1

Embedded Target for TI C6000 DSP 2.0 Release Notes

New Features

This section is organized into the following subsections:

Note The Developer's Kit for Texas Instruments™ DSP has been repackaged as two separate products, including the Embedded Target for TI C6000 and the MATLAB Link for Code Composer Studio™.

Two Virtual Targets Added

Embedded Target for TI C6000 DSP adds support for two virtual target, the C6211 DSP Starter Kit and the C6201 Evaluation module. Both are virtual because you use the C6711 DSK and C6701 EVM with fixed-point blocks to emulate the C6211 DSK and C6211 EVM.

Added C62x DSP Library

The blocks in the C62x DSP Library correspond to functions in the Texas Instruments TMS320C62x DSP Library assembly-code library, which target the TI C62x family of digital signal processors. You can use these blocks to develop simulations by building models in Simulink before generating code. Once a model is developed, you can invoke the Real-Time Workshop to generate code that is optimized to run on the C6711 DSK or C6701 EVM. During code generation, each C62x DSP Library block in your model is mapped to its corresponding TMS320C62x DSP Library assembly-code routine to create

target-optimized code. The C62x DSP Library blocks generally input and output fixed-point data types.

The following table list each block in the C62x DSP Library.

Block	Description
Conversions	
Convert Floating-Point to Q.15	Convert a floating-point signal to a Q.15 fixed-point signal
Convert Q.15 to Floating-Point	Convert a Q.15 fixed-point signal to a single-precision floating-point signal
Filters	
Complex FIR	Filter a complex input signal using a complex FIR filter
General Real FIR	Filter a real input signal using a real FIR filter
LMS Adaptive FIR	Perform least-mean-square adaptive FIR filtering
Radix-4 Real FIR	Filter a real input signal using a real FIR filter
Radix-8 Real FIR	Filter a real input signal using a real FIR filter
Real Forward Lattice All-Pole IIR	Filter a real input signal using an auto-regressive forward lattice filter
Real IIR	Filter a real input signal using a real auto-regressive moving-average IIR filter
Symmetric Real FIR	Filter a real input signal using a symmetric real FIR filter
Math and Matrices	

Block	Description
Autocorrelation	Compute the autocorrelation of an input vector or frame-based matrix
Block Exponent	Return the minimum exponent (number of extra sign bits) found in each channel of an input
Matrix Multiply	Perform matrix multiplication on two input signals
Matrix Transpose	Compute the matrix transpose of an input signal
Reciprocal	Compute the fractional and exponential portions of the reciprocal of a real input signal
Vector Dot Product	Compute the vector dot product of two real input signals
Vector Maximum Index	Compute the zero-based index of the maximum value element in each channel of an input signal
Vector Maximum Value	Compute the maximum value for each channel of an input signal
Vector Minimum Value	Compute the minimum value for each channel of an input signal
Vector Multiply	Perform element-wise multiplication on two inputs
Vector Negate	Negate each element of an input signal
Vector Sum of Squares	Compute the sum of squares over each channel of a real input
Weighted Vector Sum	Find the weighted sum of two input vectors
Transforms	

Block	Description
Bit Reverse	Bit-reverse the positions of the elements of each channel of a complex input signal
FFT	Compute the decimation-in-frequency forward FFT of a complex input vector
Radix-2 FFT	Compute the radix-2 decimation-in-frequency forward FFT of a complex input vector
Radix-2 IFFT	Compute the radix-2 inverse FFT of a complex input vector

Fixed-Point Code Generation of Product, Sum, and Gain Blocks

The built-in Simulink™ blocks Gain, Product, and Sum now generate code specifically for fixed-point code generation. This adds more support for generating true fixed-point capable code from the Embedded Target for TI C6000 DSP.

Support for Adding DSP/BIOS™ to Projects

With this release, you can generate code that incorporates DSP/BIOS™ modules and the DSP/BIOS API. Adding the DSP/BIOS functionality lets you take advantage of the modules and tools in DSP/BIOS to provide multithreading, real-time analysis, optimization, and other instrumentation in your Code Composer Studio™ (CCS) projects and executable files. For more information about adding DSP/BIOS to your generated code, refer to “Using DSP/BIOS” in the Embedded Target for TI C6000™ DSP User’s Guide.

ADC and DAC Block Requirement Removed

By adding a scheduler to the target software (synchronized to the clock on the board), we have eliminated the requirement that you include at least one ADC or DAC block in the models you develop for targeting. When your model does not include the ADC or DAC blocks, the new scheduler provides the interrupts to trigger events in your model instead of relying on the DMA on the board. Note that the interrupt scheduler assumes that the signal processor clock is running at 100 MHz for the C6701 EVM target or 150 MHz for the C6711 DSK

target. If the clock rate does not match the assumed rate, the sample rates in your model will be incorrect and your model will generate incorrect results.

Model Profiling Capability Added

When you build CCS projects that include DSP/BIOS, the Embedded Target for TI C6000 DSP software provides the ability to profile the operation of your generated code.

- If you created your project in CCS or you are using an existing CCS project, profiling returns information about the code and how it executes on the target.
- If you created your project and executable from a Simulink model, `profile` analyzes your model performance down to the atomic subsystem level and reports the results in an HTML file.

