Release Notes for CrossCore Embedded Studio 1.1.1

Introduction

This document describes the changes for CrossCore Embedded Studio (CCES) 1.1.1. This release adds support for debugging the ADSP-BF706 EZ-KIT Mini™ via the on-board Debug Agent.

Supported Operating Systems

This release of CCES is supported on the following operating systems:

- Windows XP Professional SP3 (32-bit only)
- Windows Vista Business, Enterprise, or Ultimate SP2 (32-bit only)
- Windows 7 Professional, Enterprise, or Ultimate (32 and 64-bit)
- Windows 8.1 Pro or Enterprise (32 and 64-bit)

> **Note**
> Windows Vista, Windows 7, and Windows 8 users may experience User Access Control (UAC) related errors if the software is installed into a protected location, such as `Program Files` or `Program Files (x86)`. We recommend installing the software in a non-UAC-protected location.

System Requirements

Verify that your PC has these minimum requirements for the CCES installation:

- 2 GHz single core processor; 3.3GHz dual core or better recommended
- 1 GB RAM; 4GB or more recommended
- 2 GB available disk space
- One open USB port

> **Note**
> A faster disk drive decreases the build time, especially for a large amount of source files. 4GB of RAM or more will substantially increase the performance of the IDE.

New Functionality

Support for ADSP-BF706 EZ-KIT Mini™

This release will allow you to debug the new ADSP-BF706 EZ-KIT Mini™ via the on-board USB debug agent.

Visit www.analog.com/BF706EZKitMini for more information on the ADSP-BF706 EZ-KIT Mini™ as well as instructions for downloading its associated board support package.

Workarounds for ADSP-BF70x parts silicon anomaly 19000042

The CCES interrupt support for ADSP-BF70x parts has been modified to include the changes to workaround silicon errata 19000042.

Warning

- The workaround requires user exception 15 so this will not be available for application use when the errata workaround is enabled.
- The 19000042 workaround support in CCES 1.1.1 has significantly increased the number of processor-cycles to return from an interrupt. Some applications and ADSP-BF70x BSP examples will have to be modified to decrease sampling rates to compensate for slower interrupt handling.

The exception handler is registered by the default and generated startup code. If you have a custom startup code that is not updated automatically by CCES you will have to insert the call to register the handler manually. Refer to Blackfin/lib/src/libc/crt/basiccrt.s in your CCES 1.1.1 install and copy the following source from it into your custom startup file.

```
#include <sys/anomaly_macros_rtl.h>

#if WA_19000042
      // Register an exception handler for a reserved user exception as part
      // of the workaround for silicon errata 19000042 "Self-Nested Interrupts
      // Erroneously Disable Further Interrupts".
      .EXTERN _adi_rtl_anom_19000042_user_exception_handler;
      .TYPE   _adi_rtl_anom_19000042_user_exception_handler,STT_FUNC;
      LOADIMM32REG(R0, ADI_RTL_EXCEPTION_IID(WA_19000042_USER_EXCEPTION) )
      LOADIMM32REG(R1, _adi_rtl_anom_19000042_user_exception_handler)
      R2 = 0;
      CALL.X _adi_rtl_register_dispatched_handler;
#endif // WA_19000042
```

If your startup source is not regenerated automatically or a custom startup is not modified you will get an unhandled exception fatal error for every interrupt.

Workarounds for ADSP-BF70x parts silicon anomaly 19000044

The C runtime startup (CRT) has been modified to clear the `RTSEN` bit of `BP_CFG` before calling main.

Updates to Existing Functionality

Instruction Cache enabled on ADSP-BF70x processors

As of this release, you will receive a link-time warning if you attempt to disable the instruction cache, when building an application for ADSP-BF70x processors:

```
[Warning pp0018] "..\system\startup_ldf\app.ldf":59 'A loader initialization code to disable instruction
cache during booting is required. This message may be suppressed by defining the LDF macro
USING_ICACHE_DISABLE_INIT_CODE.'
```

This warning is emitted because these processors enable the instruction cache during the boot process, for maximum boot performance, and leave it in the enabled state - any application that uses the L1 SRAM/ICache space in SRAM mode will fail during booting when it comes to writing that section of memory. Such configuration mismatches are not evident during application development, since the emulator will disable instruction cache before loading the application, so the conflict may remain undetected until application development reaches the booting stage. The link-time warning against disabling instruction cache is intended to highlight the conflict sooner.

