Configuring Emulation Parameters in Code Composer Studio v2.0

Daniel Rinkes *Software Development Systems*

ABSTRACT

This application report describes how to configure Code Composer Studio™ Integrated Development Envornment (IDE) v2.0 to work with different emulation parameters. In versions of Code Composer Studio prior to v2.0, these parameters were set through environment variables. With Code Composer Studio v2.0 and greater, the environment variables have been replaced by platform-independent configure variables specified in a new board.cfg file. This report will discuss the available configuration parameters, the format of the configuration files, and the procedure for using these files from within Code Composer Studio v2.0.

Requirements

- Microsoft Windows 98, 2000 or NT
- TI Code Composer Studio for Microsoft Windows (version 2.0 or greater)

Contents

List of Figures

1 Introduction

There are certain sets of emulation configuration parameters that may need to be set to debug a particular type of target. These configurations may correspond to the speed of the target clock, or other unique target specifics. If these parameters are not set correctly, Code Composer Studio will not be able to communicate with the target processor.

The configuration variable format is more flexible and more portable than the environment variable method. All of the parameter values are contained within a stand-alone configuration file. This allows you the flexibility of easily changing parameter values on the fly. It also offers portability between host PCs, host operating systems, and target processors.

2 Emulation Configuration Variables

There are a number of emulation configuration parameters available for use in the emulation configuration files. Many of these variables were available in previous versions of Code Composer Studio as environment variables. There are a few, however, that are new in Code Composer Studio v2.0 and were not previously supported.

The available configuration variables are:

- **UNIFY_TBCONLY**
  - Implemented to allow a trade-off between performance and continuous scan requirements.
  - Variable can be set to YES or NO. The default is NO.

    NO implies that the emulator will use both test bus controller (TBC) and TBC-XL features when available.

    YES implies that the emulator will use only TBC features, even if TBC-XL is available.
- **UNIFY_FASTCLK**
  - Implemented to allow specification of the output timing mode.
  - Variable can be set to YES or NO. The default is NO.

    NO implies that the output timing is normal. Test mode select (TMS) and test data output (TDO) will be output on the falling edge of TCLK. This choice will work with frequencies up to and above 10 MHz.

    YES implies that the output timing is fast. TMS and TDO will be output on the rising edge of the test clock (TCLK). This choice may be required for frequencies up to and above 20 MHz.
- **UNIFY_SLOWCLK**
  - Implemented to allow specification of the JTAG TCLK clock speed.
  - Variable can be set to YES or NO. The default is NO.

    NO implies that the JTAG clock is normal. The emulator will optimize for high performance. This choice will work with frequencies over 2 MHz.

    YES implies that the JTAG clock is slow. The emulator will optimize for slow clock operation at the expense of performance.

- **UNIFY_DLYMODE**
  - Implemented to allow use of a non default link delay.
  - Variable can be set to YES or NO. The default is NO.

    NO specifies that the default link delay for the emulator is used. The default value is 4 for the XDS510, and 6 for the XDS560.

    YES specifies that the value indicated in UNIFY_DLYSIZE should be used for the link delay.

    Link Delay is a measurement of the timing relationship between the emulator's output pins and input pins. To achieve acceptable signal timing at the DSP, flip-flops clocked off TCLK can be inserted into the signal paths. The effect of these flip-flops is referred to as link delay.
- **UNIFY_DLYSIZE**
  - Implemented to allow specification of a non-default link delay. The link delay allows resolution of timing delay issues between the emulator pod and the target.
  - Variable can be set to values between 0 and 31.
- **UNIFY_ECOMMODE**
  - Implemented to enable remote calls for the XDS560 to be made. This enables Microsoft Windows 32 (MS Win32) builds of XDSPROBE and XDSRESET utilities to run on both the XDS510 and XDS560 platforms.
  - Variable can be set to YES or NO. The default is NO.

    NO implies operating an XDS510.

    YES implies operating an XDS560.
- **UNIFY_TCLK_PROGRAM**
  - Implemented to allow specification of the PLL frequency.
  - Variable can be set to LEGACY, AUTOMATIC, or SPECIFIC.

    AUTOMATIC implies that the TCLK frequency will be auto-ranged up to the HS-RTDX limit (currently 35 MHz).

    LEGACY implies that the PLL will be programmed to use the legacy XDS510 frequency, which is 10.368 MHz.

    SPECIFIC implies that the TCLK frequency will be programmed to a specific frequency.
- **UNIFY_TCLK_FREQUENCY**
  - Implemented to allow the user to specify the TCLK frequency
  - There are acceptable numeric and string values.
  - DEFAULT – Use the default.
  - MINIMUM – Use the minimum internal frequency.
  - LEGACY – Use the legacy XDS510 frequency (10.368 MHz).
  - STANDARD – Use the standard TBC frequency.
  - EXCHANGE – Use the default HS-RTDX frequency (same as default).
  - MAXIMUM – Use the maximum internal frequency.

- 1.0 – 49 – Use exact whole-megahertz frequencies.
- 0.5 – 49.5 – Use exact half-megahertz frequencies.
- 0.75 – 49.75 – Use near (< 1%) and exact quarter-megahertz frequencies.

