Application Report
SPRAAL5 – August 2008

Using the ROM Image on DA830 with RTSC Content

Chris McCormick

ABSTRACT

The Primus DA830 device includes DSP/BIOS 6 in its on-chip ROM. Placing DSP/BIOS 6 in the ROM frees RAM space and improves performance, but sacrifices some configurability. This document explains how to use the DSP/BIOS ROM image in an application.

Contents

1 Introduction

The Primus DA830 device includes DSP/BIOS 6 in its on-chip ROM. Placing DSP/BIOS 6 in the ROM frees RAM space and improves performance, while sacrificing some configurability. The main limitation when configuring against a ROM image is that some RTSC module configuration parameters are “frozen” when the ROM was created, and cannot be modified.

RTSC is a standard for reusable software components. XDCtools is a set of tools that enable the creation and use of RTSC packages.

2 Using the DSP/BIOS ROM Image

To use the DSP/BIOS ROM, add a single line to your application’s RTSC configuration file (.cfg file). *The following line must be the first line in your .cfg file*:

```
Program.importRomAssembly('ti.sysbios.rom.da830.romimage');
```

The rest of your RTSC configuration script will most likely remain unchanged.

Including the “Program.importRomAssembly” command causes the RTSC configuration to bring in the ROM assembly, and any DSP/BIOS code you reference will be referenced from the ROM.

The DSP/BIOS 6 ROM code, and all of the read/write data structures that support the ROM code are located at the addresses shown in the following table.

Resource	Address
ROM code	0x11700200
RAM data structures	0x11800020

2.1 Modules in the ROM

The ROM image contains a set list of DSP/BIOS modules, shown in the following table. If your application uses modules that are not in this list, they will be brought into RAM.

Table 1. ROM modules

ti.sysbios	**ti.sysbios.knl**
BIOS	Clock
	Idle
ti.sysbios.family.c62	Swi
IntrinsicsSupport	Task
TaskSupport	
	ti.sysbios.misc
ti.sysbios.family.c64p	Intrinsics
Cache	Queue
Exception	
Hwi	**ti.sysbios.timers.timer64**
TimestampProvider	Timer
ti.sysbios.family.c64p.primus	**xdc.runtime**
TimerSupport	Assert
	Core
ti.sysbios.gates	Defaults
GateHwi	Diags
GateMutex	Error
	Gate
ti.sysbios.hal	Log
Hwi	LoggerBuf
Timer	Main
	Memory
ti.sysbios.heaps	Startup
HeapMem	SysMin
	System
ti.sysbios.ipc	Text
Event	Timestamp
Semaphore	

2.2 Frozen Module Configuration Parameters

There is an important limitation to consider when using the ROM image. Many of the module configuration parameters for the modules in the ROM image have been “frozen” to specific (usually default) values, and cannot be changed by your configuration.

The reason for this is an important optimization in RTSC code generation. When XDCtools generate code based on the client’s configuration, the module configuration parameters are generated as constants. The compiler can then fold these constants into the code, so that the code is optimized as though it were written specifically for the client’s configuration.

When the ROM image is generated, the code is frozen. So, to benefit from this important optimization, the values of the module configuration parameters must be decided on and “sealed”.

If you attempt to modify a configuration parameter that has been frozen, you will receive a configuration error message. The following example shows an error message that results from an attempt to modify the parameter Semaphore.supportsEvents, which has been frozen in the ROM image.

```
configuring SemaphoreTest.p674 from package/cfg/SemaphoreTest_p674.cfg ...
js: "./SemaphoreTest.cfg", line 13: ti.sysbios.ipc.Semaphore: ‘supportsEvents’ is
sealed
```

Table 2 lists all of the module configuration parameters in the ROM. In general, configuration parameters that are critical to DSP/BIOS performance are frozen in the ROM image, while non-critical parameters are still configurable. Parameters that are frozen are highlighted in **blue**. For modules in the ROM, all proxies have been bound and cannot be changed.

Some of the frozen module configuration parameters have been set to a value other than the default. These are described later in this document, along with an explanation of the value chosen for each.

For more information on a configuration parameter and its default value, refer to the module’s online help reference documentation.

