SCPI measuring device

Jernej Dolžan[2], Dušan Agrež[1]

[1]*Faculty of Electrical Engineering, University of Ljubljana,*
Tržaška 25, 1001 Ljubljana, Slovenia
Phone: +386 1 4768 220, Fax: +386 1 4768 426, E-mail: dusan.agrez@fe.uni-lj.si
[2]*DOMEX elektronika d.o.o., Zg. Pirniče 45c, 1215 Medvode, Slovenija*

Abstract **–** Measurement instruments capable communicate with remote controller are mainly used in today measurement systems. In our case software has been written that turned the microprocessor based measuring board into SCPI measuring device (SCPI- Standard Commands for Programmable Instruments). We wrote a SCPI interpreter that enables compatibility with SCPI 1999 standard. First the software has been written that enables microprocessor module to communicate via IEEE 488 bus. Then some routines have been added that enable format compatibility between microprocessor and control software at the remote computer. The developed software is universal platform for supporting the open architecture technology of virtual instruments.

I. Introduction

In measurement technics instruments capable communicate with other instruments and remote controller over instrumentation bus are very often in use. Three buses are enforced: serial buses according to RS-485 and USB standards and parallel bus according to IEEE 488 standard.
We can note the same standardization trends in the field of programs and commands for control programmable test and measurement devices. Thanks to standardization it's easier to transfer software from instrument of one manufacturer to another or transfer program code between different instruments of the same manufacturer. In ideal case, it's possible to use the same user program on all instruments. All these are achievable with standard commands for programmable instruments - SCPI.
The instrument constructed according to IEEE 488 and SCPI standards [1-3] enables acting the software developed with SCPI commands. In our case SCPI interpreter for process-measurement module with DSP TMS320C30 [4,5] and TNT 4882 [6] interface chip for IEEE 488 bus has been developed.

II. Process-measurement module

Universal software has been developed for microprocessor based process-measurement modular system (Fig. 1.). Beside the central processing part with TMS320C30 processor there is also TNT4882 interface chip, which take care about transfer the data over General Purpose Interface Bus - GPIB (bus according to standard IEEE 488). The TNT4882 provides a single-chip Talker/Listener interface to the GPIB. It performs all low level functions like handshake of Talker and Listener, checking the correctness of transferred data, temporarily interrupting of transfer owing to buffer overloading, etc.
Microprocessor uses registers of the TNT4882 as 32 memory locations. For acting of measurement modular system like IEEE 488 measurement device the suitable numerical values must be entered in registers of the interface chip.

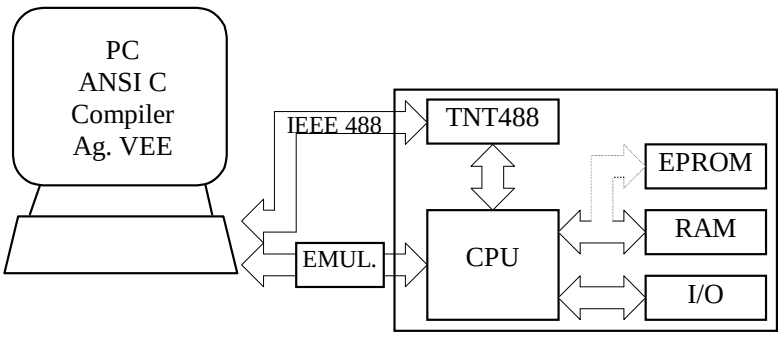


Fig. 1. Process-measurement module

Instrumentation board has been made that includes an IEEE 488 instrumentation bus interface TNT4882. The interface can transmit or receive the data (talk and listen mode) but it cannot be bus controller. The TNT4882 is able to make service request and interrupt host controller. It supports IEEE 488 Parallel Poll and Serial Poll. Measurement device to which it belongs can be settled and triggered by bus. This controller automatically reads or sends data to the bus according to the setup of its registers. What we have to do after initialisation is practically to read data from the FIFO buffer to the RAM or write data from RAM to the FIFO. There are three types of data exchange:

- Polling where CPU tries to send or receive data all the time. This method is time wasteful but in same cases this could be acceptable solution.
- Interrupt mode: main program is interrupted just at the time when data is received.
- DMA-Direct Memory Access: This mode is the fastest because data is stored from buffer directly to the RAM of the computer. DMA has also faster response time as interrupt mode. On the other hand we cannot control data while it is transferring. In our case we have to search specified character to temporary stop the transfer and partially interpret command line received yet.

