和调整都是通过 GEL 函数来实现的。

④使用时根据具体浮点数的范围和精度，确定相应 IQn 类型；利用 _IQn(f) 函数将浮点数 f 换成 IQn 格式，并通过库中相应数学函数的直接调用，完成浮点数计算。

虽然 IQmath 库在一定程度上解决了定点 DSP 的浮点数运算问题，但数值范围与精度是一对矛盾体，想要数据得到较大的表示范围，必须牺牲一定的精度；而想要数据获得较高的精度，其表示范围也就相应的减小。所以实际工程中需要根据系统要求，多方面去权衡定标值 Q。

3 F28335 浮点运算功能研究与应用

为解决 F2812 在浮点运算中的不足，TI 推出了浮点型 DSP TMS320F28335，其主要改进是在原 F2812 架构的基础上，增加了 FPU 协处理单元，该单元数据和指令系统在设计遵循了 IEEE-754 浮点标准中单精度浮点数规则 [4, 5]，使其在硬件层面可以直接进行高性能的浮点运算。

开发人员在实际开发过程中，只需要在配置和添加相关浮点指令库之后，即可在软件层面直接对浮点数进行相关操作和运算，而不需要关注定 / 浮点数据等复杂的表达和存储方式，将更多的重心放在算法的性能与可靠性上。

为明确 F28335 在工程中应用中各浮点库的意义和使用关系，论文对 CCS 中编译后 MAP 文件进行解析，研究其映射关系后，总结了如下：

① rts2800_fpu32.lib 为 F28335 浮点运算单元的标准静态支持库，须完成对该库的配置后才能启用 F28335 的 FPU 单元；

② rts2800_fpu32_fast_supplement.lib 是在 rts2800_fpu32.lib 基础上对库中函数性能进行优化和汇编处理后的快速静态支持库，由于其与标准库 rts2800_fpu32.lib 间具有调用关系，需要依赖于标准库；

③ rts2800.IQmath_fpu.lib 对反正切、指数等复杂高阶数学函数进行优化封装，方便专业领域的软件人员编程开发；

④ rts2800.IQmath.lib 为定点 DSP 处理器实现浮点运算的 IQmath 库。

基于以上研究，论文提出一种 CCS3.3 下，配置 F28335 实现浮点运算的高效方法，具体如下：

①在工程编译目录 Proprites->Build 选项中设置 Specify Floating point support 为 Float32。

②添加 rts2800_fpu32.lib 标准库到工程目录下，通过该库的加载保证编译器能够生成正确的 ASM 文件，并启用 FPU（Float Point Unit）处理相应浮点指令。

③在 Linker 链接器下配置标准库文件检索路径，如果 DSP 使用了 rts2800_fpu32_fast_supplement.lib 库，需要在 Linker 链接器中制定标准库和快速库的链接顺序。

④在工程 .cmd 配置文件中制定标准库的存储区和加载段，为其分配程序和数据空间：

```
MEMORY
{
  PAGE0：
…
FPUTABLES  : origin = 0x3FEBDC, length = 0x0006A0
…
}
SECTIONS
{
…
FPUmathTables  : > FPUTABLES,PAGE=0,
TYPE=NOLOAD
…
}。
```

⑤如果开发过程在低版本的 CCS3.3 环境下开展，在进行浮点运算的配置过程中，不仅要将 rts2800_fpu32.lib 和 rts2800_fpu32_fast_supplement.lib 静态库加载到工程，还需要将 CCS3.3 Build Options 构建项中链接器 priority linker switch 选项打开，并在 Link Order tab 中加载支持库，实现 CCS3.3 下 F28335 的 FPU 浮点运算功能。

4 实验分析

实验在 TI 集成开发环境 CCS3.3，YX8003 F28335 实验板及 XDS100 仿真器（如图 1 所示）构成的验证环境下分别进行了组名为 Standard、Fast、IQmath 和 Fixed 的 4 类对照试验。其中：

① RTS 组表示启用 F28335 的 FPU，并配置 rts2800_fpu32.lib 标准浮点库；

② Fast 组为应用 F28335 的 FPU，并配置 rts2800_fpu32.lib 标准浮点库 +rts2800_fpu32_fast_supplement.lib 快速支持库；

③ IQmath 组不应用 F28335 的 FPU，只配置 rts2800.IQmath.lib 浮点支持库，以此来模拟定点 F2812 应用 IQmath 库进行浮点运算效果；

④ Fixed 组不应用 FPU，不配置任何浮点支持库，仅通过编译器优化完成浮点运算功能。

图 1 F28335 开发板及仿真器

4 组实验分别完成正弦函数值 arctan(sin(333.555+x)) 的计算，并通过 CCS3.3 断点插入中周期计数器，统计各组实验花费的 CPU 时钟周期数。实验时为避免编译器在重复计算时对代码性能进行优化，在每轮计算中给正弦参数加入了动态量 x，实验结果如表 1 所示。

通过实验结果对比可以看出，当 loop=1 时，Standard、Fast 和 IQmath 三组实验可以很好完成正选浮点的计算，而此时 Fixed 组时钟开销已经远远超出其他三组；而当 loop=20 时，IQmath 组所需时钟数达到 17411，是 Standard、Fast 两组实验结果的 10 倍以上，而 Fixed 组的结果，因为已超过 CCS 时钟计数功能的最大统计数而无法给出。因此，根据 4 组实验可以得出，在定点 F2812 DSP 上利用 IQmath 库，完成简单的数学浮点运算是方便且高效的；而在复杂度较大且对性能要求较高的任务中，还是要靠专用的高性能浮点型 DSP 来完成。图 2 显示了 Standard、Fast 和 IQmath 三组实验的对比曲线图。