Table 2. Configuration Parameters for Modules in the ROM.

ti.sysbios

BIOS
cpuFreq
taskEnabled
swiEnabled
clockEnabled
rtsGateType
startupFxns[]

ti.sysbios.family.c64p

Cache
EMIFA_BASE
EMIFA_LENGTH
EMIFB_BASE
EMIFB_LENGTH
EMIFC_BASE
EMIFC_LENGTH
EMIFA_CFG
EMIFB_CFG
EMIFC_CFG
initSize
MAR0_31
MAR32_63
MAR64_95
MAR96_127
MAR128_159
MAR160_191
MAR192_223
MAR224_255

Exception
enableExternalMPC
exceptionHook
internalHook
externalHook
nmiHook
returnHook

Hwi
dispatcherAutoNestingSupport
dispatcherSwiSupport
dispatcherTaskSupport
dispatcherIrpTrackingSupport
enableException
swiDisable
swiRestoreHwi
taskDisable
taskRestoreHwi
ierMask
hooks[]

TimerSupport
timer[]

TaskSupport
defaultStackSize
stackAlignment

ti.sysbios.hal

Hwi
dispatcherAutoNestingSupport
dispatcherSwiSupport
dispatcherTaskSupport
dispatcherIrpTrackingSupport
hooks[]

ti.sysbios.ipc

Semaphore
supportsEvents
eventPost
eventSync

ti.sysbios.knl

Clock
tickSource
timerId
swiPriority
tickPeriod

Idle
funcList[]

Swi
taskDisable
taskRestore
numPriorities
hooks[]
numConstructedSwis

Task
initStackFlag
numPriorities
defaultStackSize
defaultStackSection
defaultStackHeap
hooks[]
idleTaskStackSize
idleTaskVitalTaskFlag
numConstructedTasks

ti.sysbios.timers.timer64

Timer
anyMask
numTimerDevices
timerSettings[]
defaultHalf

xdc.runtime

Diags
setMaskEnabled
dictBase

Error
policy
raiseHook
maxDepth

LoggerBuf
enableFlush

Memory
defaultHeapInstance
defaultHeapSize

Startup
maxPasses
firstFxns[]
lastFxns[]
resetFxn

SysMin
bufSize
flushAtExit
outputFxn

System
maxAtexitHandlers
extendFxn

Text
nameUnknown
nameEmpty
nameStatic
isLoaded

Log.Event (all)

Assert.Id (all)

Error.Id (all)

2.3 Non-default Frozen Values

Most frozen module parameters are set to their default values in ROM. The following excerpt shows the frozen configuration parameters that are set to a value other than the default. These frozen parameters are shown in **blue**.

```
var System = xdc.useModule('xdc.runtime.System');
var SysMin = xdc.useModule('xdc.runtime.SysMin');
System.SupportProxy = SysMin;

var Memory = xdc.useModule('xdc.runtime.Memory');
var HeapMem = xdc.useModule('ti.sysbios.heaps.HeapMem');
var defaultHeap = HeapMem.create({size: Memory.defaultHeapSize});
Memory.defaultHeapInstance = defaultHeap;

var BIOS = xdc.useModule('ti.sysbios.BIOS');
BIOS.rtsGateType = BIOS.GateMutex;

var Timer = xdc.useModule('ti.sysbios.timers.timer64.Timer');
var Clock = xdc.useModule('ti.sysbios.knl.Clock');
Timer.anyMask = 0;
Clock.timerId = 0;  // Not frozen

var Diags = xdc.useModule('xdc.runtime.Diags');
Diags.setMaskEnabled = true;

var Text = xdc.useModule('xdc.runtime.Text');
Text.isLoaded = false;  // Not frozen
```

These non-default configuration settings have some implications. The preceding excerpt performs the following actions:

- Freezes the System proxy to 'xdc.runtime.SysMin'.
- Creates a default heap of type ti.sysbios.heaps.HeapMem. This cannot be replaced with another instance, however the size of this heap can still be configured. See Section 2.5, "Modifying Existing Instances".
- Assigns a ti.sysbios.gates.GateMutex instance to the runtime support library. This cannot be changed.
- Freezes Timer.anyMask to 0. This disables the "any timer" feature, and forces the user to select the timer explicitly. 'Clock.timerId' has not been frozen, but the default value has been changed from -1 ("any") to 0.
- Enables the diagnostics mask for all modules.
- Changes the default value of Text.isLoaded from true to false. This parameter is not frozen.

2.4 The “common$” Fields

The common$ module configuration parameters are treated specially. For modules in the ROM, these configuration parameters generally cannot be changed. Attempting to change them will not result in a configuration error, but the changes will have no effect.

