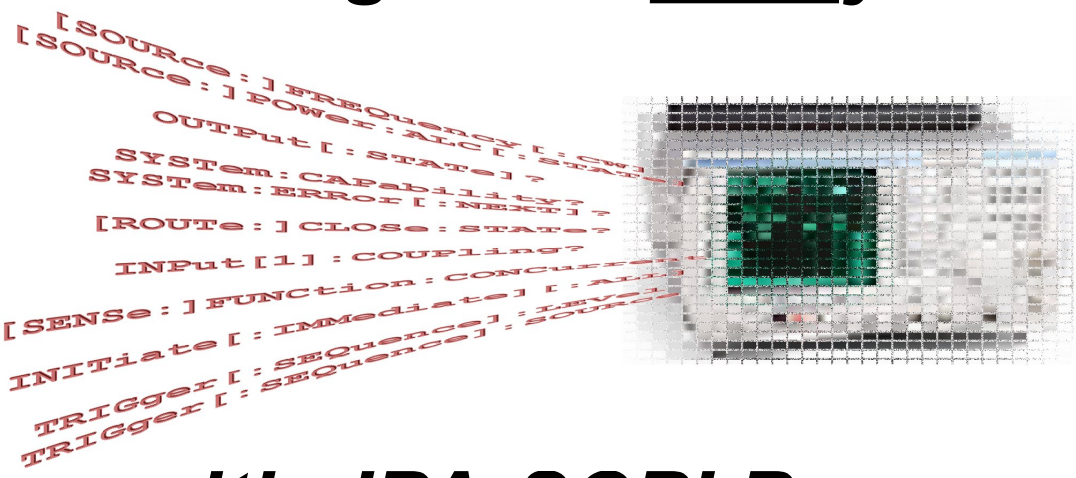
Your Instrument could be Talking SCPI Today...



...with JPA-SCPI Parser

Give your programmable instrument the professional finish that it deserves with a SCPI command interface!

SCPI is the command language of choice supported by all the major manufacturers of programmable instruments. Join their ranks with **JPA-SCPI Parser**.

By supporting SCPI both you and your customers will benefit:

- Familiar command language means faster learning curve for the customer
- Save time defining your own command language, and supporting customers afterwards
- Increased sales! Customers see SCPI support as a sign of a truly proficient manufacturer

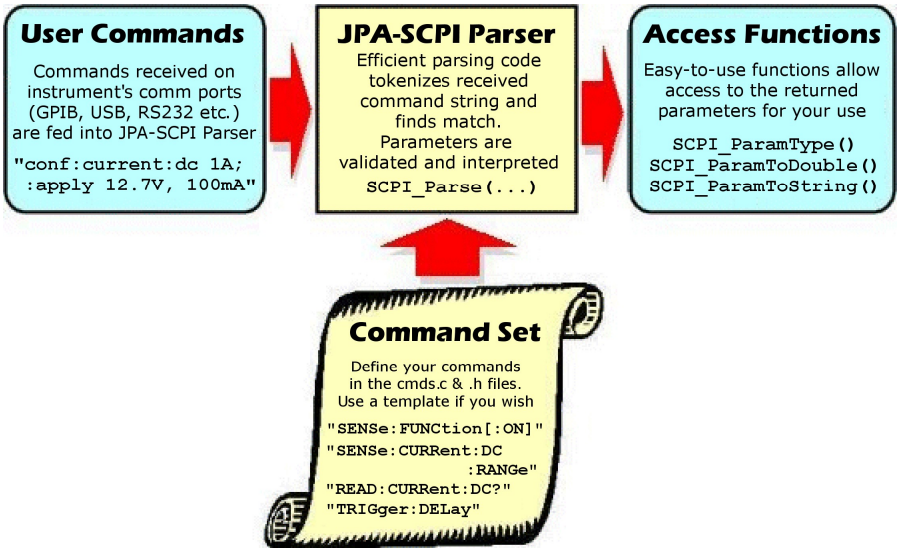
"I thought SCPI was too complicated. Don't we need a consultant?"