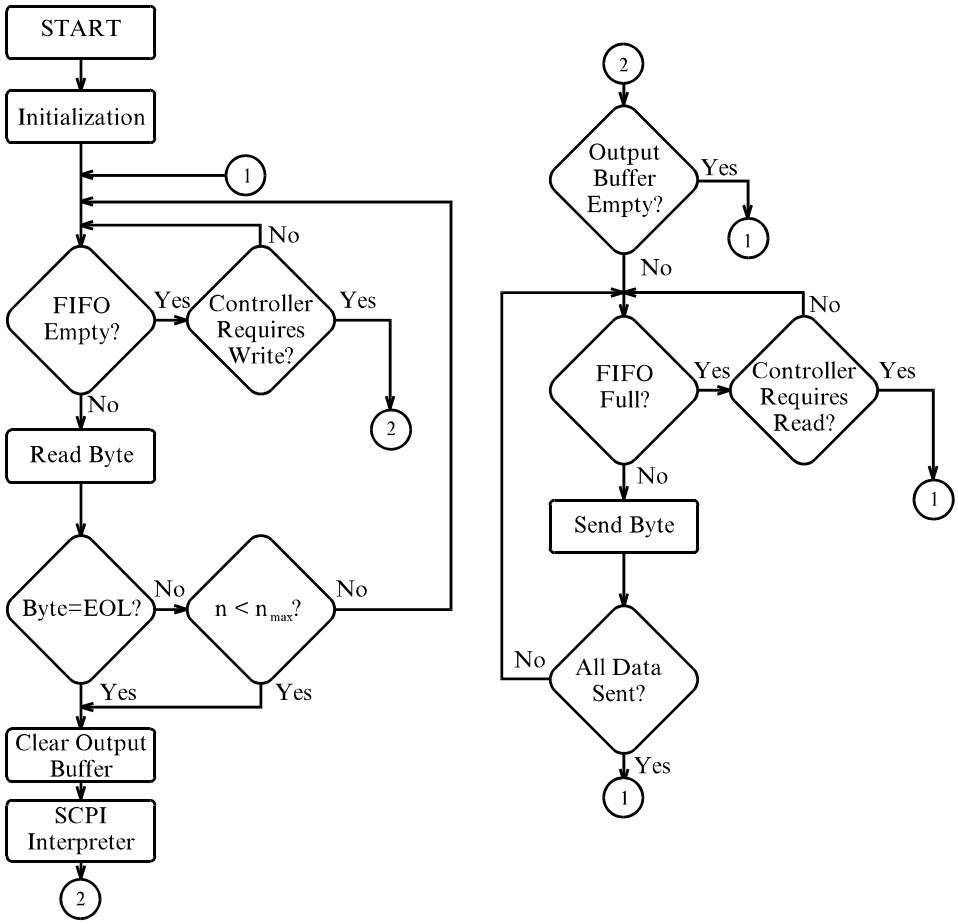


Fig. 2. Flow diagram of receiving and sending the data by IEEE 488 bus

Referring to above modes we decided to implement polling mode. This method is suitable for the nature of measurement module that requires sequence of command-execute-report commands. Besides that we use TNT4882 in one-chip mode what additionally speeds up data transfers using FIFO registers. If we want to put our measurement module into device capable communicating via IEEE 488 instrumentation bus we have to initialize TNT4882's registers first. We set our module to operate as Talker and Listener (Fig. 2).