5 结语

论文对 F2812 和 F28335 型 DSP 浮点运算功能进行了研究，提出了具体的配置方法，并通过实验分析了各模式下的性能，为实际工程开发提供了指导。

表 1 仿真实验结果

实验名 / 次数	Standard（单位 cycle）	Fast（单位 cycle）	IQmath（单位 cycle）	Fixed（单位 cycle）
loop=1	69	65	122	2360
loop=5	416	414	1667	12930
loop=10	862	857	3398	25993
loop=20	1597	1518	17398	—
loop=50	4032	3988	—	—

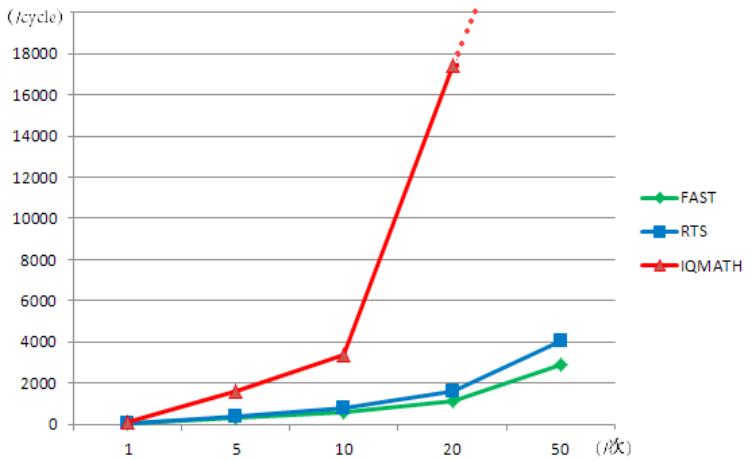

图 2 仿真实验结果对比图

参考文献

[1] 张卿杰,徐友.手把手教你学习DSP-基于TMS320F28335[M].北京:北京航空航天大学出版社,2015.

[2] 孙丽明.TMS30F2812原理及其C语言程序开发[M].北京:清华大学出版社,2008.

[3] C281x C/C++ Header Files and Peripheral Examples,Texas Instruments,2003.

[4] TMS320C28x-Floating-Point-Unit-and-Instruction-Set,TexasInstrument,june-2007.

[5] 许瑾晨,郭邵忠,浮点数学函数异常处理方法[J].软件学报,2015,26(12):3088-3103.

软件复用过程及关键技术研究

Research on Software Reuse Process and the Key Techniques

刘晓佳[1] 左强[2] 刘晶[1]

Xiaojia Liu[1] Qiang Zuo[2] Jing Liu[1]

1. 北京青云航空仪表有限公司 中国·北京 101300

2. 空军装备部驻北京地区第五军事代表室 中国·北京 101300

1. Beijing Qingyun Aviation Instrument Co., Ltd., Beijing, 101300, China

2. The Fifth Military Representative Office of the Air Force Equipment Department in Beijing, Beijing, 101300, China

摘 要：为解决软件危机问题，软件复用及相应技术应运而生。通过软件复用可以有效提高软件生产率和软件质量。论文介绍了软件复用涉及的可复用软件构件的开发活动以及基于可复用软件构件的软件开发过程，并对其中涉及的主要关键技术如在工程和领域工程活动进行介绍。同时，论文介绍资产管理的相关内容和要求，为软件复用实施提供实际的指导。

Abstract: In order to solve the problem of software crisis, software reuse and related technologies emerge at the historic moment. Software reuse can effectively improve software productivity and quality. This paper introduces the process of the development of the reusable software and the process of the development based on reusable software. This paper introduces the key techniques such as re-engineering and domain engineering. At the same time, this paper introduces the contents and requirements of asset management to provide guidance for the implementation of software reuse.

关键词：复用；领域分析；再工程；资产管理

Keywords: reuse; domain engineering; re-engineering; asset management

DOI: 10.12346/etr.v5i3.7809

1 引言

为解决“软件危机”问题，降低软件开发和维护的成本，20 世纪 60 年代末在 NATO 软件工程会议上，首次提出了软件复用的概念[1]。软件复用是将已有的软件成分用于构建新的软件系统的方式，是在软件开发过程中避免重复性劳动的有效解决方案。

通过软件复用活动使得软件的开发不再“从零开始”，可在已有软件资产的基础上实现高效、高质量的开发工作。经长期实践证明软件复用是提高软件生产率和质量的重要手段。一方面，通过复用可以有效减少重复性工作量，提高软件开发的效率；另一方面，通过复用高质量的劳动成果，可以有效避免新开发过程中引入的错误，从而提高软件质量。

2 复用的分类和方式

2.1 复用的分类

根据复用的对象，软件复用可分为产品复用和过程的复用。产品复用是指复用已有的软件构件，通过对构件的组装和集成得到新的应用软件。过程复用是指复用已有的软件开发过程，对过程的复用往往需要依赖自动化工具。

目前，现实、主流的复用内容是对产品的复用，对过程的复用目前仅在一些特殊的应用领域中应用。产品复用的内涵比较广泛，包括需求分析成果、软件设计方案、数据结构、源代码、测试用例等内容。论文中主要介绍对产品的复用。

2.2 复用的方式

可复用软件构件的复用方式分为黑盒复用和白盒复用两种方式。黑盒复用是指对已有的构件不进行任何修改，直接进行复用；白盒复用是指根据待开发软件系统的具体需求，

【作者简介】刘晓佳（1989-），女，中国河北石家庄人，硕士，工程师，从事导航与制导控制、嵌入式软件研究。