For modules in the ROM, the following common$ settings have been applied. Any changes shown here to the Defaults module effectively change the default values for the client’s configuration.

```
var Defaults = xdc.useModule('xdc.runtime.Defaults');

Defaults.common$.namedInstance = true;
Defaults.common$.namedModule = true;

Defaults.common$.instanceHeap = Memory.defaultHeapInstance.$orig;

var LoggerBufParams = new LoggerBuf.Params();
LoggerBufParams.exitFlush = true;
Defaults.common$.logger = LoggerBuf.create(LoggerBufParams);

Defaults.common$.diags_ASSERT = Diags.ALWAYS_OFF;
Defaults.common$.diags_USER1 = Diags.RUNTIME_OFF;
Defaults.common$.diags_USER2 = Diags.RUNTIME_OFF;

var GateMutex = xdc.useModule('ti.sysbios.gates.GateMutex');
var HeapMem = xdc.useModule('ti.sysbios.heaps.HeapMem');
HeapMem.common$.gate = GateMutex.create();
HeapMem.common$.fxntab = true;
```

The above configuration settings have implications *for modules in the ROM.* (All settings can be modified for modules not in the ROM.) This excerpt does the following:

- Enables module and instance names for modules in the ROM. This cannot be changed.
- Sets the default logger to a LoggerBuf instance. The logger field can be changed to point to another instance, but only another xdc.runtime.LoggerBuf instance. The parameters of this LoggerBuf instance can still be changed. See Section 2.5, “Modifying Existing Instances”.
- Sets the default instance heap (the heap used for allocating dynamically created instances) to the default heap. This can be changed.
- Sets some items in the Defaults module. In the client configuration, modifying settings for the Defaults module affects only modules not in the ROM.
- Sets diags_ASSERT to ALWAYS_OFF for the modules in the ROM. This setting cannot be changed.
- Sets diags_USER1 and diags_USER2 to RUNTIME_OFF for all ROM modules. These diagnostics can be turned on for ROM modules by setting them to RUNTIME_ON explicitly for each ROM module.
- Sets diagnostics to ALWAYS_OFF or ALWAYS_ON for ROM modules; these cannot be changed. Diagnostics set to RUNTIME_OFF can only be changed to RUNTIME_ON.

- Turns asserts off for all ROM modules; they cannot be turned on. Asserts can still be turned on for RAM modules.
- Turns Entry and Exit trace off for all ROM modules; they cannot be turned on. This is the default value. Entry and Exit trace can still be turned on for RAM modules.
- Configures HeapMem to use a GateMutex to protect its internal state and enables its function table so that it may be used abstractly as an IHeap instance. These settings cannot be changed.

2.5 Modifying Existing Instances

The ROM configuration contains a ti.sysbios.heaps.HeapMem instance and an xdc.runtime.LoggerBuf instance, which have been assigned to the Memory.defaultHeapInstance and Defaults.common$.logger fields, respectively. These instances cannot be deleted, but their properties can be modified through the $orig property of the fields to which they have been set.

The following example shows how to change the size of the default heap to 0x2000 by setting the $orig.size variable.

```
var Memory = xdc.useModule('xdc.runtime.Memory');
Memory.defaultHeapInstance.$orig.size = 0x2000;
```

To change the number of entries in the default logger to 512, set the $orig.numEntries variable.

```
var Defaults = xdc.useModule('xdc.runtime.Defaults');
Defaults.common$.logger.$orig.numEntries = 512;
```

2.6 Flushing LoggerBuf

The configuration parameter LoggerBuf.enableFlush, which flushes all logs at system exit, is frozen to its default value of false.

To work around this, you can use the LoggerBuf "flushAll" API, which you can call to manually flush the logs. To have this API called at system exit, simply add the following line of code to main():

```
System_atexit(LoggerBuf_flushAll);
```

3 Conclusion

Using the DSP/BIOS 6 image in the DA830 ROM frees up RAM space and allows XDCtools to generate optimized code based on the client's configuration. Keep in mind that the values of certain module configuration parameters that are critical to DSP/BIOS performance are frozen and cannot be changed by the user configuration, and that some of these parameters are set to non-default values.

4 References

- *XDCtools Getting Started Guide* (*xdc_install_dir*/docs/XDCtools_Getting_Started_Guide.pdf). Includes steps for installing and validating the installation. Provides a quick introduction to XDCtools using a "hello world" application.

- *XDCtools User's Guide* (*xdc_install_dir*\docs\rtscpedia\XDCtools_User's_Guide\XDCtools_User's_Guide.html)

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 TI 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. Information of third parties may be subject to additional restrictions.

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.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
Low Power Wireless	www.ti.com/lpw	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265