



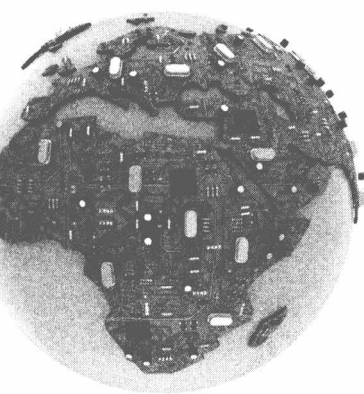
TI C66X多核软件开发 (MCSDK) 技术

基于CCSV5 SYS/BIOS的高级应用 与实例精解

牛金海◎编著



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS



TI C66X多核软件开发 (MCSDK) 技术

基于CCSV5 SYS/BIOS的高级应用 与实例精解

牛金海◎编著



上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

内容提要

本书主要包含 3 部分内容: 第一部分主要介绍 CCS V5 的安装使用以及高级技术实例, 具体包括: 如何输入一个数据文件到 CCS V5, 并使用 Graph 工具显示波形图; 如何实现 C, ASM 以及线性汇编的混合编程; 如何对 YUV 格式的图像/视频数据进行软件仿真处理以及显示等内容。第二部分主要介绍 SYS/BIOS 的高级技术, 具体包括 Hook(钩子)函数的使用、SYS/BIOS 内存管理、SYS/BIOS 设备使用、SYS/BIOS 系统尺寸优化等, 并给出 SWI、Mailbox、TSK 等实例及解析。第三部分主要介绍基于 SYS/BIOS 的多核编程技术(MCSDK), 重点包括 MCSDK 软件包的架构和使用、多核镜像部署(MAD)、自启动(Boot)、HUA, Imaging 等实例。MCSDK 是多核编程非常有用的工具, 在此基础上, 用户可以把主要精力放在应用程序的设计与开发上, 而不是花大量时间精力去研究多核系统底层的配置与任务调度等烦琐的工作。本书适合于高校科研院所高年级的大学生、研究生以及从事 DSP 开发的工程师学习参考。

图书在版编目(CIP)数据

TI C66X 多核软件开发(MCSDK)技术: 基于 CCSV5
SYS/BIOS 的高级应用与实例精解/牛金海编著. —上海:
上海交通大学出版社, 2015
ISBN 978-7-313-13271-0

I. ①T… II. ①牛… III. ①微处理器-并行编译程
序-程序设计 IV. ①TP332②TP314

中国版本图书馆 CIP 数据核字(2015)第 141811 号

TI C66X 多核软件开发(MCSDK)技术

基于 CCSV5 SYS/BIOS 的高级应用与实例精解

编 著: 牛金海

出版发行: 上海交通大学出版社

邮政编码: 200030

出 版 人: 韩建民

印 制: 常熟市梅李印刷有限公司

开 本: 787 mm×1092 mm 1/16

字 数: 607 千字

版 次: 2015 年 7 月第 1 版

书 号: ISBN 978-7-313-13271-0/TP

定 价(含光盘): 88.00 元

地 址: 上海市番禺路 951 号

电 话: 021-64071208

经 销: 全国新华书店

印 张: 26.25

印 次: 2015 年 7 月第 1 次印刷

ISBN 978-7-89424-079-8

版权所有 侵权必究

告读者: 如发现本书有印装质量问题请与印刷厂质量科联系

联系电话: 0512-52661481

前言

本书是《TMS320C66X KeyStone 架构多核 DSP 入门与实例精解》的姊妹篇。两本书内容上相互补充,形成相对完整的知识体系。通过这两本书的阅读,期望读者可以更好地掌握 CCS V5、SYS/BIOS 的使用,以及在 MCSDK 的基础上,尽快地开展基于 C66X 系列多核平台的应用开发。