Not any more. No longer does supporting SCPI mean employing a consultant or spending days and weeks writing a SCPI parser in-house. By using JPA-SCPI Parser, your in-house developers are empowered to build a full SCPI parser for your instruments **quickly**, **easily**, and at **very low cost**.

Take a look at some of the highlights:

- JPA-SCPI Parser works on almost **any processor/hardware platform**:
  - Written in C, fully ISO/ANSI-compatible
  - Minimal ROM and RAM requirements
  - No particular compiler/processor requirements (e.g. no recursion used)
- **NO RUNTIME FEES**
- **Define your commands easily** using a set of constant array/structure tables
- **Templates** included for 10 popular types of instrument, plus a base SCPI template for all others
- **Full source code and documentation** supplied

How it Works...



Step 1: Define Your Command Set

If you wish, use one of the 10 instrument templates supplied as your starting point. Otherwise use the base template that contains all the base commands required for SCPI-compliancy. Now add your own commands. The set of supported commands is defined using a series of constant definition tables.

For example, say your command specification is this: **[SENSe:]VOLTage:DC:RANGe {<voltage range>}**

First add the command keywords to the command keywords table:

```
const char *SSpecCmdKeywords[] =
{
  /*                                Index
                                    -----          */
  :
  "[SENSe:]VOLTage:DC:RANGe",  /*   15             */
  :
};
```

Now add the parameter definition to the parameter specifications table:

```
const struct strSpecCommand sSpecCommand[] =
{
  /*                                Index
                                    -----          */
  :
  { REQ   NUM    sVolts    },  /*   15             */
  :
};
```

...where **REQ** means parameter is required (not optional), **NUM** means numeric value, and **sVolts** indicates the value is in volts, and will scale values entered in millivolts, kilovolts etc. to volts automatically.

Your instrument will now accept commands such as:

SENSE:VOLTAGE:DC:RANGE 1KV

VOLT:DC:RANG 1000

Volt:DC:Range 100mV

Step 2: Integrate Parser into Your Code

All you need to do now is pass the incoming command lines received by your instrument to JPA-SCPI Parser and let it get on with the validation, tokenizing and parsing.

a) Copy the command line received on your instrument's communication port to the **SCPI_Parse()** function. It returns a code indicating a matching command found or an error (invalid command, wrong parameter type, etc.)

b) If the command received was valid, then translate the command parameters into C variables using one or more of the included Access Functions: **SCPI_ParamToDouble()**, **SCPI_ParamToUnsignedInt()**, **SCPI_ParamToString()**, **SCPI_ParamToBOOL()**, etc.

c) Repeat steps (a) and (b) until all the commands in the command line are parsed.

Still Not Convinced?

This is what one of our customers had to say:

"I had the parser implemented in a test form in our embedded code the day after we received [JPA-SCPI Parser] and it would have been sooner but I didn't have time to look at it on the first day.

The following day our programmer that is dealing with the PC side of the software also had it running. We both found it very straight forward to use and the documentation is superb (unusual for software products)."

