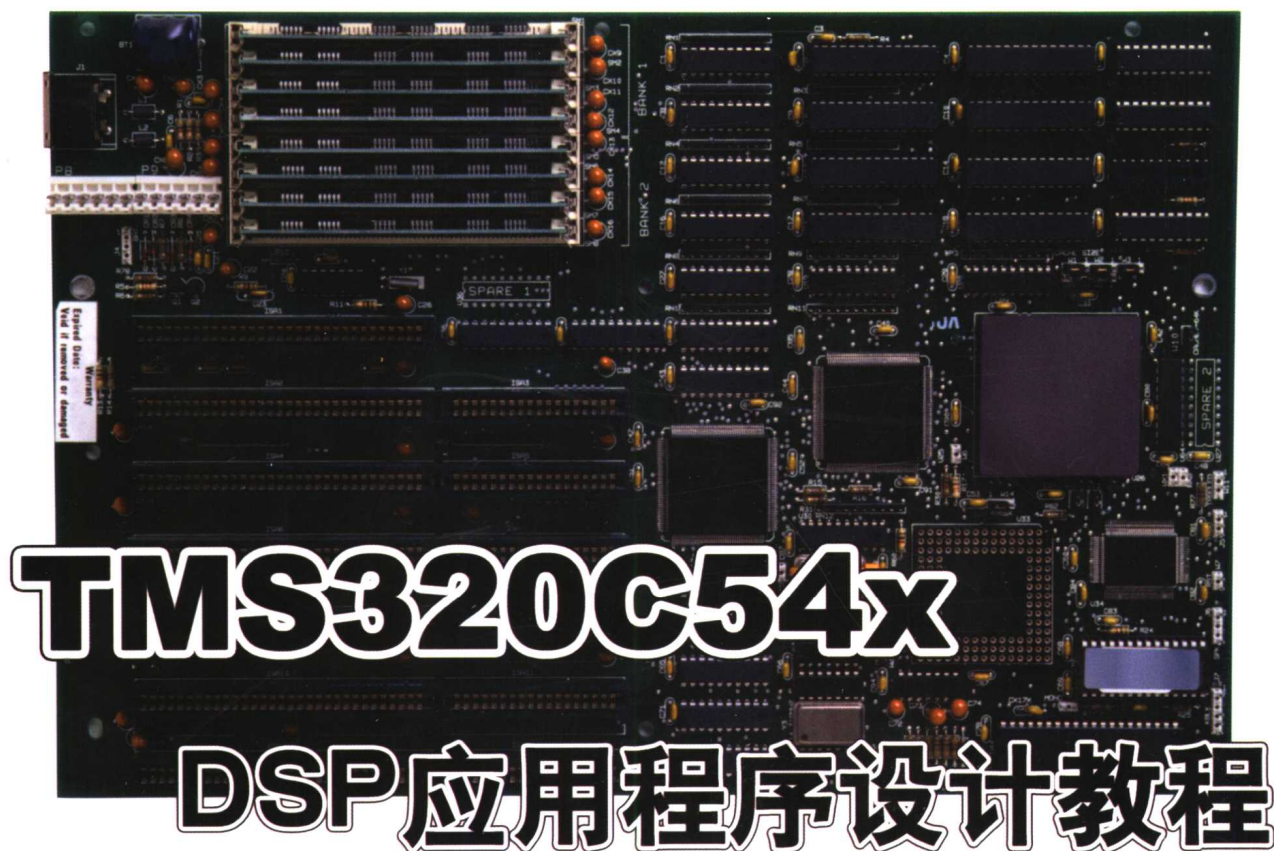


DSP应用开发教程系列



TMS320C54x

DSP应用程序设计教程

清源科技 编著

DSP 应用开发教程系列

TMS320C54x DSP 应用程序设计教程

清源科技 编著



机械工业出版社

本书主要讲述了 TMS320C54x DSP 硬件概况、内部资源、汇编语言寻址方式和指令系统、汇编程序的编写方法、C/C++编译器、C/C++代码的优化和链接、C/C++运行和支持库、C54x DSP 的 C/C++语言、CCS 集成调试环境以及程序设计实例。本书还针对信号处理的应用,讲述如何在 C54x DSP 平台上实现相应应用程序的开发。

本书主要面向从事自动控制、信号处理、图像处理、语言处理、通信以及相关电子仪器仪表系统设计的技术人员,也适合高校师生学习参考,是一本全面而实用的针对 TMS320C54x 系列 DSP 的学习教程。

图书在版编目(CIP)数据

TMS320C54x DSP 应用程序设计教程/清源科技编著. —北京:机械工业出版社, 2004.1

(DSP 应用开发教程系列)

ISBN 7-111-13500-8

I. T… II. 清… III. 数字信号—信号处理—数字通信系统, TMS320C54x DSP—程序设计—教材 IV. TN914.3

中国版本图书馆 CIP 数据核字(2003)第 108388 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

责任编辑:吉玲 封面设计:张静

责任印制:闫焱

北京交通印务实业公司印刷·新华书店北京发行所发行

2004 年 1 月第 1 版第 1 次印刷

787mm×1092mm 1/16·21 印张·521 千字

0 001—4 000 册

定价:33.00 元

编辑信箱: jiling@mail.machineinfo.gov.cn

凡购本书,如有缺页、倒页、脱页,由本社发行部调换

本社购书热线电话(010) 68993821、88379646

封面无防伪标均为盗版

前言

目前, DSP (Digital Signal Processor) 已经广泛应用于高速自动控制、图像处理、通信技术、无线电、语音处理、网络设备、医疗设备、仪器仪表和家电等领域; DSP 为数字信号处理提供了高效而可靠的硬件基础。目前, 应用最广泛的 DSP 是 TI (德州仪器) 公司的产品。TI 公司 DSP 的主流产品包括 TMS320C2000 系列(包括 TMS320C2x/C2xx)、TMS320C5000 系列(包括 TMS320C5x/C54x/C55x)、TMS320C6000 系列(包括 TMS320C62x/C67x), 本书主要介绍 TMS320C54x 系列 DSP。