应读者要求,我们在本书的出版中附带了光盘。光盘中包含了本书涉及的软件代码,部分开放的软件工具,以及我们工作室基于 C6000DSP 的课件 PPT 等内容。本书的部分内容来自我们工作室近几年教学与科研工作,借此机会与各位同行分享。此外,书中的部分实例,也可以在 TI 的相关网站,如维基百科、德州仪器在线技术支持社区(www.deyisupport.com)、TI 的官网(www.ti.com)等网站找到。

如果需要在 TI C66X 系列多核系统上开发项目,学习掌握 MCSDK 软件工具是必须的。在 MCSDK 的基础上,用户可以把自己的精力更加专注在应用程序的开发上,而不是专注在复杂的多核系统本身上。

在本书的编著过程中,为了帮助读者更好地理解其中的概念,书中的部分关键词保留了英文原词以及我们的理解与翻译。

上海交通大学-美国德州仪器联合创新工作室 2013~2014 期间选修高性能数字信号处理的研究生参与了本书的部分实例的整理,在此表示感谢。

虽然,我们努力试图提供可重复的工作,由于参考的软件版本以及软件安装的环境可能会有细微差别,请在理解本书内容的基础上重复书中涉及的实例,不要简单照搬,简单照搬不一定能重复出结果,敬请注意。

随着时间的推移,本书的相关内容可能会与 TI 最新技术不同步,请读者及时了解 TI 官网相关内容的升级与更新。

本书的编写与出版得到德州仪器半导体(上海)有限公司的赞助,特别感谢该公

司大学计划部的沈洁、潘亚涛、黄争、崔萌等的支持。

我们努力列出所有参考资料的源出处,如有遗漏,也请谅解。

在编著本书的过程中,我一直战战兢兢,力争精益求精。唯一的愿望,就是希望能对购买以及阅读本书的读者有所帮助。但是限于我们的水平以及时间,书中存在的疏忽、甚至错误,恳请读者批评指正。

作 者

2015 年 5 月于上海

第 1 篇 CCS V5 使用以及高级应用实例精解

第 1 章	CCS V6 的新特性与安装使用	3
1.1	CCS V6 的新特性	3
1.2	CCS V6 的下载与安装	4
1.3	基于 CCS V6 使用 C2000 LaunchPad 硬件仿真	7
1.3.1	准备工作	8
1.3.2	新建工程	8
第 2 章	如何从外部输入数据文件处理并用 Graph 工具显示波形	17
2.1	*.dat 文件的格式介绍	17
2.2	基于 CCS V5 的例程演示——读入 sine.dat 并显示波形	20
2.3	实验中所遇到的问题	31
2.4	实例程序编程语句的解释说明	32
第 3 章	汇编、线性汇编、C 语言混合编程以及基于 Profile 的性能统计	34
3.1	建立混合编程实例工程	35
3.2	编写程序	41
3.3	运行结果以及性能统计	45
3.3.1	用 Profile 进行混合编程性能测试	46
3.3.2	使用 C 语言库 clock 函数	47
3.3.3	比较分析 profile 和 clock() 函数得到的结果	49
3.4	主函数调用汇编子函数参数传递示例	50
3.5	汇编代码优化	52
3.6	遇到的问题	53

第 4 章 利用 CCS Image Analyzer 工具对 YUV 格式图像处理精解	57
4.1 YUV 格式详解	57
4.1.1 概述	57
4.1.2 与 RGB 格式的转换	57
4.1.3 YUV 格式	57
4.1.4 YUV 存储方式	58
4.1.5 常见的 YUV 格式举例	59
4.2 Image Analyzer 介绍	60
4.2.1 概述	60
4.2.2 property 参数介绍	61
4.3 图像处理示例	63
4.3.1 新建工程	63
4.3.2 代码要点	63
4.3.3 图像处理	64
4.4 视频处理示例	70
4.4.1 新建工程	70
4.4.2 转换格式	70
4.4.3 代码要点	70
4.4.4 视频处理	72
4.5 附录	74
4.5.1 图像处理的 Main.c 代码	74
4.5.2 视频处理的 Main.c 代码	76
4.5.3 C6416 地址空间	78