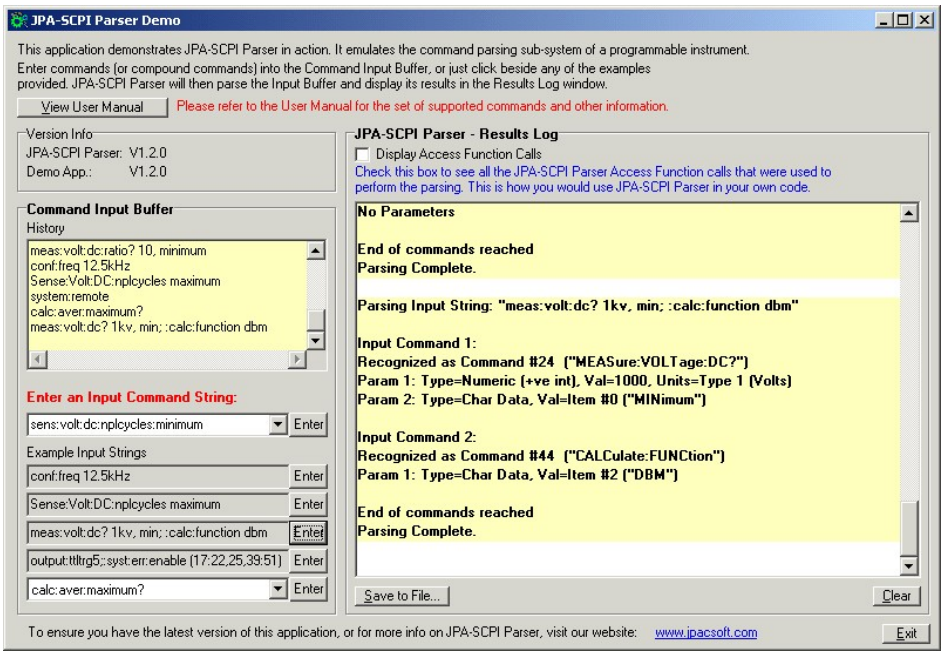
"Our product is a multichannel, modular instrument (we don't even know what some of the modules will be yet) and so we need a command language that can expand with it, which is why we chose SCPI.

"[JPA-SCPI Parser] will be of enormous use to us. We want to spend our time on writing the code that makes our product unique and not spend time writing the parser."

Malcolm Hartnell, Principal Engineer,
PerkinElmer Optoelectronics (opto.perkinelmer.com)

Try JPA-SCPI Parser for Free Now!

...with the JPA-SCPI Parser Demo Application



Representing the communications port of a programmable instrument, the application allows you to type in command lines and then see just how JPA-SCPI Parser handles them in its stride.

At the heart of the application is JPA-SCPI Parser itself—the same JPA-SCPI Parser that you will receive. Optionally, you can display all the Access Function calls made, showing exactly how JPA-SCPI Parser would **integrate with your own code**.

More than 70 commands are defined in the Demo Application, demonstrating all the different types of command and parameter you may need. Full documentation is also provided, including a list of all the supported commands.

Try it now. Download the Demo Application from our website: https://jpacsoft.com

JPA-SCPI Parser Demo Application requires a PC running any recent version of Windows (e.g. Windows XP, Windows 7, Windows 8, Windows 10).

JPA-SCPI Parser Specifications

SCPI Language Support

SCPI Version	V1999.0 (current version)
Long and Short-form Keywords	Yes, e.g. **VOLTage** allows **VOLT** or **VOLTAGE**
Compound Commands *(multiple commands separated by semi-colons)*	Yes
Optional Keywords	Yes, e.g. **[:AMPLitude]**
Default Nodes	Yes
Query Commands	Yes
Optional Parameters	Yes
Numeric Suffix(ces)	Yes, e.g. **OUTPut<*channel number*>:VOLTage:DC**
Parameter Types	**Numeric Values** (all variations as defined in SCPI, i.e.: integer, floating-point and scientific notation (up to and beyond ±9.9 e±37, regardless of processor/ complier limitations) **Character Data** (mnemonics such as **MINimum**, **DEFault**) **Booleans** **Strings** (delimted by single or double-quotes) **Unquoted Strings** **Numeric Lists**, e.g. **(12,15,17:20,22.8)** **Channel Lists** (single or multi-dimensional), e.g. **(@2!3:5!8,7!7)** **Expressions**[1]
Default Parameter Values	Yes
Multiple Parameter Types Allowed for same Parameter	Yes
Units	Numeric values can be followed by a unit suffix, e.g. **kV**, **GR**, **uH**. Unit Multipliers (**p**, **u**, **m**, **k**, **Ma**, **G**, etc.) automatically scaled to base units. User-definable units allow complete flexibility.
Number Bases	Numeric values can be entered in decimal, binary, octal or hexadecimal
Max Length of Command Line	255 characters or more (only limited by target processor/compiler)
Maximum Number of Commands that can be Specified	255 or more (only limited by target processor/compiler)