TMS320C54x 系列 DSP 是适合于语音处理、图像处理、医疗仪器和通信技术等领域的一种 DSP。TMS320C54x 系列 DSP 具有先进的修正哈佛结构(1 条程序总线、3 条数据总线和 4 条地址总线)、专门硬件逻辑的 CPU、片内存储器、片内外设和专用的指令集合将 TMS320C54x 的 CPU 和片内存储器与外设配置组合在一起的螺旋结构, 使得它可以满足众多领域的应用要求。TI 公司还为 TMS320C54x DSP 的应用程序开发提供良好的继承环境和工具 CCS(Code Composer Studio), 目前最新的版本为 CCS2.0。用户可以使用 TI 公司提供的编译和链接工具, 使用汇编语言和 C/C++ 语言进行应用程序开发。

本书主要讲述 TMS320C54x DSP 的应用程序设计的相关内容。具体包括 TMS320C54x DSP 硬件概况、内部资源、汇编语言寻址方式和指令系统、汇编程序的编写方法、C/C++ 编译器、C/C++ 代码的优化和链接、C/C++ 运行时支持库、TMS320C54x DSP 的 C/C++ 语言、CCS 集成调试环境以及程序设计实例。全书共分 13 章, 第 1~4 章主要讲述 DSP 的结构原理、存储器、程序控制和寻址方式; 第 5、6 章主要讲述汇编语言和汇编环境的编译器和链接器的有关知识; 第 7~11 章主要讲述了 TMS320C54x 的 C/C++ 编译器、C/C++ 代码的优化、C/C++ 程序的链接、汇编和 C/C++ 混合编程、运行时支持库以及与 TMS320C54x 相关的 C/C++ 语言知识; 第 12 章主要讲述了 TMS320C54x DSP 的集成开发工具 CCS2.0; 第 13 章详细讲述了如何进行应用程序开发的编程技巧和实例, 并以信号处理的几个经典滤波器为例讲述了如何开发应用程序的过程。全书紧扣实例进行讲解, 理论和实例环环相扣, 适于读者学习参考。

本书非常适合使用 TMS320C54x 系列 DSP 的技术人员参考, 也适合高等学校师生学习参考, 是一本全面而实用的 TMS320C54x 系列 DSP 应用程序设计的学习教程。

本书由清源科技公司策划支持, 具体由李健博士、张林启博士和胡波博士编写, 江思敏、姚鹏翼、胡荣、陈明、孙鲲鹏、李晓峰、李伟、张志翔、孙旭东等参与了部分章节的编写和资料的准备工作, 吴明君、张华、戴向国、李兵、严维东、廖克明、黄杜和徐斌等为本书的编写提供了详细的技术支持, 江思敏对全书进行统稿。由于水平有限, 时间仓促, 书中缺点和不足在所难免, 敬请广大读者批评指正。作者 Email: jiangsimin@yahoo.com。

编 者
于清华园

目 录

前 言

第 1 章 TMS320C54x 系列 DSP 概述	1	2.1 TMS320C54x DSP 存储器	
1.1 DSP 芯片概述	1	概述	25
1.2 TMS320C54x 系列 DSP 的		2.2 存储器空间	25
结构和特点	2	2.3 程序存储器	33
1.2.1 TMS320C54x 系列 DSP 的概述	2	2.3.1 程序存储器的可配置性	33
1.2.2 TMS320C54x 系列 DSP		2.3.2 片内 ROM 的组织	33
的特点	3	2.3.3 程序存储器地址映射	
1.2.3 TMS320C54x DSP 的内部		和片内 ROM 内容	34
硬件结构	5	2.3.4 片内 ROM 的代码内容	
1.3 总线结构	7	和映射	34
1.4 CPU 状态和控制寄存器	8	2.3.5 扩展程序存储器	35
1.4.1 状态寄存器 ST0 和 ST1	8	2.4 数据存储器	37
1.4.2 处理器工作模式状态 (PMST)		2.4.1 数据存储器的可配置性	37
寄存器	11	2.4.2 片内 RAM 的组织	38
1.5 算术逻辑单元 (ALU)	13	2.4.3 存储器映射寄存器	39
1.5.1 ALU 的输入	13	2.5 I/O 空间及其访问	40
1.5.2 溢出处理	15	第 3 章 寻址方式和程序控制	43
1.5.3 进位位	15	3.1 立即寻址	43
1.5.4 双 16 位模式	15	3.2 绝对寻址	45
1.6 累加器 A 和 B	15	3.2.1 数据存储器地址 (dmda)	
1.6.1 保存累加器的内容	16	寻址	45
1.6.2 累加器移位和循环操作	16	3.2.2 程序存储器地址 (pmda)	
1.6.3 饱和和处理累加器内容	17	寻址	45
1.6.4 专用指令	17	3.2.3 端口地址 (PA) 寻址	45
1.7 桶式移位器	18	3.2.4 *(lk)寻址	46
1.8 乘法器/加法器单元	19	3.3 累加器寻址	46
1.8.1 乘法器的输入源	19	3.4 直接寻址	46
1.8.2 乘法/加法(MAC)指令	20	3.4.1 基于 DP 的直接寻址	48
1.8.3 MAC 和 MAS 乘法运算的饱和		3.4.2 基于 SP 的直接寻址	49
处理	21	3.5 间接寻址	49
1.9 比较、选择和存储单元	21	3.5.1 单操作数寻址	49
1.10 指数编码器	23	3.5.2 ARAU 和地址产生操作	50
第 2 章 存储器和 I/O 空间操作	25	3.5.3 单操作数地址的修改	50