第 2 篇 SYS/BIOS 高级技术以及实例精解

第 5 章 SYS/BIOS 以及线程调度实例	81
5.1 SYS/BIOS 的下载及安装	82
5.2 在 CCS5.4 中建立基于 SYS/BIOS 的 helloworld 例程	86
5.3 SYS/BIOS 下 Idle 线程的使用实例	91
5.4 SYS/BIOS 中 SWI 实例精解	93
5.4.1 带 SWI 工程的建立过程	93
5.4.2 程序代码及解析	99
5.5 SYS/BIOS 中 TASK 实例精解	105

5.5.1	SYS/BIOS 中相同优先级间的任务切换	105
5.5.2	代码解释	106
5.5.3	例程解析	111
第 6 章	SYS/BIOS 中的同步模块	113
6.1	信号量	113
6.1.1	应用信号量的实例	114
6.1.2	上述实例的输出结果	118
6.2	事件模块(Events)	119
	隐性触发事件	122
6.3	“门”	124
6.3.1	基于屏蔽优先级的“门”应用	125
6.3.2	基于信号量的“门”应用	126
6.3.3	优先级反转	126
6.4	邮箱	127
6.5	队列	128
6.5.1	队列的“FIFO(先入先出)”基本操作	128
6.5.2	队列循环	130
6.5.3	队列中对象的插入与删除	130
6.5.4	队列的原子操作	130
6.6	SYS/BIOS 中 Semaphore 实例精解	131
6.6.1	建立工程	131
6.6.2	代码	131
6.6.3	例程解析	135
6.6.4	Semaphore 中的 API 函数	135
6.7	SYS/BIOS 在 CCS V5.4 中的 MBX 例程	138
6.7.1	程序代码详解	138
6.7.2	MBX, TASK, IDLE 在 cfg 文件中配置	140
6.7.3	DSP/BIOS 中例程的结果	143
第 7 章	SYS/BIOS 中的钩子函数(hook)及其应用	145
7.1	Windows 操作系统中的钩子函数	145
7.2	SYS/BIOS 中的 hooks(钩子)	146
7.3	硬件中断钩子	148
7.3.1	Register 函数	148

7.3.2	Create 和 Delete 函数	148
7.3.3	Begin 和 End 函数	149
7.3.4	硬件中断钩子实例	149
7.4	软件中断钩子	154
7.4.1	Register 函数	155
7.4.2	Create 和 Delete 函数	155
7.4.3	Ready, Begin 和 End 函数	155
7.4.4	软件中断钩子实例	156
7.5	任务钩子	161
7.5.1	Register 函数	162
7.5.2	Create 和 Delete 函数	162
7.5.3	Switch 函数	163
7.5.4	Ready 函数	163
7.5.5	Exit 函数	163
7.5.6	任务钩子实例	163
第 8 章	SYS/BIOS 中的定时服务	170
8.1	定时服务概述	170
8.2	时钟	170
8.3	计时器模块	173
8.4	时间戳模块	173
第 9 章	SYS/BIOS 中的内存管理	175
9.1	背景	175
9.2	内存映射	176
9.2.1	选择有效的平台	176
9.2.2	创建一个自定义的平台	177
9.3	将段放置到内存块中	181
9.3.1	配置简单的段存放	181
9.3.2	使用 SectionSpec 配置段存放	181
9.3.3	提供附加的编译器命令文件	182
9.3.4	默认链接器命令文件和自定义选项	183
9.4	MSP430, Stellaris M3 和 C28 的段和内存映射	184
9.5	堆栈	184
9.5.1	系统堆栈	184