SCPI Instrument Class Templates Supplied

DC Voltmeter	Four-Wire Ohm Meter
AC Voltmeter	Digitizer (e.g. oscilloscope)
DC Ammeter	DC Power Supply
AC Ammeter	Signal Switcher (e.g. programmable switch)
Ohm Meter	RF Source (RF & microwave signal generators)
Base Template, containing the SCPI commands required by all types of instrument	

Source Code

Language	C, fully ANSI/ISO-compliant	
External Libraries Required	none	
Typical compile-size[2]	13.3KB ROM, 165 Bytes RAM[3]	
Access Functions Available	SCPI_Parse(); SCPI_ParamType(); SCPI_ParamUnits(); SCPI_ParamToCharDataItem(); SCPI_ParamToBOOL(); SCPI_ParamToString();	SCPI_ParamToUnsignedInt(); SCPI_ParamToInt(); SCPI_ParamToUnsignedLong(); SCPI_ParamToLong(); SCPI_ParamToDouble(); SCPI_GetNumListEntry(); SCPI_GetChanListEntry();

Notes

[1]Expressions are returned as a string pointer. Parsing of contents of expression parameters must be done by user's code

[2]Complie-size excludes space occupied by command set, and with optional support features disabled (optional support features are numeric suffix, numeric lists, channel lists and expressions—5KB ROM extra with all supported)

[3]These figures were obtained when compiling for a Microchip PIC18 using the Microchip XC8 compiler (optimizations on)

What You Get When You Buy JPA-SCPI Parser

- ☑ **C source code modules** comprising the SCPI parser itself and its Access Functions
- ☑ **Templates** for 10 different types of instrumentation (including measurement devices, e.g. digital meters, programmable source/supply devices, oscilloscopes)
- ☑ **Base Template** containing commands required by all SCPI instruments
- ☑ **Example Source Code**
- ☑ **Comprehensive User Manual** explaining how to use JPA-SCPI Parser for your particular instrument, and how to define your set of commands
- ☑ **Design Notes Manual** describing every function and data structure used in the source code
- ☑ An **unlimited** number of **royalty-free runtime licences** for your organization
- ☑ **Free product upgrades** for first year of ownership
- ☑ **Free technical support** via email for first year of ownership

Two Types of License Available

Single Brand	For **in-house developers**, this license allows your organization to develop software for as many instruments branded under the organization's name as you require.
Multi-Brand	For **consultancy houses or independent IT contractors**. This license entitles your organization to use JPA-SCPI Parser to develop software for as many instruments as you require, regardless of what organization name the instrument/software is to be branded under. Alternatively, you can purchase one single-brand licence per brand name.

Both types of license entitle your organization to:

- Make as many copies of JPA-SCPI Parser as you require for use within your own organization
- Distribute the compiled software produced using JPA-SCPI Parser without runtime fees

For a copy of our full terms and conditions, please visit our website or contact our sales department.

Pricing

License Type	US
Single Brand	$495
Multi-Brand	$1995

Prices correct at time of going to press. For latest prices visit https://jpacsoft.com

How To Order

Visit our website https://jpacsoft.com and order direct via a secure server. Many payment methods are accepted. Immediate delivery (24 hours per day) is available via electronic download, or on CD sent via airmail (US$20 extra).

Contact Us

Please contact us if you have any questions, or visit our website to find more information.

Website:	**https://jpacsoft.com**	**Enquiries:**	**sales@jpacsoft.com**

JPA Consulting Ltd.