Relcom, Ltd.

DSP Software Support for Relcom PCDSP6 DSP Board

Design house for Digital Signal Processing and Automotive Control

URL: http://www.relcom.hu

Introduction

The PCDSP6 board has several peripherals such as base-band analog I/O, high-speed analog inputs, programmable gains and filters, high-resolution DDS-based clock generator, E1/T1/J1 interfaces, sync- and LVDS buses. All these peripherals can be reached via registers implemented in the on-board CPLD and FPGA chips. On the board, there are two high-performance C6416 TI DSP chips, which can access the CPLD and the FPGA registers.

Board Support Library (as an extension of Chip Support Library)

Typically, all of the DSP applications, written to the PCDSP6 boards, use some of the on-board peripherals. To make easier the access to the on-board peripherals, a Board Support Library (BSL) has been created. The BSL is similar to the TI's Chip Support Library (CSL). While the CSL provides a C-language interface for configuring and controlling on-chip peripherals, the BSL provides a C-language interface for configuring and controlling on-board peripherals. It consists of discrete modules that are built and archived into a library file. Each module relates to a single peripheral. The BSL granularity is designed such that each peripheral is covered by a single API module.

The benefits of the BSL include peripheral ease of use, shortened development time, hardware abstraction, and a level of standardization. The BSL uses and extends the HAL macros of the TI's CSL, providing a more flexible configuring possibility of the on-board peripherals' registers and let you generate faster DSP code. A complete symbolic description of all peripheral registers and register fields has been created. In general, the BSL makes it easier for you to get your system up and running in the shortest length of time.

Please note that the BSL provides access and symbolic description to the CPLD and the default FPGA registers only. If you change the FPGA program, either you can create the symbolic description for your new registers, or you have to use directly the addresses of your new registers.

Examples, test applications, tools

There are several DSP projects created by Code Composer Studio 3.1 to show how to access/use the on-board peripherals. Some of them have been used as low-level hardware test, while others are tools. Each example's source code is available.

BBAIO – Base-band analog input/output test

Introduces how to configure the BBAIO peripheral to sample two analog channels at 100 kHz and place the received samples to the analog outputs. The example has a fixed-address run-time control/status memory area, which can be read and written from the Host PC by the Monitor application. Using the control/status memory words, all the digital potentiometers on the board can be controlled in run-time to set the desired gain, gain offset and filter offset values.

DDS – DDS-based clock test

Shows how the DDS chip registers can be setup. The example has a typical configuration of the DDS chip. The generated clock signal is routed to the sync bus 0 for displaying it on an oscilloscope easily.

DigPot – Digital potentiometers hardware test

The example includes hardware test for the on-board digital potentiometer chips and software module tests for the implemented functions. The example sets and reads the potentiometer positions, saves and loads the set values into/from the potentiometers' non-volatile memories.

FPGAProg – FPGA programmer tool

The example has a "raw" DSP application that is able to program the FPGA. The "raw" application means that the generate COFF (.out) file of the FPGAProg project does not have any FPGA bit-code to download. This project can be used as an Input COFF file for the FPGA2COFF tool. For more details, see the section Programming the on-board FPGA.

FRAMER – E1/T1/J1 interface demo

It is a quite complex example to test both E1 interfaces at a time, invoking the on-chip EDMA peripheral to transfer data via the McBSP ports between the DSP chip and the E1 interface chips. The example can run on the DSP1 only. The aim of the

example is to see that the DSP chip can configure the interface chips via their memory-mapped registers and can transmit and receive data through the McBSP serial ports at high frequency. The example configures the interface chips to E1 mode with no loopbacks as well as applies ping-pong buffers for both receive and transmit data for each interface, respectively.

HADC – High-speed analog-digital converter test

The HADC example flushes the FIFOs of the high-speed analog input channels at the beginning of the application. Sets 1.25 MHz sample time for both high-speed analog input channels and sets 125 kHz sample (update) time for both base-band analog output channels. The example samples the high-speed analog input channels and places every 10^{th} samples on the base-band analog outputs. Using the Monitor application on the Host PC, you can check in run-time the FIFO status flags, reading the appropriate registers. The HADC can run on the DSP1 chip only.

HADC2 – High-speed analog-digital converter test 2

The HADC2 is an upgraded version of the HADC example. The HADC2 has a fixed-address run-time control/status memory area, which can be read and written from the Host PC by the Monitor application. Using the control/status memory words, you can select the sample time (even clocked by the DDS chip) for the high-speed analog input channels and can start the sampling. After the FIFOs of both channels are full, the sampling is stopped and the received samples are transferred into the DSP's memory to be able to read them periodically more times and to put them on the base-band analog outputs at 100 kHz. To get new samples in the FIFOs and put them on the outputs, the sampling has to be re-started by writing the proper control/status memory word. The HADC2 can run on the DSP1 chip only.

IT – Interrupt (stress) test

The Monitor application (running on the Host PC) uses this IT example to generate DSP-to-PCI interrupts periodically for hardware and software tests.

PDMA – "PCI DMA" test

In the Host PC's memory, there are two blocks reserved (not used by the Windows' Memory Manager) for the capability of transfer data blocks at high-speed between the DSP board and the host memory through the PCI bus. The PDMA example has to be downloaded and started on both DSP chips. The DSP program recognizes which DSP chip it runs in. The DSP program running in the DSP1 is the master and the other one in the DSP2 is the slave. Each DSP program has a fixed-address run-time control/status memory area, which can be read and written from the Host PC by the Monitor application. You should operate with the memory area of the master (DSP1). Sending commands to the master DSP program in the DSP1, it will control and

synchronize the slave DSP program in the DSP2 for simultaneous data block transfer tests when both DSP processors race for the PCI bus.