9.5.2	任务堆栈	185
9.5.3	系统堆栈和任务堆栈的 ROV	185
9.6	缓存配置	186
9.6.1	在启动时配置缓存大小寄存器	186
9.6.2	配置参数以便设置 MAR 寄存器	187
9.6.3	缓存执行的程序接口	187
9.7	动态内存分配	187
9.7.1	内存规则	187
9.7.2	设定默认的系统堆	188
9.7.3	使用 xdc.runtime.Memory 模块	189
9.7.4	为模块动态实例指定一个堆	190
9.7.5	使用 malloc() 和 free()	190
9.8	堆实现	190
9.8.1	HeapMem	191
9.8.2	HeapBuf	192
9.8.3	HeapMultiBuf	193
9.8.4	堆跟踪(HeapTrack)	195
第 10 章	SYS/BIOS 中的硬件抽象层(HAL)	197
10.1	硬件抽象层(HAL)应用程序接口(APIs)	197
10.2	Hwi 模块	198
10.2.1	C 函数与系统中断源的关联	198
10.2.2	Hwi 实例配置参数	198
10.2.3	使用非缺省的实例配置参数创建一个 Hwi 对象	199
10.2.4	使能和关闭中断	200
10.2.5	Hwi 应用的一个简单示例	200
10.2.6	中断调度器	203
10.2.7	中断调度器保存和释放寄存器	203
10.2.8	附加的指定目标/设备的 Hwi 模块功能	203
10.3	定时器模块	205
10.4	高速缓存模块	210
10.5	HAL 封装组织	211
第 11 章	SYS/BIOS 中的设备	214
11.1	设备综述	214

- 11.2 负载(Load)模块 214
 - 11.2.1 Load 模块配置 215
 - 11.2.2 获取 Load 统计数据 215
- 11.3 错误处理 216
- 11.4 代码调试器(CCS)中的设备工具 217
- 11.5 RTA Agent 218
 - 11.5.1 从暂停的目标板获取日志(Log)数据 219
 - 11.5.2 系统日志自动配置 219
- 11.6 CCS 中的 RTA Tools 220
 - 11.6.1 原始日志数据 220
 - 11.6.2 打印日志 222
 - 11.6.3 执行图 223
 - 11.6.4 执行图形数据 225
 - 11.6.5 CPU 负载 225
 - 11.6.6 线程负载(Thread Load) 226
 - 11.6.7 CPU 负载数据 226
 - 11.6.8 RTA 控制面板 227
- 11.7 性能优化 229
 - 11.7.1 配置日志 229
 - 11.7.2 诊断配置 230
 - 11.7.3 选择一个堆管理 231
 - 11.7.4 硬中断配置 231
 - 11.7.5 栈校验 231
- 第 12 章 SYS/BIOS 中的输入输出 (Input/Output) 232
 - 12.1 综述 232
 - 12.2 配置设备列表中的驱动 233
 - 12.3 使用 GIO APIs 236
 - 12.3.1 使用 GIO APIs 的限制 236
 - 12.3.2 创建与删除 GIO 通道 237
 - 12.3.3 使用 GIO_read()与 GIO_write()——标准模式 239
 - 12.3.4 使用 GIO_issue(),GIO_reclaim()以及 GIO_prime()——发出/回收模式
..... 241
 - 12.3.5 GIO_abort()以及错误处理 243
 - 12.4 在多种线程环境中使用 GIO 244

12.4.1	在任务中使用 GIO	244
12.4.2	在软中断中使用 GIO	244
12.4.3	在事件中使用 GIO	245
12.5	GIO 与同步机制	245
第 13 章	重新构建 SYS/BIOS	247
13.1	概述	247
13.2	前提条件	247
13.3	使用 bios.mak Makefile 构建 SYS/BIOS	247
13.4	使用 SYS/BIOS 重建工具构建用户的项目	250
第 14 章	SYS/BIOS 的定时基准	252
14.1	定时基准	252
14.2	中断等待	252
14.3	硬件中断基准	252
14.4	软件中断基准	253
14.5	任务基准	254
14.6	信号量基准	256
第 15 章	SYS/BIOS 的尺寸基准	258
15.1	概述	258
15.2	与 DSP/BIOS 5 的比较	258
15.3	默认配置尺寸	259
15.4	静态模块应用尺寸	260
15.4.1	Hwi 应用	260
15.4.2	时钟应用	260
15.4.3	时钟对象应用	261
15.4.4	Swi 应用	261
15.4.5	Swi 对象应用	261
15.4.6	Task 应用	261
15.4.7	Task 对象应用	262
15.4.8	Semaphore(旗语或者信号量)应用	262
15.4.9	旗语对象应用	262
15.4.10	内存应用	262
15.5	动态模块应用尺寸	263

