模块 3

实验 3：ARM Cortex M 架构

实验 3：ARM Cortex M 架构

3.0 目标

本实验的目标是介绍 Cortex M 架构。

1. 您将学习有关寄存器、RAM 和 flash ROM 的知识。
2. 您将编写一个含有输入和输出参数的汇编函数，其中包含条件执行和数值运算。
3. 您将学习利用单步运行、断点和查看窗口进行调试。
4. 您将使用一种名为黑匣子函数的自动测试方法来验证您的算法是否正确。

小知识：我们将用 C 语言编写机器人的软件代码。然而编译器将会把 C 代码转化为汇编代码。真正运行在基于 Cortex-M 架构的 MSP432 微控制器上的是底层汇编代码。在本实验中，您将体验到软件究竟是如何在微控制器中被执行的。了解底层的细节有助于您成为一个更好的顶层软件开发者。

3.1 入门

3.1.1 从下面的软件工程起步

浏览以下 3 个工程：
SimpleProject_asm （一个简单的随机数生成函数）
LinearInterpolation_asm （计算正弦值）
Lab_Assembly （本实验的起步工程）

3.1.2 参考资料（在 datasheets 文件夹内）

spmu159a.pdf , Cortex-M3/M4F Instruction Set

3.1.3 阅读材料

Volume 1 Section 1.7, Chapter 3, and Section 5.3
Embedded Systems: Introduction to the MSP432 Microcontroller",
或
Volume 2 Sections 1.1, 2.1, and 2.5
Embedded Systems: Real-Time Interfacing to the MSP432 Microcontroller",

3.1.4 本实验所需组件

Quantity	Description	Manufacturer	Mfg P/N
1	MSP-EXP432P401R LaunchPad	TI	MSP-EXP432P401R

3.1.5 所需实验设备（无）

3.2 系统设计要求

本课程中您将学到很多解决机器人挑战所需的知识。本实验的目标是是更深入地理解处理器运行的过程。您将用 C 语言编写后续的机器人程序，然而在本实验中您将编写一些简单的汇编程序。

注：在机器人挑战环节中您将一种距离传感器，它是由红外传感器和集成的位置探测器构成的。这种传感器也被叫做接近传感器，在机器人上，它一般被用于测量距离。

在模块 4 的实验中您将编写一个 C 语言程序，用于将 GP2Y0A21YK0F 接近传感器的 ADC 采样值转换成距离。

设 **n** 为 14 位 ADC 的采样值（从 0 到 16383），且 **D** 为距离，单位是 mm。二者之间存在如下的非线性转换关系

$$D = 1195172/(n - 1058)$$

其中 1195172 和 -1058 是校准系数的经验值，可以通过模块 15 的实验，也就是 ADC 实验取得。

传感器可以测量的最大距离为 800 mm，因此如果 ADC 值小于 2552，您的函数应返回 800。函数的 C 语言原型是

int32_t Convert(int32_t n);

然而，鉴于您要使用汇编语言编写此函数，您必须遵从一个程序编写标准，名为 ARM 架构过程调用标准（**ARM Architecture Procedure Call Standard**，简称 AAPCS）。这个标准由很多部分组成，但与本实验相关的主要有下列部分：

- 如果有 1 个输入参数，它将被传入 R0
- 如果有 2 个输入参数，它们将被传入 R0 和 R1
- 如果有 3 个输入参数，它们将被传入 R0-R2
- 如果有 4 个输入参数，它们将被传入 R0-R3
- 如果有一个输出参数，它将被返回至 R0
- 函数可以自由修改 R0-R3，还有 R12
- 如果一个函数想要使用 R4-R11，那么它必须使用栈来保存并还原它们
- 如果一个函数调用了另一个函数，那么它必须保存并还原 LR
- 函数必须保证栈平衡

遵从上述标准将是您能够开发能被 C 语言调用的汇编程序，而且允许您的 C 代码被汇编程序调用。具体来说，编译器在生成机器代码时将遵从这个标准。

3.3 实验准备

本实验使用 LaunchPad，但无需任何输入输出硬件设备。

3.4 实验步骤

3.4.1 函数和调试

在本实验中您将对 **SimpleProject_asm** 例程进行编译和调试。请在主程序中使用单步调试功能，同时观察函数的输入和输出参数。

回答下列问题：

i) 数据是如何传入 **Seed** 函数中的？
ii) **Rand** 函数的结果是如何返回的？
iii) 当一个函数被调用时，LR 寄存器会发生什么现象？
iv) 函数如何返回？
v) 软件如何对 RAM 进行读写？
vi) 把数据存在寄存器中和存在全局 RAM 中有什么区别？
vii) 机器代码存在什么地方？
viii) **.data** 和**.text** 有什么含义？
ix) 常数 1664525 和 1013904223 存储在什么地方？
x) 您可以通过将变量 M 和 n 的地址放入 **Memory Browser** 窗口来查看它们的值。
xi) 使用 step-over 命令，执行几次 Rand 函数，并观察 M 和 n 的值。特别注意观察 M 的第 0 位，在第 0 位中您观察到什么规律？

接下来您将编译并调试 **LinearInterpolation_asm** 工程。如果您对“线性插值”这个概念不熟悉，请先到网络上搜索一下相关的话题，并尝试对它有一个基本的了解。

在 **Sin** 函数中放置一个断点，并使用调试器观察每次执行 Sin 函数后**寄存器**值的变化。从编程理论的角度出发，这些寄存器应被看作函数的**局部变量**。