3.5.4 双操作数寻址	55	4.1.3 流水线中的返回指令	94
3.5.5 修正(ARP)模式	58	4.1.4 流水线中的条件执行	96
3.6 存储器映射寄存器寻址	59	4.1.5 流水线中的条件调用和 分支转移	97
3.7 堆栈寻址	60	4.2 中断和流水线	98
3.8 数据类型	61	4.3 双访问存储器和流水线	99
3.9 产生程序存储器地址	62	4.3.1 解决取指和读操作数之间 的冲突	101
3.10 程序计数器(PC)	63	4.3.2 解决写操作数和读双操作数 之间的冲突	101
3.11 程序控制	64	4.3.3 解决写操作数、写操作数 和读双操作数的冲突	102
3.11.1 分支转移	64	4.4 单访问存储器和流水线	103
3.11.2 调用	65	4.5 流水线等待	104
3.11.3 返回	67	4.5.1 访问存储器映射寄存器的 推荐指令	104
3.11.4 条件操作	68	4.5.2 更新 ARx、BK 或 SP—解决冲突	105
3.11.5 重复单条指令	70	第 5 章 汇编程序设计	107
3.11.6 重复一个指令块	72	5.1 应用软件开发流程和工具	107
3.12 复位操作	73	5.2 COFF 文件概述	108
3.13 TMS320C54x DSP 的 中断系统	73	5.2.1 COFF 文件的 基本单元——段	109
3.13.1 TMS320C54x 的中断系统 概述	73	5.2.2 汇编器对段的处理	109
3.13.2 中断标志寄存器(IFR) ..	74	5.2.3 链接器对段的处理	114
3.13.3 中断屏蔽寄存器(IMR) ..	76	5.2.4 重新定位	115
3.13.4 接收中断请求	77	5.2.5 运行时重新定位	116
3.13.5 应答中断	78	5.2.6 加载程序	117
3.13.6 执行中断服务程序(ISR) ..	78	5.2.7 COFF 文件中的符号	117
3.13.7 保存中断上下文	79	5.3 汇编语言程序格式	118
3.13.8 中断等待时间	79	5.3.1 汇编程序格式	118
3.13.9 中断操作流程	79	5.3.2 汇编语言的数据格式	120
3.13.10 重新映射中断向量地址 ..	81	5.3.3 表达式	125
3.13.11 中断和中断向量表	82	5.3.4 内置函数	128
3.14 节电模式	86	5.3.5 加载值到扩展程序存储器 ..	129
3.14.1 IDLE1 模式	87	5.4 汇编伪指令	129
3.14.2 IDLE2 模式	87	5.4.1 定义段的伪指令	129
3.14.3 IDLE3 模式	87	5.4.2 初始化常数的伪指令	130
3.14.4 保持(Hold)模式	88	5.4.3 对准段程序计数器的伪指令 ..	133
3.14.5 其他节电性能	88		
第 4 章 流水线	89		
4.1 流水线操作	89		
4.1.1 流水线的分支转移指令	91		
4.1.2 流水线中的调用指令	93		

5.4.4	格式化输出清单文件的 伪指令	134	7.3.1	常用的选项	182
5.4.5	引用其他文件的伪指令	135	7.3.2	指定文件名	183
5.4.6	条件汇编伪指令	136	7.3.3	修改编译器对文件名的 解释	183
5.4.7	汇编时符号伪指令	136	7.3.4	修改编译器对文件名扩展的 解释和命名	184
5.4.8	混合伪指令	138	7.3.5	指定目录	184
5.4.9	常用伪指令及其语法格式 ..	139	7.3.6	控制汇编器的选项	185
5.5	程序汇编	139	7.4	使用环境变量改变编译器 特性	185
5.5.1	运行汇编程序	140	7.4.1	指定文件目录(C_DIR 和 C54X_C_DIR)	185
5.5.2	条件汇编	142	7.4.2	设置默认编译器选项(C_OPTION 或 C54X_C_OPTION)	186
5.5.3	列表文件	143	7.5	控制预处理器	187
5.5.4	交叉引用清单	146	7.5.1	预定义的宏名	187
5.6	程序链接	147	7.5.2	#include 文件的搜索路径 ..	187
5.6.1	链接器的调用	147	7.5.3	用-i 选项改变#include 文件 的搜索路径	188
5.6.2	链接器的选项	148	7.5.4	产生预处理的列表 文件(-ppo 选项)	188
5.6.3	链接器命令 文件(.cmd 文件)	149	7.5.5	预处理后继续 编译(-ppa 选项)	189
5.7	编写链接器命令文件 (.cmd 文件)	151	7.5.6	产生一个带注释的预处理列 表文件(-ppc 选项)	189
5.7.1	MEMORY 指令	151	7.5.7	产生一个具有行控制信息的 预处理列表文件(-ppi 选项) ..	189
5.7.2	SECTIONS 指令	153	7.5.8	产生由#include 伪指令包含 的文件列表(-ppi 选项)	189
5.7.3	MEMORY 和 SECTIONS 指令的默认算法	157	7.6	使用直接插入函数展开	189
5.7.4	命令文件编写及链接实例 ..	157	7.6.1	直接插入内部操作数	189
5.8	宏定义和调用宏	160	7.6.2	无保护的定义控制直接插入 ..	190
5.8.1	宏定义	160	7.6.3	使用预处理器符号_INLINE 的保护直接插入	190
5.8.2	调用宏	162	7.6.4	直接插入限制	191
5.8.3	使用递归和嵌入宏	162	7.7	使用交互列表工具	192
第 6 章	汇编语言指令	164	7.8	编译错误简介	192
6.1	指令集概述	164	7.8.1	控制诊断消息	194
6.1.1	算术操作	164	7.8.2	如何使用诊断选项	194
6.1.2	逻辑操作	168			
6.1.3	程序控制	169			
6.1.4	加载和存储指令	172			
6.2	指令集的符号和缩写	175			
第 7 章	C/C++编译器概述	180			
7.1	C/C++编译器命令解释程序 ..	180			
7.2	调用编译器解释命令程序 ..	181			
7.3	改变编译器的选项	181			

