TMS320 DSP
DESIGNER'S NOTEBOOK

Bootload of C Code for the TMS320C5x

APPLICATION BRIEF: SPRA235

Jason Chyan
Digital Signal Processing Products
Semiconductor Group

Texas Instruments
May 1994

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software 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 application 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, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

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

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI 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.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Contents

Bootload of C Code for the TMS320C5x

Abstract

This document discusses how boot code can be generated using C code. Specifically addressed is how to use the `-c` option in the linker to build a single code section that includes the `.text, .cinit, .bss,` etc., sections that you want to be in the boot code.

Design Problem

How can I generate my boot code with C?

Solution

Use the `-c` (not `-cr`) option in the linker and build a single section that includes the `.text, .cinit, .bss,` etc., sections that you want to be in the boot code. Then use DSPHEX to convert this single section into boot code. Following is an example linker command file to link several sections and `.cinit` into one output section.

```
-c
-o filename.out
-m filename.map
filename.obj
-stack 64
-l rts50.lib
-l flib50.lib
MEMORY
{
   PAGE 0: PROG: origin = 0x0800, length = 0x1a00
   PAGE 1: DATA: origin = 0x0060, length = 0x0020
}
SECTIONS
{
   bootsect: {
            rts50.lib(.text) = 0800h
            *(.text)
            .=e00h;
            .cinit=.;
            *(.cinit)
            .+= 1;
            .=00f00h;
            *(.const)
            .=01000h;
            *(.stack)
            .=01040h;
            *(.bss) } load=0800h PAGE 0
}
```

The command file for the DSPHEX will be:

```
filename.out
-t
-bootorg 08000h
SECTIONS { bootsect = boot }
```

The program entry point of C code is `_c_int0.` Therefore, in the linker command file the `_c_int0` has to be assigned the starting address. This was done by first line in the SECTIONS:

```
rts50.lib(.text) = 0800h
```

Since the .cinit section was hidden in another section, we need to make it visible to the linker by:

```
cinit=.;
*(.cinit)
.+=1;
```

The commands in the SECTIONS assign a starting address to each input section and it is relative to the starting address of first section. This means that .cinit starts from 0x1600, .const starts from 0x1700, .stack starts from 0x1800, and .bss starts from 0x1840. If you don't want to generate unused space in between each section, you can remove the ".=0xxxxh;" command and all the sections will be placed consecutively. When you link the file with the example linker command file above, you will get the following warning messages "out-put file has no .text section" and "output file has no .bss section." You can ignore these messages.