III. SCPI - Standard Commands for Programmable Instruments

Standard commands for programmable instruments SCPI simplifies programming process of measurement instruments for both users and manufacturers. SCPI consists of standard ASCII

commands that are portable to any system that supports SCPI standard. The same command can cause the same function to execute even the instruments are different. It also enables easy replacement of old instruments (upgrade) with the instruments from the same or other manufacturer. SCPI command allows use of keywords or mnemonics. Mnemonic is a short form (4 characters) of a keyword determined by the SCPI regulations. Command can be <MEAS:ARR:VOLT?> or <MEASURE:ARRAY:VOLTAGE?>. SCPI does not differ upper and lower case characters except parameters. Command <MeaS:ARRaY:VOltaGe?> is also correct.
SCPI command structure tree is generated hierarchically what means one command consists of two or more mnemonics or keywords separated by ':' character. There are many reasons to use this type of structure. Instruments support many different commands so the "4 character commands" are not enough. Besides that tree structure enables use of one mnemonic many times. Its position in the tree presents the meaning of it. Some functions are in use often than others. It is important that such commands are made of short and simple form. SCPI allows headers where one or more mnemonics are missed. Interpreter automatically calls the appropriate service function.

MEASure:SCALar:VOLTage? or MEASure:VOLTage?

Commands for instrumentation control can be divided in two major groups, common commands (commands and queries mandated by IEEE 488.2 - Fig. 3a.) and SCPI commands (Standard commands - Fig. 3b.). Common commands mostly control instrument operations with meaning of checking instrument status registers. Common command consists of 4 or 5 ASCII characters starting with '*' mark. Command can have normal and query form (*IDN?, *RST, *CLR, . . .).

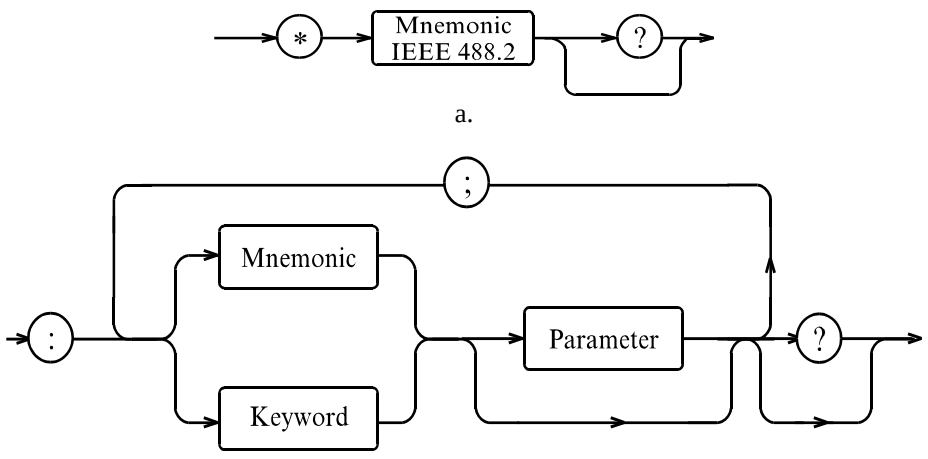


Fig. 3. Creating form of common commands and queries (a.) and general form of SCPI commands and queries (b.)

In general SCPI commands control instrument functions and enable usage of measurement functions. Each command consists of header and parameters. Header consists of one or more mnemonic or keywords separated by ':' character. As shown in figure query form of command and parameters are not obligatory.

IV. Interpreter

According to the regulations of SCPI standard we developed software that enables SCPI commands recognition and appropriate function calls (Fig.4).
Data delivered via IEEE 488 bus has to be processed by interpreter. Data array can consist of one or more command lines separated by EOL character. Interpreter recognizes these commands and calls service functions. First software finds out keyword or mnemonic that is a sequence of alphabetic characters ("space" and "TAB" are not included). Keyword or mnemonic ends when interpreter finds SCPI punctuation mark. At that point recognition can begin. Because of two types of commands, interpreter checks keyword for the first character. In case of character '*' the common command that consists of only one mnemonic or keyword is treated. If there are more keywords, common or SCPI, interpreter parses data array according to punctuation marks and sequentially calls service functions.