If you intend to use L1 SRAM/ICache in the SRAM mode, then you can disable the instruction cache at the start of the boot process, by using an initialization code (initcode); two such initcodes are provided as part of CCES 1.1.1, located in folder `Blackfin/ldr`:

- BF707_init_only_icache_disable_v00.dxe
  - Only disables instruction caching.

- BF707_init_icache_disable_v00.dxe
  - Disables instruction caching. Also initializes CGU and L3 for configurations matching the ADSP-BF707 Ez-board.

The link-time warning can be disabled by defining the link-time macro USING_ICACHE_DISABLE_INIT_CODE, either by:

- Using -flags-link -MDUSING_ICACHE_DISABLE_INIT_CODE to the command line.
- Or by adding USING_ICACHE_DISABLE_INIT_CODE to Project>Properties>C/C++ Build>Tool Settings>CrossCore Blackfin Linker>Preprocessor>Preprocessor Definitions

Note that instruction cache was disabled in silicon revision 0.0 of ADSP-BF70x processors, to address a silicon anomaly. This has been corrected in 1.0 silicon, so instruction cache will be enabled in 1.0 silicon.

elf2dyn problem with relocations for global array values fixed (CCES-10229)

Fixes a problem that causes the byte offsets created by `elf2dyn.exe` to be wrong for some global array data accesses and results in the dynamically loaded code using this data to be corrupted after `dyn_Relocate` is called.

Highlighted Limitations

ADSP-BF70x 1.0 silicon

CCES 1.1.1 recognizes only the 0.0 revision of ADSP-BF70x processors. ADSP-BF70x 1.0 will be classed as an "unrecognized" revision of the processor, so you must take extra steps, when using CCES 1.1.1 to build and run applications on ADSP-BF70x 1.0 silicon. ADSP-BF70x silicon revision 1.0 will be fully supported as a recognized processor in future revisions of CrossCore Embedded Studio.

When building applications for ADSP-BF70x 1.0 silicon with CCES 1.1.1 or earlier CCES releases, ensure that the application is built for silicon revision "any". This can be done either:

- By using the `-si-revision any` command-line switch;
- By selecting Silicon revision "any" when creating a new CrossCore Project;
- Or by selecting Silicon Revision "any" from the *`Settings > Processor Settings`* option page in the IDE.

Applications built for ADSP-BF70x revision 0.0 will not operate correctly on silicon revision 1.0, unless:

- The application was built for silicon revision "any";
- Or the application was built with the `-no-utility-rom` switch.

If your ADSP-BF70x 0.0 application was not built using one of these options, the application will access invalid Utility ROM locations when executed on ADSP-BF70x 1.0 silicon, and will fail to run correctly.

PLLCLK Restriction

The ADSP-BF70x processors have a PLLCLK restriction which is based on the desired Core Clock and SYSCLK frequencies. The ADSP-BF70x datasheet specifies the PLLCLK restrictions. The Power Service Add-in has the capability to change the Core Clock, SYSCLK and PLLCLK frequencies. The power service API `adi_pwr_SetFreq()` does not currently follow the PLLCLK restrictions outlined in the datasheet. It's recommended that you not use the `adi_pwr_SetFreq()` API but instead use the `adi_pwr_SetClkControlRegister()` and `adi_pwr_SetClkDivideRegister()` API's. The following power service code will set clock frequencies within specification:

```
adi_pwr_SetClkDivideRegister(0, ADI_PWR_CLK_DIV_CSEL, 2);   /* 400 MHz core */
adi_pwr_SetClkDivideRegister(0, ADI_PWR_CLK_DIV_SYSSEL, 4); /* 200 MHz sysclk */
adi_pwr_SetClkControlRegister(0, ADI_PWR_CLK_CTL_MSEL, 32); /* 800 MHz PLL */
```