- 15.5.1 动态 Task 应用 263
 - 15.5.2 动态旗语应用 264
- 15.6 时钟应用尺寸 264
- 第 16 章 最小化应用程序占用空间 265**
 - 16.1 概述 265
 - 16.2 减少数据大小 265
 - 16.2.1 移除动态内存分配堆(memory allocation) 265
 - 16.2.2 减少主函数参数空间 266
 - 16.2.3 减少栈大小 266
 - 16.2.4 禁用指定模块(Disabling Named Modules) 266
 - 16.2.5 使得文本字符脱离目标板 266
 - 16.2.6 禁用功能模块表 266
 - 16.2.7 减少退出程序操作的数量 267
 - 16.3 减少代码大小 267
 - 16.3.1 使用自定义构建的 SYS/BIOS 动态库 267
 - 16.3.2 禁用日志 267
 - 16.3.3 设置内存策略 267
 - 16.3.4 禁用内核特征 268
 - 16.3.5 消除打印函数 printf()..... 268
 - 16.3.6 禁用 RTS 线程保护 268
 - 16.3.7 禁止任务堆栈溢出检查 268
 - 16.4 基本大小基准配置脚本 269
- 第 17 章 SYS/BIOS 中的 IOM(I/O Mini-driver)微型驱动接口 271**
 - 17.1 微型驱动接口概述 271
 - 17.2 微型驱动接口函数汇总 273
 - 17.3 微型驱动接口函数定义 275

**第 3 篇 基于 SYS/BIOS 的多核软件编程技术
(MCSDK)以及实例精解**

- 第 18 章 基于 SYS/BIOS 的多核软件开发(MCSDK)技术概述 281**
 - 18.1 引言 281

18.1.1	缩写词和定义	281
18.1.2	支持的设备/平台	283
18.1.3	其他资源	283
18.1.4	MCSDK 信息	283
18.1.5	入门指南	284
18.1.6	API 和 LLD 使用指南	284
18.1.7	硬件-EVM 概述	284
18.1.8	硬件-处理器概述	285
18.1.9	相关软件	285
第 19 章	多核软件开发(MCSDK)软件架构	286
19.1	软件概述	286
19.1.1	平台开发套件	287
19.1.2	操作系统适配层	287
19.1.3	资源管理器	287
19.1.4	芯片支持库	288
19.1.5	底层驱动	289
19.1.6	平台库	295
19.1.7	传输系统	296
19.1.8	EDMA3 底层驱动程序	296
19.1.9	SYS/BIOS RTOS	297
19.2	处理器间通信	298
19.2.1	IPC 传输	299
19.2.2	IPC 流程	301
19.3	网络开发套件	305
19.4	算法库	308
19.4.1	DSP 库	308
19.4.2	图像处理库	309
19.4.3	浮点数学函数库	309
第 20 章	MCSDK 实例演示软件	311
20.1	高性能 DSP 实用应用程序	311
20.2	图像处理示例	311
20.3	引导装载程序和引导工具	311
20.4	工具	312