第 8 章 优化 C/C++ 代码	196	10.3 关键词	219
8.1 使用 C 编译优化器	196	10.3.1 const 关键词	219
8.2 使用 -O3 选项	197	10.3.2 ioport 关键词	219
8.2.1 控制文件级 优化 (-Oln 选项)	197	10.3.3 interrupt 关键词	220
8.2.2 创建优化信息 文件 (-Omn 选项)	197	10.3.4 near 和 far 关键词	221
8.3 执行程序级优化	198	10.3.5 volatile 关键词	221
8.3.1 控制程序级优化 (-Opn 选项)	198	10.4 寄存器变量和全局寄存器 变量	221
8.3.2 混合 C/C++ 语言和汇编的 优化考虑因素	199	10.4.1 寄存器变量	221
8.4 自动直接插入扩展 (-Oi 选项)	200	10.4.2 全局寄存器变量	222
8.5 使用交互列表工具	200	10.5 Pragma 伪指令	223
8.6 访问优化代码中的别名变量	201	10.6 初始化静态和全局变量	228
8.7 调试优化的代码和优化类型	201	10.7 C/C++ 和汇编语言混合编程	229
8.7.1 调试优化的代码	202	10.7.1 用 C/C++ 代码调用汇编语言 模块	229
8.7.2 压缩优化的 代码 (-gp 和 -O 选项)	202	10.7.2 在 C 语言中插入汇编语句	230
8.7.3 可执行的优化类型	202	10.7.3 在程序中访问汇编语言 变量	231
第 9 章 链接 C/C++ 代码	211	10.7.4 访问汇编语言的常数	232
9.1 调用链接器	211	第 11 章 运行时环境和支持库	233
9.1.1 单步调用链接器	211	11.1 存储器模式	233
9.1.2 作为编译的步骤调用 链接器	212	11.1.1 段	233
9.2 禁用链接器 (-c 编译器选项)	212	11.1.2 C 系统堆栈	234
9.3 控制链接过程	212	11.1.3 分配 const 到程序存储器	234
9.3.1 链接运行时支持库	212	11.1.4 动态存储器分配	235
9.3.2 运行时初始化	213	11.1.5 变量初始化	236
9.3.3 全局对象构造器	213	11.1.6 为静态和全局变量分配 存储器	236
9.3.4 指定初始化类型	213	11.1.7 字段/结构体定位	236
9.3.5 指定段在存储器中的分配	214	11.1.8 字符串常数	236
9.3.6 链接器命令文件实例	214	11.2 寄存器规定	237
第 10 章 TMS320C54x C/C++ 语言	217	11.2.1 状态寄存器的位	238
10.1 TMS320C54x C/C++ 语言 的特点	217	11.2.2 寄存器变量	239
10.1.1 TMS320C54x C 语言特点	217	11.3 函数结构和调用规定	239
10.1.2 TMS320C54x C++ 语言 特点	218	11.3.1 函数如何进行调用	239
10.2 数据类型	218	11.3.2 被调用函数如何响应	240
		11.3.3 访问变量和局部变量	241
		11.3.4 分配帧和使用 32 位存储器 读指令	241

11.4 中断处理	241	12.9.5 建立“合成 (Constellation)” 图形.....	277
11.4.1 C 语言中断的基本知识.....	241	第 13 章 应用程序开发实例	279
11.4.2 使用 C/C++ 中断程序	242	13.1 提高系统性能的几种方法 ...	279
11.4.3 保存中断入口点的上下文	242	13.1.1 有效存储器分配的建议 ...	279
11.5 系统初始化	243	13.1.2 存储器定位要求	280
11.5.1 变量自动初始化	243	13.1.3 堆栈初始化	281
11.5.2 全局构造器	243	13.1.4 重叠管理	281
11.5.3 初始化表	244	13.1.5 存储区之间的移动	281
11.5.4 运行时变量的自动初始化	245	13.1.6 有效的功耗管理	283
11.5.5 在加载时变量的初始化 ...	245	13.2 算术运算	283
11.6 运行时支持库函数	246	13.2.1 除法和求模运算	283
11.6.1 库	246	13.2.2 正弦和余弦运算	286
11.6.2 C 的 I/O 函数	247	13.2.3 扩展精度运算	290
11.6.3 头文件	252	13.2.4 浮点运算	295
第 12 章 集成开发环境 (CCS)	258	13.3 快速傅里叶变换 (FFT) 的 DSP 实现	298
12.1 CCS2.0 系统的安装和配置 ¹ ...	258	13.3.1 离散傅里叶变换 (DFT)	298
12.2 建立工程文件	261	13.3.2 快速傅里叶变换 (FFT)	299
12.3 设置工程项目选项	262	13.3.3 快速傅里叶变换的 DSP 实现.....	300
12.3.1 编译器环境参数设置	262	13.4 FIR 滤波器的 DSP 实现	312
12.3.2 链接器环境参数设置	266	13.5 IIR 滤波器的 DSP 实现	317
12.4 编译、链接和运行目标文件	268	13.6 自适应滤波的 DSP 实现	320
12.4.1 编译和链接	268	13.7 CODEC 应用程序的 DSP 实现.....	323
12.4.2 仿真运行输出目标文件 ...	268	参考文献	328
12.5 查看存储器信息	269		
12.6 查看寄存器信息	270		
12.7 修改存储器和寄存器内容 ...	271		
12.8 设置断点和探测点	272		
12.9 图形显示	273		
12.9.1 定义探测点和输入数据 ...	273		
12.9.2 将探测点与数据文件 链接起来	274		
12.9.3 建立幅值—时间图形	275		
12.9.4 建立“眼(Eye)”图形	276		