Invoke the profile capability either from the **RTW Options** panel in the **Simulation Parameters** dialog in Simulink, or from the MATLAB command line using `profile`. For more information on profiling generated code, refer to "Profiling Generated Code."

Added Support for Multirate Models Through a Software Scheduler

Now you can create models for targeting that use more than one processing rate. The processing rate of your model can change from the model base rate, such as using decimation or interpolation in your algorithm.

One important note—the target software assumes that your target C6000 digital signal processor (DSP) is running at the factory default rate of 100 MHz when your target is the C6701 EVM and 150 MHz for the C6711 DSK. The scheduler calculates all the interrupt rates for your model using the clock rate. If you change the rate of the DSP clock, the interrupt rates in your model will still be calculated with the base rate, not the one you set. As a result, the rates in your model will be wrong and the results will not be correct. For example, if your model contains a Sine block running at 1 KHz sample rate and your target is your C6701 EVM, the scheduler uses the 100 MHz rate to calculate the interrupt timing to generate the sin wave sampled at 1KHz :

interrupt rate = DSP clock rate/Sine block sample rate

= 100 MHz/1KHz

= 100 KHz

yielding a sample period of 10 μsec, one interrupt sent to the sine wave generator every 100000 clock cycles.

If your actual clock rate on your C6701 EVM has been reset to 150 MHz, the 10 μsec period is wrong and the sine wave is generated incorrectly.

Inline DSP Blockset Functions Option Added

Code generated from blocks in DSP Blockset use functions in a static run-time library. With this option, you can elect to inline those functions in your generated code. Inlining the functions creates more optimized code at the expense of some increased memory used.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving from the Developer's Kit for Texas Instruments ™ DSP 1.2 to Embedded Target for TI C6000 DSP Version 1.0.

Note The Developer's Kit for Texas Instruments™ DSP has been repackaged as two separate products, including the Embedded Target for TI C6000 DSP and the MATLAB Link for Code Composer Studio.

To upgrade your current Developer's Kit to Embedded Target for TI C6000 DSP Version 1.0, do the following tasks in order:

- Install the appropriate version of Texas Instruments Code Composer Studio 2.1. Note that the Embedded Target does not support targeting with the C5000 DSP family.
  - To target C6000 hardware, install the Code Composer Studio Version 2.1 for TMS320C6000.
  - To link to C6000 hardware, install the Code Composer Studio Version 2.1 for TMS320C6000.
- Download and apply a patch for DSP/BIOS—"CCStudio v2.1 - Config Server Registration Fix"—to Code Composer Studio 2.1. The patch is available only from the following URL:

  ```
  <https://www-a.ti.com/downloads/sds_support/
  ALL-2.00-SA-to-TI-BIOSRTA-REG01.htm>
  ```
- Install MATLAB Link for Code Composer Studio™ Development Tools 1.0
- Install Embedded Target for TI C6000 DSP Version 1.0

If you are upgrading from a version earlier than 1.2, refer to "Upgrading from an Earlier Release" on page 2-9.

Known Software and Documentation Problems

This section includes a link to a description of known software and documentation problems in Version 2.0.

For a list of bugs reported in the previous release that remain open, see "Known Software and Documentation Problems" on page 2-10.

Targeting

- For reasons related to the Texas Instruments C compiler, you cannot use certain Simulink blocks for targeting. They are
  - Singular Value Decomposition block in the library dspfactors
  - Pseudoinverse block in the library dspinverses
  - SVD Solver block in the library dspsolvers

  These blocks are not included in the run-time libraries and do not build when you generate code from your model.
- For this of the Embedded Target, the LED overrun indicator has been removed.

Demos

In MATLAB Demos (select **Help->Demos** from the MATLAB menu bar), the Embedded Target demonstration program LMS Adaptive Filtering runs slowly on DSP Starter Kit (DSK) boards. Design features of the board cause the slow processing across the RTDX link. Using the Texas Instruments XDS 510 JTAG/PC Controller, or an equivalent emulator, to connect your host to the DSK can alleviate the problem.

Data Types

C5000 and C6000 family processors handle integer data types differently. On C5000 family processors, 8-bit values do not have unique addresses. The C5000 family processors use 16-bit addresses natively. When you read and write to C5000 family processors, 8-bit integers in MATLAB are stored as 16-bit integers on the processor. In detail, the differences are

- When you write an 8-bit integer from MATLAB to a C5000 family processor, MATLAB pads the value to 16 bits and the processor stores the value as a 16-bit integer.

- When you read an 8-bit integer from the processor into MATLAB, the read function takes 16 bits from processor memory, discards the eight most significant bits, and returns the remaining eight bits to MATLAB as the integer value.

For these reasons, scripts that run on C6000 family processors may not run without modification on C5000 family processors.

General Target Operations

When your target digital signal processor is running, CCS uses up to 99% of the CPU cycles on your PC. Consequently, some other operations may be very slow, such as running the MATLAB help system or processing other applications.

LED Overrun Indicator

For this version of the product, the LED overrun indicator available in earlier product versions is not included.