capabilities are at your disposal.