第 1 章 TMS320C54x 系列 DSP 概述

TMS320C54x 系列低功耗的 16 位定点数字信号处理器 DSP (Digital Signal Processor) 是 TMS320 家族的 DSP 产品系列之一。本章是对当前 TMS320 家族作一个概述, 并介绍 TMS320C54x 系列 DSP 的结构和特点。本书除标题外, 其他地方均使用 C54x DSP 代替 TMS320C54x 系列 DSP。

1.1 DSP 芯片概述

数字信号处理 (Digital Signal Processing) 是一门广泛应用于许多领域的新兴学科。20 世纪 60 年代以来, 随着计算机和信息技术的飞速发展, 数字信号处理技术应运而生并得到迅速的发展。在过去的 20 多年时间里, 数字信号处理已经在通信等领域得到极为广泛的应用。

1. 什么是 DSP 芯片

DSP 芯片, 也称数字信号处理器, 是一种具有特殊结构的微处理器。DSP 芯片的内部采用程序总线 and 数据总线分开的哈佛结构, 具有专门的硬件乘法器, 广泛采用流水线操作, 提供特殊的 DSP 指令, 可以用来快速地实现各种数字信号处理算法。根据数字信号处理的要求, DSP 芯片一般具有如下的主要特点:

- (1) 在一个指令周期内可完成一次乘法和一次加法。
- (2) 程序和数据空间分开, 可以同时访问指令和数据。
- (3) 片内具有快速 RAM, 通常可通过独立的数据总线同时访问两块芯片。
- (4) 具有低开销或无开销循环及跳转的硬件支持。
- (5) 快速的中断处理和硬件 I/O 接口支持。
- (6) 具有在单周期内操作的多个硬件地址产生器。
- (7) 可以并行执行多个操作。
- (8) 支持流水线操作, 使取指、译码、取操作数和执行等操作可以重叠执行。

2. DSP 芯片的基本结构

DSP 芯片的基本结构包括:

(1) 哈佛结构 哈佛结构的主要特点是将程序和数据存储在不同的存储空间中, 即程序存储器和数据存储器是两个相互独立的存储器, 每个存储器独立编址、独立访问。与两个存储器相对应的是系统中设置了程序总线 and 数据总线, 从而使数据的吞吐率提高了一倍。由于程序和数据存储在两个分开的空间中, 因此取指和执行能完全重叠进行。

(2) 流水线操作 流水线与哈佛结构相关。DSP 芯片广泛采用流水线, 以减少指令执行的时间, 从而增强了处理器的处理能力。处理器可以并行处理 2~4 条指令, 每条指令处于流水线的不同阶段。图 1-1 所示为一个四级流水线操作的例子。

(3) 多总线结构 许多 DSP 芯片内部都采用了多总线结构, 这样可以保证在一个机器周期内, 同时访问数据和程序存储空间。因此可以解决传统芯片的总线冲突问题, 使系统的

速度和效率大大提高。

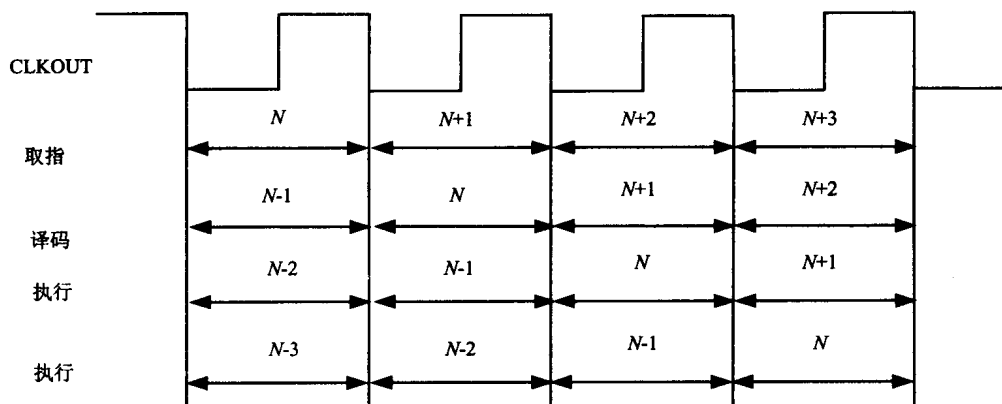


图 1-1 四级流水线操作

(4) 专用的硬件乘法器 乘法速度越快，DSP 的性能越高。由于具有专用的应用乘法器，乘法可在一个指令周期内完成。

(5) 特殊的 DSP 指令 为了更好地满足数字信号处理应用的需要，在 DSP 指令系统中，设计了一些特殊的 DSP 指令，以完成一些专门的运算。例如 C54x DSP 的 FIRS 指令，专门用于 FIR 滤波运算。

(6) 快速的指令周期 哈佛结构、流水线操作、专用的硬件乘法器、特殊的 DSP 指令再加上集成电路的优化设计，可使 DSP 芯片的指令周期在 50ns 以下，现在高档的 DSP 指令周期可以达到 5ns。

1.2 TMS320C54x 系列 DSP 的结构和特点

1.2.1 TMS320C54x 系列 DSP 的概述

C54x 系列 DSP 具有很高的操作灵活性和速度。它具有一个先进的修正哈佛结构（1 条程序总线、3 条数据总线和 4 条地址总线）、专门硬件逻辑的 CPU、片内存储器、片内外设和专用的指令集、将 C54x CPU 和片内存储器与外设配置组合在一起的螺旋结构，使得可以满足电子市场众多领域的应用要求。

C54x 系列 DSP 具有以下优点：

- 增强的哈佛结构。DSP 具有 1 条程序总线、3 条数据总线和 4 条地址总线，使系统的性能大大增强。
- 具有高度平行和带有专门硬件逻辑的先进 CPU 设计。
- 为快速算法而设计的高度专用的指令系统，以及优化的高级语言开发系统。
- 模块化结构设计。
- 高性能和低功耗的先进 IC 工艺技术。
- 新的静电设计结构而获得低功耗和增强的抗辐射能力。