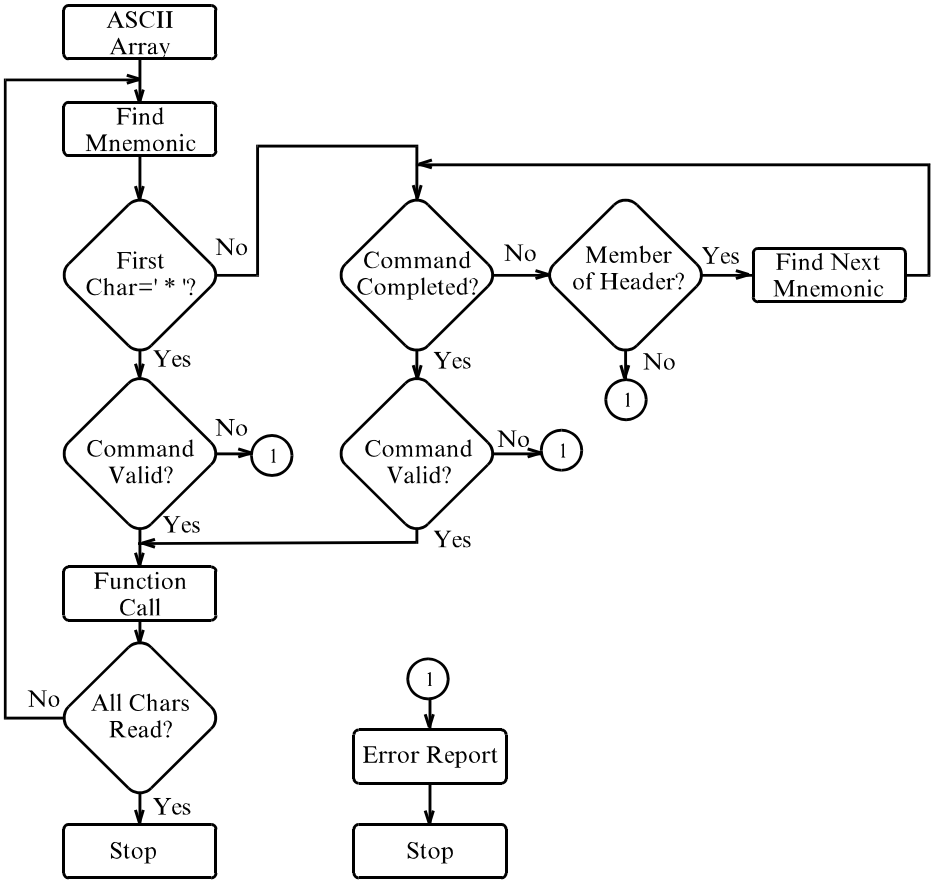


Fig. 4. Interpreter for SCPI commands in instrument control headers

Punctuation marks used by SCPI are:

- Character ';' marks end of common command. Character must be set at the end of parameters if command includes them. When the command in command line represents SCPI command the above punctuation mark means crossing of the SCPI command tree. It means that interpreter executes command completed by previous header only last mnemonic or keyword is replaced.
- The punctuation mark ':' separates two mnemonics or keywords. Interpreter finds out mnemonic that follows the mark and collects the header of the command together. Command is completed and executed after all mnemonics and parameters are recognized and compared in the SCPI tree structure. Interpreter checks each mnemonic if it belongs to its position in the SCPI command tree. In case of incorrect mnemonic or keyword the recognition is stopped. Commands right recognized till last command are executed normally. Error in command line only causes interruption of all commands following incorrect command including current command. Interpreter also calls error function to log event.
- Sequence of characters ';:' sets interpreter's pointer to the root of the SCPI command tree. Using this option more than one command in single command line is made possible.

All operations based on sampling of the signals and data processing. We wrote subsystem "TRACe" that enables creation of data arrays and writing/reading values from these data fields. Sampled and calculated values are stored in the appropriated data arrays created automatically or manually. Subsystem enables creation of user defined data fields or manipulation with fields created by other subsystems ("MEASURE").

MEASure:ARRay:VOLTage?

creates two one dimensional data arrays accessed by subsystem "TRACe":

CH1 - samples of the first A/D converter;
CH2 - samples of the second A/D converter.

Samples are held in buffer till the initialization is made. Data array can be deleted by command <TRACe:DELete name_of_field > or common command <*RST>.