2.1 Examples for Setting TCLK Frequency

Auto-range up to the legacy frequency 10.368 MHz
UNIFY_TCLK_PROGRAM] ‘AUTOMATIC’
[UNIFY_TCLK_FREQUENCY] ‘LEGACY’

Auto-range up to 25.0 MHz
[UNIFY_TCLK_PROGRAM] ‘AUTOMATIC’
[UNIFY_TCLK_FREQUENCY] ‘25.0’

Select a specific frequency of 20.0 MHz
[UNIFY_TCLK_PROGRAM] ‘SPECIFIC’
[UNIFY_TCLK_FREQUENCY] ‘20.0’

- **POD_DRVR**
  - Implemented to provide the name of the adapter DLL for the SourceLess EPK or a DSK or EVM. This variable is required when using adapters.
  - Values can be any adapter DLL contained in single quotes.
    evm54xx.dll
    dsk6211.dll

3 Format of the Configuration File

The emulation variable configuration file is a text file that contains the list of emulation variables to be implemented. The convention is to name this file with an extension of .cfg. This is not a requirement. You are free to name the file with whatever name and file extension they choose.

Each emulation configuration file must begin with the same header. The header tells Code Composer Studio that the file is indeed an emulation variable configuration file. The header is:

;CFG-2.0

All of the configuration variable specifications will follow this header. Each configuration entry will begin with the name enclosed in square brackets. Following the variable name will be the user-defined values of the variables. Comments can also be added to the file. A semicolon is used to denote comments. The parser ignores all data on a line that follows a semicolon. A typical configuration file is shown in Figure 1.

```
;CFG-2.0
[UNIFY_SLOWCLK]   YES   ; Set the UNIFY_SLOWCLK variable to YES
[UNIFY_DLYMODE]   YES   ; Set the UNIFY_DLYMODE variable to YES
[UNIFY_DLYSIZE]   5     ; Set the UNIFY_DLYSIZE variable to 5
```

Figure 1. A Typical Code Composer Studio Emulation Configuration File

4 Integrating the Emulation Variable Configuration File With Code Composer Studio v2.0

Once the emulation variable configuration file has been created, the information needs to be included when configuring Code Composer Studio for the specific target. This can be done within Code Composer Studio Setup. Within CC Setup, when an emulation driver is added to “My System”, a Board Properties Box appears. It should look similar to Figure 2. You should select ”Auto-Generate board data file with extra configuration file” from the drop down box. The extra configuration file mentioned is the emulation variable-configuration file. You should then specify the path and name of the variable-configuration file text box. This file can also be located via the browse button.

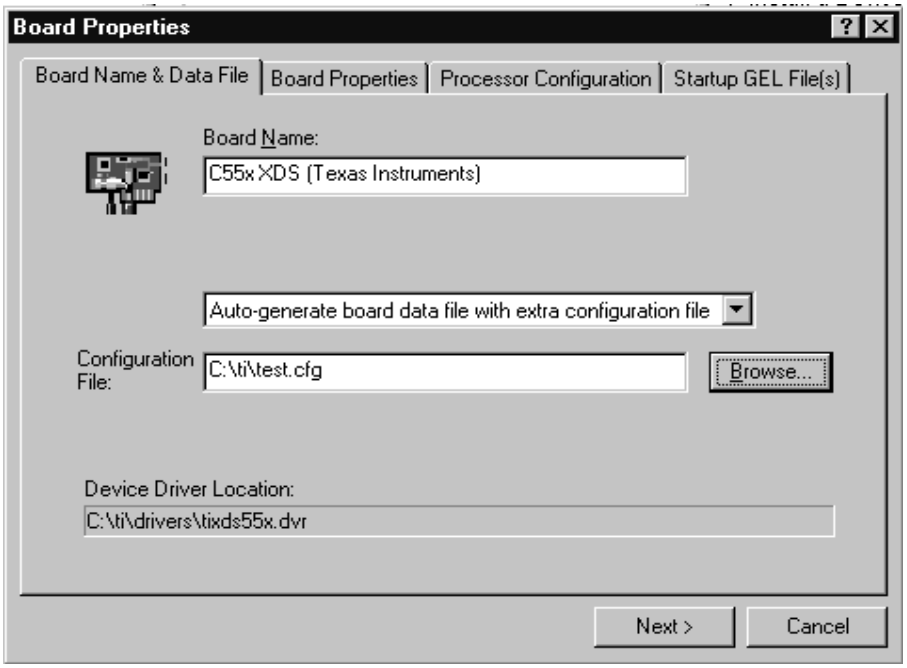

Figure 2. Specifying the Additional File in CCS Setup

Once this procedure is finished, the next steps through the configuration menu remain unchanged. Once the Board Properties, Processor Configuration, and Startup Gel File(s) menus are completed, you should save the configuration as normal.

4.1 Changing the Configuration File

If the configuration file is changed, the setup process must be performed again for the changes to take effect. The data in the configuration is only used whenever the save option is chosen from the CC Setup menu. Simply changing the configuration file will have no effect.

5 Hardware Emulators Example

A common application where the emulation variables need to be configured is the case where you wish to debug code on a hardware emulator (for example, a Quickturn emulator by Cadence Design Systems). The JTAG test clock (TCLK) on a hardware emulator runs slowly; thus, you must set the UNIFY_SLOWCLK variable appropriately. A typical configuration file used in configuring Code Composer Studio to debug on a hardware emulator is shown in Figure 3.

```
;CFG-2.0
[UNIFY_SLOWCLK]   YES    ; Set the UNIFY_SLOWCLK variable to YES
[UNIFY_DLYMODE]    NO     ;
;[UNIFY_DLYSIZE]   5      ; If UNIFY_DLYMODE is set to NO, this setting is ir-
relevant
```

Figure 3. Emulation Configuration for Use With a Hardware Emulator

The setting of UNIFY_DLYMODE and UNIFY_DLYSIZE can be dependent on the hardware emulator configuration. There have been cases where the default can be used, and cases where the values must be specified. The correct value can be found by experimentation with different values.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third–party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265