1.2.2 TMS320C54x 系列 DSP 的特点

C54x 是为实现低功耗、高性能而专门设计的定点 DSP 芯片，C54x 的主要特点包括：

(1) CPU

- 先进的多总线结构（1 条程序总线、3 条数据总线和 4 条地址总线）。
- 40 位算术逻辑单元（ALU），包括一个 40 位的桶式移位器和两个独立的 40 位累加器。
- 17 位×17 位并行乘法器和一个 40 位专用的加法器，用于非流水线的单周期乘法/累加（MAC）操作。
- 比较、选择、储存单元（CSSU），用于维特比算子的加法/比较选择。
- 指数编码器 E，用来在一个单周期内计算一个 40 位累加器中数值的指数。
- 两个地址产生器，包括 8 个辅助寄存器和 2 个辅助寄存器算术单元。

注意：C5420 具有双 CPU/核心结构。

(2) 存储器

- 192K 字×16 位可寻址的存储器空间（64K 字的程序空间、64K 字的数据空间和 64K 字的 I/O），对于 C548、C549、C5402、C5410 和 C5420 DSP，可以扩展程序空间为 8MB。
- 片内配置数量见表 1-1。

表 1-1 片内配置数量

器件	程序 ROM	程序/数据 ROM	DARAM ^①	SARAM ^②
'C541	20	8	5	0
'C542	2	0	10	0
'C543	2	0	10	0
'C545	32	16	6	0
'C546	32	16	6	0
'C548	2	0	8	24
'C549	16	16	8	24
'C5402	4	4	16	0
'C5410	16	0	8	56
'C5420	0	0	32	168

① 双访问 RAM。

② 单访问 RAM。

(3) 指令集

- 单指令重复和块重复操作。
- 用于程序和数据管理的块存储器移动指令。
- 32 位长操作数指令。
- 2 或 3 个操作数同时读的指令。
- 具有并行存储和并行加载的算术指令。
- 条件存储指令。
- 从终端快速返回。

(4) 片内外设

- 软件可编程的等待状态发生器。
- 可编程的存储器转换。
- 具有内部振荡器或外部时钟源的片内锁相环 (PLL) 时钟发生器。当使用外部时钟源时, 可以从表 1-2 的选项中选择倍频值, 每种器件只能从表中的某一个选项中选择时钟 PLL 倍频, 而不能同时从两个选项中选择。

表 1-2 使用外部时钟时的可选 PLL 倍频值

选项 1	选项 2	选项 3
1.0	1.0	软件可编程 PLL ^①
1.5	4.0	
2.0	4.5	
3.0	5.0	

① C541B、C545A、C546A、C548、C549、C5402、C5410 和 C5420 DSP 具有软件可编程 PLL 和两个附加的饱和模式。软件可编程 PLL 将在第 2 章中讲述, 饱和模式将在处理器模式状态寄存器 (PMST) 中讲述。

- 外部总线关控制可禁止外部数据总线、地址总线和控制信号。
- 具有总线保持器特性的数据总线。
- 可编程的定时器。
- 端口。C54x DSP 根据器件的信号不同, 具有如表 1-3 所示的配置端口数量。

表 1-3 C54x DSP 的端口配置

器件	主机端口接口	串行端口			
		同步	缓冲	多通道缓冲 (McBSPs)	时分复用 (TDM)
'C541	0	2	0	0	0
'C542	1	0	1	0	1
'C543	0	0	1	0	1
'C545	1	1	1	0	0
'C546	0	1	1	0	0
'C548	1	0	2	0	1
'C549	1	0	2	0	1
'C5402	1	0	0	2	0
'C5410	1	0	0	3	0
'C5420	1	0	0	6	0

(5) 25/20/15/12.5/10/6.25ns 单周期、定点指令执行周期, 具体可参考相关的器件手册。

(6) 电源。

- 可用 IDLE 1、IDLE 2 和 IDLE 3 指令来控制功耗, 以工作在节电方式。
- 控制可以禁止 CLKOUT 信号。

(7) 仿真标准符合 IEEE1149.1 标准。

1.2.3 TMS320C54x DSP 的内部硬件结构

C54x 系列芯片同 TI 公司其他 DSP 定点数字信号处理器一样,为典型的哈佛结构,图 1-2 为其结构图。

C54x DSP 采用先进的修正哈佛结构和 8 条总线,使处理器的性能大大提高。其独立的程序和数据总线提供了高度的并行操作,允许同时访问程序存储器和数据存储器。例如,在单个周期中,可以执行 3 次读操作和 1 次写操作。具有并行存储和专门用处的指令可以利用这种并行结构。

另外,数据可以在数据和程序空间之间进行传送。这种并行操作提供了强大的算术、逻辑和位操作指令集,这些操作可以在单个机器周期中执行。此外,C54x DSP 还包括管理中断、重复操作和函数调用的控制机制。

C54x 的内部硬件结构包括如下功能单元:

(1) 中央处理单元(CPU)。C54x 系列 DSP 的所有芯片的 CPU 都相同,可以进行高速并行算术和逻辑处理。

(2) 内部总线结构。C54x DSP 有 8 条 16 位总线,包括 4 条程序/数据总线和 4 条地址总线,因此,可以在每个指令周期内产生两个数据存储地址,大大提高了并行数据处理速度。

(3) 特殊功能寄存器。C54x DSP 共有 26 个特殊功能寄存器,用于对片内各功能模块进行控制、访问和其他管理。这些寄存器位于一个具有特殊功能的 CPU 映射存储区内。