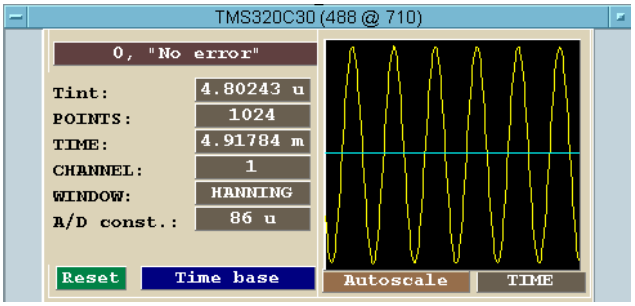
If we want to compute signal characteristics via FFT some additional data arrays have to be add. Each channel requires two - one for real and other for imaginary components of the FFT.

As mentioned, data can be transferred by many ways. SCPI allows use of all standard forms. Our module supports ASCII, INTeger and REAL (Floating Point) formats. By default the ASCII data format is in use because all headers must be coded in this form. Numerical parameters can be transformed to INT or REAL but in case of using small amount of numbers the profit is minimal. Besides we have to write conversion function to the user program too.
Main function of the measurement module TMS320C30 is to measure electrical signals via A/D converters, make fast calculations and send appropriate data to the control unit (Visual software on the computer).

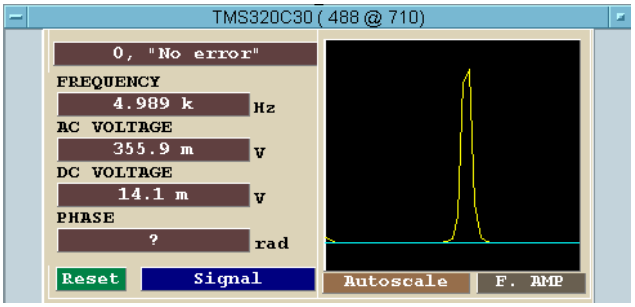
FORMat:
[DATA]
ASCii [{, Number_of_Decimal_Points}]
INTeger [{, Number_of_Bits}]
REAL [{, Number_of_Bits }]
DATA?
BORder:
SWAPped
NORMal
BORDer?

V. Virtual instrument

The developed software for TMS320C30 based measuring board is universal platform for supporting the open architecture technology of virtual instruments for which an application has been made.
With data-acquisition module (Burr-Brown: DSP102) supported by TMS320C30 processing module is possible to simultaneously sampling on two channels. A virtual instrument in window's environment has been developed for controlling of measure-process system (Fig. 5).



a.



b.

Fig. 5. Control panels for time base settings (a.) and parameters estimation (b.) on virtual instrument

This virtual instrument contains control panels for: device identification, time-base control (Fig. 5a.), marker settings, signal parameter estimation in time and frequency space (Fig. 5b.) and direct communication on SCPI level. All these functions enable very easy work with modular measurement-process system.

II. Conclusions

Software has been written that turned the microprocessor based measuring board into SCPI measuring device. According to the regulations of SCPI standard we developed software that enables SCPI commands recognition and appropriate function calls. The software has been written that enables microprocessor module to communicate via IEEE 488 bus with TNT4882 chip and some routines have

been added that convert a TMS320C30 data formats into ANSI data format. Finally some measuring functions have been built on measuring system with data acquisition module. A virtual instrument has been made for easy controlling of measuring operations.

References

[1] SCPI Consortium: SCPI 1999 - *Volume 1: Syntax and Style*, ver. 1999.0, USA, May 1999.
[2] SCPI Consortium: SCPI 1999 - *Volume 2: Command Reference,* ver. 1999.0, USA, May 1999.
[3] SCPI Consortium: SCPI 1999 - *Volume 3: Data Interchange Format,* ver. 1999.0, USA, May 1999.
[4] Texas Instruments: *TMS320C3x, User's Guide*, rev. L, USA, March 2004.
[5] Texas Instruments: *TMS320C3x Assembly Language Tools, User's Guide*, rev. D, USA, June 1998.
[6] National Instruments: GPIB TNT4882 Programmer's Reference Manual, USA, July 1995.