The following types of the tests are implemented:

- Data transfer in both directions between a single DSP and the host.
- Data transfer with both DSP processors at a time. Both DSP processors transfer data in the same direction at a time.
- Data transfer with both DSP processors at a time. The DSP 1 writes the host memory, while the DSP2 reads it.
- Data transfer with both DSP processors at a time. The DSP 1 reads the host memory, while the DSP2 writes it.

SDRAM – SDRAM hardware test

The example writes the entire SDRAM memory with several patterns in more rounds, and then, reads back the content of the SDRAM and compares it to the used patterns. The DSP application can run in both DSP chips.

TrnsRate – Data transfer rate measuring

The example performs data block transfers between different types of the DSP board's memories, and measures the elapsed time used for the data transfer. The TrnsRate example can reach the internal on-chip RAM of the DSP chip where the program runs as well as the attached external SDRAM. The DSP program can access to any address regions of the memory space of a DSP chip. You can even test the access-time of the FPGA registers, although it can be calculated according to the hardware manual.

The following types of the data transfers are implemented:

- C-compiler-optimized DSP routine copies data block.
- Half-way hand-optimized DSP routine copies data block, using four 64-bit registers.
- Half-way hand-optimized DSP routine copies data block, using eight 64-bit registers.
- EDMA copies data block.

The selected type of data transfers is run in more rounds to be able to get minimum, maximum, and average time values. The example can run in both DSP chips.

Tools

Some useful tools have been created to make easier to start a new DSP project, to program the on-board FPGA chip by a user-made (not default) FPGA bit-code, as well as to download a DSP application to the board and access the DSP chip's address space.

PGTool – Project generator tool

The DSP software support has a prepared directory structure. There is a subfolder for the include files of the modules implemented in the BSL. There is another one for the root directory of all the examples. The used TI libraries, such as the C run-time and the dspLib, are placed in a third subfolder. The PGTool has its own subfolder.

The DSP project files of all the examples have the same basic configuration to set the path of the include files, to find and add the TI's C run-time library to the project, to set certain compiler and linker settings.

Using the PGTool, you can easily create a new project within the examples' subfolder with the basic project configuration.

FPGA2COFF – Inserting FPGA bit-code into COFF file

The FPGA2COFF can insert an FPGA bit-code into the FPGAProg DSP project's output COFF file, generating a new COFF file that will be able to program the on-board FPGA with the used FPGA bit-code.

Programming the on-board FPGA

The on-board CPLD has a flash-based, while the FPGA has a RAM-based configuration memory. That is, after each power-on-reset, the FPGA has to be programmed. The FPGAProg example has a "raw" DSP application that is able to program the FPGA. The "raw" application means that the generate COFF (.out) file of the FPGAProg DSP project does not have the FPGA bit-code to download. Invoking the FPGA2COFF application, you can insert even the default FPGA program or your own FPGA program into the FPGAProg COFF file. The FPGA2COFF generates a new COFF (.out) file that contains both the FPGA bit-code and the FPGA programmer DSP routine. If you download and start this generated COFF file (including the FPGA bit-code) from the Host PC by the Monitor application, the FPGA is going to be programmed. Please note that only the DSP1 can program the FPGA, that is, you have to download the generated COFF file into the DSP1 chip. There are two debug LED's on the edge of the board. Each DSP can control its own LED only. The LED is used to provide visual information about the result of the FPGA programming. During programming, the LED is dark. If the programming was successful, the LED flashes with 20% duty and 1 sec period. That is, the LED is on for 200 ms, and it is off for 800 ms. If the FPGA

programming failed, the LED flashes with 80% duty. That is, it is on for 800 ms and off for 200 ms. The programming takes less than 1 sec.

Monitor – Accessing the DSP board via PCI bus

The entire memory spaces of both DSP chips of the PCDSP6 board can be accessed from the PCI bus. It means that, not only all the on-chip DSP registers and internal RAM can be read and written from the PCI bus from the Host PC, but all the on-board CPLD- and FPGA-realized registers and memories and FIFOs, too. The Monitor application provides some type of the access possibilities to the DSP board.

The Monitor application has the following key features:

- Provides basic information about the connected PCDSP6 board, such as PCI BAR addresses and the physical addresses and the length of the reserved DMA memory blocks in the host memory space.
- Provides very low-level (PCI register-level) access to the DSP board through the PCI bus.
- Provides memory dump and update to read and write all the memories (and memory-mapped registers) that the DSP chips can see, including the on-chip DSP registers, the on-chip internal RAM, the external SDRAM, the on-board CPLD and FPGA registers, memories.
- Can download DSP programs in both DSP chips.
- Can send soft reset to the DSP devices and can start the downloaded DSP program to run.
- Has interrupt and DMA tests.

Contact

If you have any question related to our products please do not hesitate to contact us

Relcom, Ltd.

Address: H-1112, Budapest, Táska u. 17., Hungary

Phone: +36 30 201 9919

Fax: +36 1 246 1240

Mail: info@relcom.hu

Web: http://www.relcom.hu