(4) 数据存储器(RAM)。C54x 片内的数据存储器(RAM)分成两类:一类是每个指令周期内可以进行两次存取操作的双访问 RAM(DARAM);另一类是每个指令周期只能进行一次存取操作的单访问 RAM(SARAM)。不同型号的 C54x DSP 的 DARAM 和 SARAM 的容量和存取速度不同。

(5) 程序存储器(ROM)。C54x 的片内程序存储器有片内 ROM、双访问 RAM(DARAM)、单访问 RAM(SARAM)、双访问和单访问可共享的 RAM(可通过软件配置为程序存储空间)。大部分 C54x DSP 可寻址 64K 字 16 位程序存储空间。

程序存储空间不仅定义在 ROM 上,也可以定义在片上 RAM 中。当需要高速运行的程序时,可以应用自动加载的方法,将程序载入片内 RAM,提高运行效率,降低对外部 ROM 的速度要求。不同的 C54x 器件的 ROM 容量配置不同。

(6) I/O 端口。所有 C54x 只有两个通用 I/O,即 $\overline{\text{BIO}}$ 和 XF。为了访问更多的通用 I/O,可以对主机通信并行口和同步串行口进行配置,以用作通用 I/O。另外,还可以扩展外部 I/O,C54x DSP 可以访问 64K 字的 I/O,外部 I/O 必须使用缓冲或锁存电路,配合外部 I/O 读写控制时序构成外部 I/O 的控制电路。

(7) 主机通信接口(HPI)。HPI 提供与主处理器接口的并行端口。通过 C54x 的片内存储器实现 C54x 器件和主处理器之间的信息交换。不同型号的器件 HPI 配置不同。

(8) 串口。C54x DSP 的串口随器件的不同而不同,C54x DSP 的串口分为 4 种:同步串口、缓冲同步串口(BSP)、多通道缓冲(McBSP)的串口和时分复用(TDM)串口。

(9) 定时器。C54x 器件具有一个带 4 位预定标器的 16 位定时电路。定时器可以由专门的状态位编程实现停止、重启动、复位和禁止。定时器/计数器每次减少到 0,则产生一个定时中断。在每个 CLKOUT 周期,定时器/计数器减少 1。

(10) 中断系统。C54x 的中断可以由硬件驱动（硬件中断）或软件驱动（软件中断）。

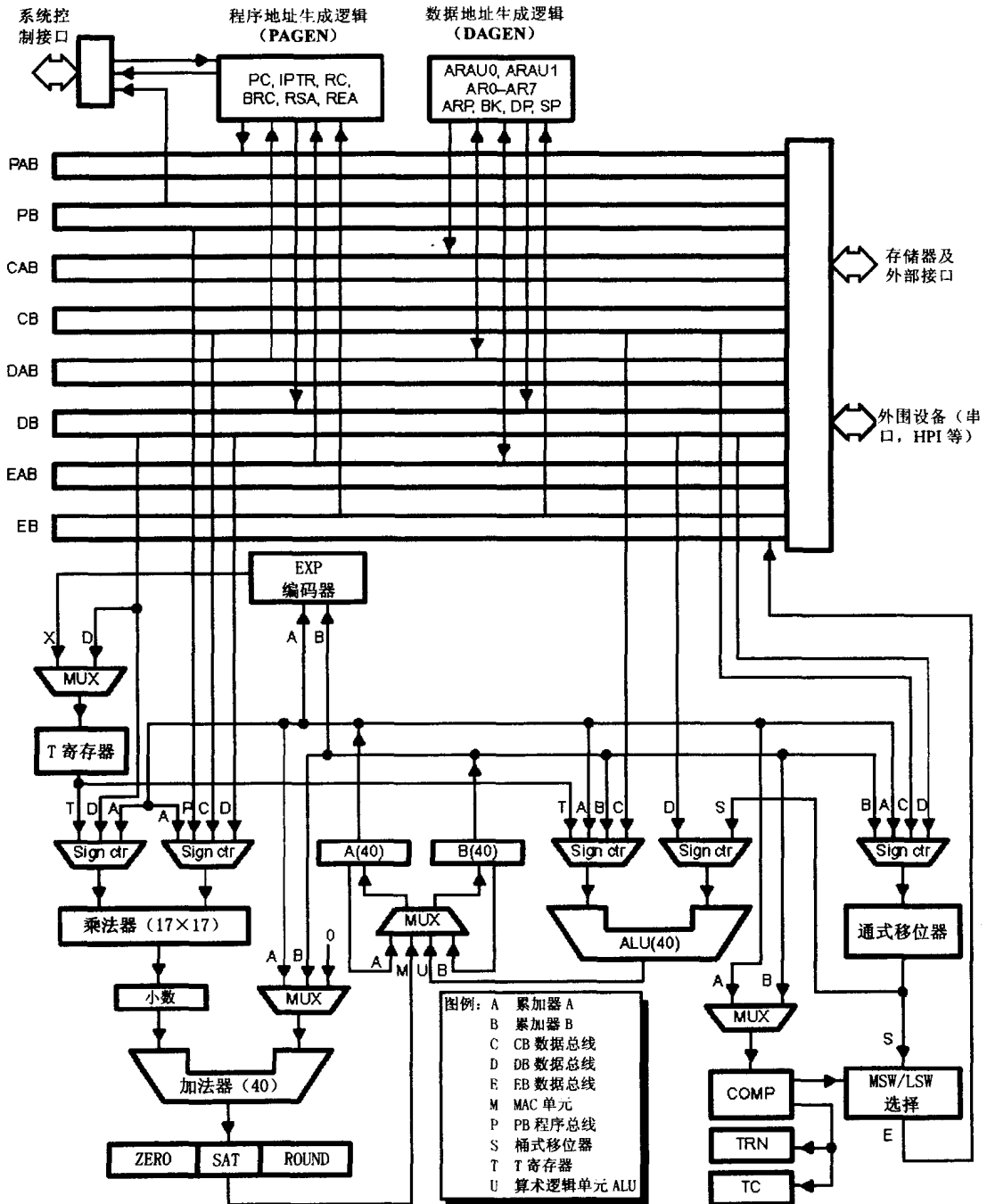


图 1-2 TMS320C54x 的内部硬件结构图

当中断产生后，C54x DSP 会挂起它的主程序，而执行中断服务程序 (ISR)。一般地，中断由需要取数据或给数据的硬件产生（例如 ADC、DAC 和其他处理器等）。

