Application Report
SPRA518

TMS320C6x Code Development Flow

Digital Signal Processing Solutions

Abstract

This document descibes the process used to develop and compile C code for the Texas Instruments (TI™) TMS320C6x digital signal processor (DSP).

Step 1: Obtain/Write, Compile, Validate Native C Code on Workstation

The purpose of this step is simply to obtain standard portable C code according to specifications. This should be compiled and validated with any available test vectors on the workstation (PC, Sun, HP, etc).

Step 2: Compile Native C Code with the C6x Compiler

- ❒ Compile and verify test vectors with C6x compiler with -o switch to invoke software pipelining and -mw switch to invoke software pipeline loop feedback.
- ❒ Profile to determine the important, high MIP areas of the code.
- ❒ Try -pm, -o3, and -mt options to improve loops.
- ❒ Declare all read only variables as const.
- ❒ Make sure all 32-bit values are defined as int (not long) and 16-bit values are defined as short.
- ❒ Define all loop counters to be int.
- ❒ Note memory bank hits (with release 1.1 of the simulator) as these might need to be avoided by using linear assembly.
- ❒ For any important area that does not satisfy performance, proceed to step 3.
- ❒ For loops where code size needs to be improved proceed to step 7.

Step 3: Optimize C Code for C6x

- ❒ Use intrinsics to define specific C6x operations.

- ❒ Use intrinsics and casting short arrays to int arrays to enable word access for short data.
- ❒ Use _nassert intrinsic to pass loop count information where necessary.
- ❒ Unroll loops when C6x resources are unbalanced; i.e., when there are 3 multiplies in one iteration, 2 cycles are required; but six multiplies requires only three cycles.
- ❒ Profile to determine improved performance, if any.
- ❒ Note memory bank hits (with release 1.1 of the simulator) as these might need to be avoided by using linear assembly.
- ❒ For any important area that does not satisfy performance, proceed to step 4.
- ❒ For loops where code size needs to be improved proceed to step 7.

Step 4: Write Linear Assembly

- ❒ Write C6x linear assembly without functional units using variable names.
- ❒ Add mptr directives as necessary for avoiding memory bank hits.
- ❒ Draw dependency graph to determine if there is a live-too-long, loop carry path, or split-join path problem.
  - Split-join path—insert a move or unroll loop.
  - Live-too-long problem (also specified in loop feedback)—unroll loop.
  - Loop carry path—determine if minimum loop carry path has been achieved.
- ❒ Check to see if partitioned resource bound is greater than resource bound. This implies a partitioning problem so go to step 5.
- ❒ Determine if MII has been achieved (shown in loop feedback).
- ❒ For any important area that does not satisfy performance, proceed to step 5.
- ❒ For loops where code size needs to be improved proceed to step 7.

Step 5: Add Partitioning Information

- ❒ Add partitioning to the linear assembly. Try to split the dependency graph to balance operations on each side and also limit the number of cross paths used.
- ❒ If MII has still not been achieved try setting all functional unit values, balancing operations among individual units as evenly as possible.
- ❒ If II is still not good enough, contact Tools Support for more help.
- ❒ For loops that have an important outer loop that is not software pipelined, proceed to step 6.
- ❒ For loops where code size needs to be improved proceed to step 7.

Step 6: Software Pipeline Outer Loop

Future versions of the tools will software pipeline outer loop automatically. For now, this will have to be done by hand if necessary. Also, combining loop setup with prologue will be done by the tools in the future but for now this will have to be hand coded if necessary.

For loops where code size needs to be improved proceed to step 7.

Step 7: Reduce Code Size

Future versions of the tools will reduce prologue and epilogue automatically. For now, this will have to be done by hand, if necessary. You can remove epilogues by hand and execute the loop more times. The same can be done with prologues as long as some initial instructions are conditionally suppressed the first one or more iterations.

Summary

All of the techniques and steps described above are documented in the *TMS320C6000 Programmer's Guide* (TI literature number SPRU198C). It is highly recommended that you read the Programmer's Guide, especially the "Optimizing C" and "Optimizing Assembly" chapters.

TI Contact Numbers

INTERNET

TI Semiconductor Home Page
www.ti.com/sc

TI Distributors
www.ti.com/sc/docs/distmenu.htm

PRODUCT INFORMATION CENTERS

Americas
Phone +1(972) 644-5580
Fax +1(972) 480-7800
Email sc-infomaster@ti.com

Europe, Middle East, and Africa
Phone
- Deutsch +49-(0) 8161 80 3311
- English +44-(0) 1604 66 3399
- Español +34-(0) 90 23 54 0 28
- Francais +33-(0) 1-30 70 11 64
- Italiano +33-(0) 1-30 70 11 67

Fax +44-(0) 1604 66 33 34
Email epic@ti.com

Japan
Phone
- International +81-3-3457-0972
- Domestic 0120-81-0026

Fax
- International +81-3-3457-1259
- Domestic 0120-81-0036

Email pic-japan@ti.com

Asia
Phone
- International +886-2-23786800
- Domestic
  - Australia 1-800-881-011
    - TI Number -800-800-1450
  - China 10810
    - TI Number -800-800-1450
  - Hong Kong 800-96-1111
    - TI Number -800-800-1450
  - India 000-117
    - TI Number -800-800-1450
  - Indonesia 001-801-10
    - TI Number -800-800-1450
  - Korea 080-551-2804
  - Malaysia 1-800-800-011
    - TI Number -800-800-1450
  - New Zealand 000-911
    - TI Number -800-800-1450
  - Philippines 105-11
    - TI Number -800-800-1450
  - Singapore 800-0111-111
    - TI Number -800-800-1450
  - Taiwan 080-006800
  - Thailand 0019-991-1111
    - TI Number -800-800-1450
  - Fax 886-2-2378-6808
  - Email tiasia@ti.com

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

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

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE (“CRITICAL APPLICATIONS"). TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF TI PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty, or endorsement thereof.