20.4.1	多核系统分析仪	312
20.4.2	Eclipse RTSC Tools(XDC)	313
20.5	第三方软件和工具	314
20.5.1	CriticalBlue 公司的棱镜	314
20.5.2	PolyCoreSoftware 公司的 Poly-Platform	314
20.6	创建和实例指南	315
20.6.1	创建构建环境	315
20.6.2	构建软件	315
20.6.3	修改库	315
20.6.4	平台库(platform libraries)	315
20.6.5	构建 CSL 和底层设备驱动程序	317
20.6.6	构建设备驱动程序示例工程	317
20.6.7	编辑大端 MCSDK 样板和示例	318
20.6.8	使用软件模拟器构建和运行 NDK client 示例	322
20.6.9	构建 NDK	325
20.6.10	示例	326
20.7	使用 MAD 完成一个多核的可启动的镜像	339
第 21 章	使用 IPC 明确的编程模型	341
21.1	使用和配置导航/QMSS 传输	343
21.1.1	通过使用 QMSS 传输模块 & 其他 TransportQmssSetup 参数改变 GEM 中断	343
21.1.2	TransportQmss 配置选项(TransportQmss Configuration Options)	343
21.1.3	TransportQmss 队列分配的注意事项	345
21.2	使用和配置 sRIO 传输	345
21.2.1	使用 RapidIO 传输配置 IPC	345
21.2.2	修改 sRIO 传输模块 & 其他 TransportSrioSetup 使用的 GEM 中断的 参数	346
21.2.3	TransportSrio 配置选项	346
21.2.4	TransportSrio 内核镜像配置和 IPC 群集参数	348
21.2.5	TransportSrio 单设备内核映射和 IPC 群集配置	348
21.2.6	TransportSrio 多设备内核映射 MAP 和 IPC 群集配置	350
21.2.7	TransportSrio 队列分配的注意事项	354
21.2.8	TransportSrio 应用程序配置要求	354

第 22 章 Booting 和 Flash	356
22.1 Boot 概述	356
22.2 上电自检	356
22.3 初始引导加载程序 (IBL) 和实例	358
22.4 NAND 启动	359
22.5 NOR 启动	359
22.6 TFTP 启动	360
22.7 Flash 和 flash 工具	361
第 23 章 MCSDK 技术支持和产品更新	363
23.1 技术支持和论坛	363
23.2 产品更新	363
23.3 Eclipse 更新管理器	363
23.3.1 Eclipse 更新 (自动)	364
23.3.2 Eclipse 更新 (手动)	364
23.4 MCSDK 常见问题	366
23.4.1 怎样才能恢复 EVM 的出厂默认状态	366
23.4.2 刚更新了 BIOS MCSDK 软件, 该如何加载到 EVM 中	366
23.4.3 在 CCS 环境汇总中使用 JTAG	369
23.4.4 访问发行版中提供的文件的简单方法	370
23.4.5 卸载 BIOS - MCSDK	370
23.4.6 各种外设的示例代码	370
23.4.7 加速下载 BIOS MCSDK 安装文件	371
23.4.8 在 CCS 5.1 下使用 BIOS MCSDK 2.0	371
23.4.9 同一 CCS 的实例中连接并使用类型相同的两个硬件仿真器	371
第 24 章 空 RTSC 工程解决方案	377
24.1 概述	377
24.2 问题	377
24.3 解决方案	378
24.4 警告	378
第 25 章 HUA 以及 Imaging 实例	382
25.1 MCSDK HUA 指南	382
25.1.1 概述	382

- 25.1.2 要求 383
- 25.1.3 软件设计 384
- 25.1.4 构建说明 384
- 25.1.5 运行说明 385
- 25.1.6 排除故障 386
- 25.2 MCSDK 图像处理例程手册 386
 - 25.2.1 概述 386
 - 25.2.2 要求 387
 - 25.2.3 软件设计 387
 - 25.2.4 了解更多处理算法 387
 - 25.2.5 多核框架 388
 - 25.2.6 算法的性能分析 388
 - 25.2.7 用户界面 388
 - 25.2.8 运行说明 389
 - 25.2.9 建立说明 390
 - 25.2.10 软件路径结构概述 391
 - 25.2.11 图像处理示例串行代码概述 391
 - 25.2.12 多核系统分析器的集成和使用 392
 - 25.2.13 用 Prism 分析图像处理示例 392
 - 25.2.14 使用 MAD 功能文件链接和创建可引导的应用程序镜像 393
 - 25.2.15 使用 IBL 引导应用程序镜像 394
- 25.3 MCSA 和 MCSDK 例程 398

参考文献 401