1.3 总线结构

C54x DSP 片内有 8 条 16 位的总线：4 条程序/数据总线和 4 条地址总线。这些总线的功能如下：

- 程序总线 (PB) 传送取自程序存储器的指令代码和立即操作数。
- 3 条数据总线 (CB、DB 和 EB) 将内部各单元 (如 CPU、数据地址产生逻辑、程序地址产生逻辑、片上外设以及数据存储器) 连接在一起。
 - CB 和 DB 传送从数据存储器读取的操作数。
 - EB 传送写到存储器的数据。
- 4 条地址总线 (PAB、CAB、DAB 和 EAB) 传送执行指令所需的地址。

C54x DSP 可以利用两个辅助寄存器算术单元 (ARAU0 和 ARAU1)，在每个周期内产生两个数据存储器的地址。

PB 能够将存放在程序空间 (如系数表) 中的操作数，传送到乘法器和加法器，以便执行乘法/累加操作，或通过数据移动指令 (MVPD 和 READA 指令) 传送到数据空间的目的地。此种功能和双操作数的读特性，支持在单个周期内执行 3 个操作数指令 (如 FIRS 指令)。

C54x 还有一条片内双向总线，用于寻址片内外设。这条总线通过 CPU 接口中的总线交换器连接到 DB 和 EB。使用这条总线的读/写访问需要 2 个或更多个周期，具体时间取决于外设的结构。

表 1-4 列出了各种寻址方式所用到的总线。另外，C54x DSP 可以寻址多达 64K 字的数据存储空间，64K 字的程序存储空间 (某些器件可以多达 8M 字) 和 64K 字的 16 位并行 I/O 端口。对外部存储器和 I/O 端口的访问，通常通过外部接口来实现。 $\overline{DS}$ 、 $\overline{PS}$ 和 $\overline{IS}$ 信号可以用来实现对各种存储空间的选择操作。

表 1-4 各种寻址方式所用到的总线

访问类型	地址总线				数据总线			
	PAB	CAB	DAB	EAB	PB	CB	DB	EB
程序读	√				√			
程序写	√							√
单数据读			√				√	
双数据写		√	√			√	√	
长数据 (32 位) 读		√ (hw)	√ (lw)			√ (hw)	√ (lw)	
单数据写				√				√
数据读/数据写			√	√			√	√
双数据读/系数读	√	√	√		√	√	√	
外设读			√				√	
外设写				√				√

注：hw=高 16 位字；lw=低 16 位字。

接口的外部准备输入信号和软件产生等待状态允许处理器以不同速度与存储器和 I/O 器件进行通信。接口的保持模式允许外部器件对 C54x DSP 总线进行控制，通过这种方式，外部

期间可以访问程序、数据和 I/O 空间的资源。

外部存储器可以由大部分 C54x DSP 指令来访问。然而，I/O 端口要求使用特殊的指令来访问：PORTR 和 PORTW。

1.4 CPU 状态和控制寄存器

C54x DSP 有 3 个状态和控制寄存器：

- 状态寄存器 0 (ST0)。
- 状态寄存器 1 (ST1)。
- 处理器工作模式状态寄存器 (PMST)。

ST0 和 ST1 中包含各种工作条件和工作方式的状态；PMST 中包含存储器的设置状态及控制信息。因为这些寄存器都是存储器映射寄存器，所以它们可以存放到数据存储器，或者从数据存储器加载它们；处理器的状态可以由子程序或者中断服务程序 (ISR) 保存或恢复。

1.4.1 状态寄存器 ST0 和 ST1

ST0 和 ST1 寄存器的各位可以使用 SSBX 和 RSBX 指令来设置和清除。例如，符号扩展模式可以使用 SSBX 1、SXM 来设置，或者使用 RSBX 1、SXM 来复位。ARP、DP 和 ASM 位可以使用带短立即操作数的 LD 指令来加载。ASM 和 DP 段也可以使用 LD 指令加载数据寄存器的值。

ST0 的位如图 1-3 所示，并且在表 1-5 进行了详细的描述。ST1 的位如图 1-4 所示，并且在表 1-6 进行了详细的描述。

15~13	12	11	10	9	8~0
ARP	TC	C	OVA	OVB	DP

图 1-3 ST0 的位

表 1-5 状态寄存器 ST0 每位所表示的意义

位	位名称	复位后的值	作用和意义
15~13	ARP	0	辅助寄存器(AR)指针。间接单操作数寻址时，ARP 的 3 个位段用来选择当前的辅助寄存器 AR。当 DSP 处于标准模式 (CMPT=0) 时，ARP 总是置于 0
12	TC	1	测试/控制标志位。TC 保存算术逻辑单元 (ALU) 的测试位操作的结果。TC 受 BIT、BITF、BITT、CMPM、CMPR、CMPS 和 SFTC 指令的影响。条件转移、调用、执行和返回指令可根据 TC 状态位 (设置或清除) 的条件来执行 如果下列条件为真，则 TC = 1: ● BIT 或 BITT 测试的位等于 1 ● 当执行 CMPM、CMPR 或 CMPS 比较指令时，所比较的数据存储器的值和一个立即操作数、ARO 和另一个辅助寄存器、或者累加器的高位字和累加器的低位字，它们的比较条件成立 ● 由 SFTC 指令测试的一个累加器的位 31 和位 30 具有不同的值