回答下列问题：

i) 您是否能证明以下三个减法指令 (Ix-x1), (y2-y1) 和 (x2-x1) 永远不会溢出？
ii) 您是否能证明在计算 (y2-y1)*(Ix-x1) 时乘法指令永远不会溢出？
iii) 您是否能证明此函数永远不会除以 0？
iv) 此函数中为什么要用 SDIV，而不是 UDIV？

观察主程序是如何测试 **Sin** 函数的。我们把 main.asm 中的测试方法称为**黑匣子**函数测试，因为这种测试方法仅设置输入，并观察输出。换言之，我们是从函数外部来观察它，而不对内部做任何探测。黑匣子测试方法在不探究软件内部如何工作的前提下对它进行整体测试。

3.4.2 距离转换

写一个汇编函数，将 14 位 ADC 数据转换为以 mm 为单位的距离数据。使用**.field** 声明来封装校准参数。

```
IRSlope  .field 1195172,32
IROffset .field -1058,32
IRMax    .field 2552,32
```

您可以使用 **Lab_Assembly** 工程中的主程序来测试您的 **Convert** 函数。与 **LinearInterpolation_asm** 类似，这里也用了黑匣子测试方法。这个测试程序包括 16 种测试情况（输入及期望输出）。期望的结果如图 1 所示。

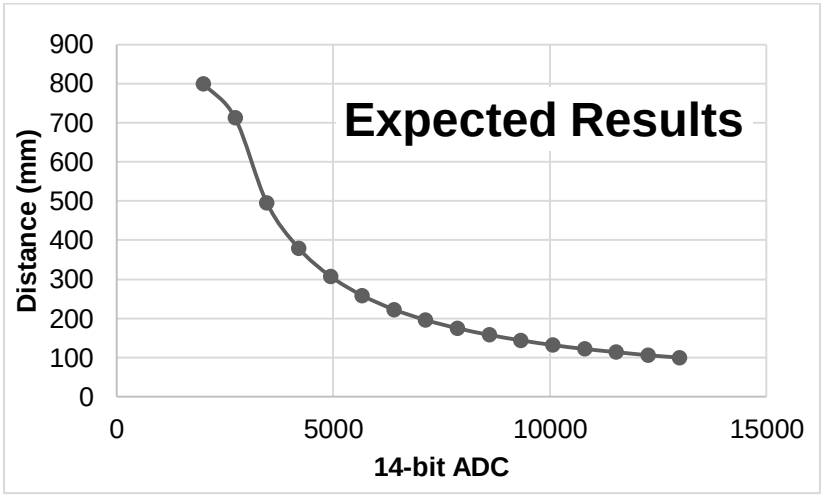

图 1. GP2Y0A21YK0F 转换结果的期望值

运行 **main** 函数，并将您的运行结果与期望结果相比较。如果您的运行结果与期望结果相差±1 的话，是可以接受的（误差可能是四舍五入造成的）。

3.4.3.查看由编译器生成的汇编代码

重新查看实验 1 中您运行过的 C 语言例程。在调试器中打开一个汇编（disassembly）窗口。单步运行 C 代码，并观察实际的汇编指令

3.5 疑难解答

转换不成功

- 使用 **main** 函数，找到无法正常转换的输入值，写一个 main 函数，在其中仅调用您的函数计算该输入值的转换结果，然后使用单步调试对比函数内部的计算结果和期望结果。观察内部计算结果的方法叫做**白匣子**测试。
- 如果仍有其他 bug，请与您的老师或同学进行讨论。您可能从和他们的讨论中获得寻找问题解决方案的全新视角。

3.6 请思考

本环节中将列出一些问题，供您在完成实验之后思考。这些问题是为了检验您对于本实验中所涉及到的概念的理解。本模块的目标是让您掌握足够的汇编语言技巧，以帮助您能够看懂有编译器生成的汇编程序。

- ROM 中存储的是什么信息？为什么？
- RAM 中存储的是什么信息？为什么？
- R0-R12 寄存器中存储的是什么信息？为什么？
- R4-R11 与 R0-R3 及 R12 有何不同？
- 如何使用 LR 寄存器？
- 如何使用 SP 寄存器？
- 如何使用 PC 寄存器？
- 函数是如何工作的？输入参数和输出参数呢？
- 您是否能证明 (**n** – 1058) 的减法永远不会溢出？
- 您是否能证明除法运算永远不会尝试将一个数除以 0？
- 对于任意大于 1 的数字 **n**，使用整型除法计算 **1/n** 的结果？这个错误（信息丢失）叫做丢弃（**dropout**）。
- 输入是 14 位数字（从 0 到 16383），但输出仅为 10 位数字（从 0 到 800）。这 4 位的减少是一种轻微的丢弃（**dropout**）。请问您有什么办法可以减轻这个丢弃的情况？
- 请注意 **SimpleProject_asm** 工程 只用了一个源文件，但 **LinearInterpolation_asm Lab_Assembly** 有两个源文件。这两个源文件是如何被使用的？把执行代码和测试代码分开有什么好处？
- 列出本实验中所使用的调试技巧。

实验 3：ARM Cortex M 架构

3.7 其他挑战

本环节中将列出一些与本章知识点有关的挑战项目，您可以在尝试去完成它们。您可以对系统进行扩展，或提出全新的问题。例如：

- 考虑如何用穷举法测试从 0 到 16383 的所有 14 位输入。如何生成测试条件？如何更改主函数？穷举法的优点有哪些？
- 机器人可能有多个接近传感器。重新设计 **Convert** 函数来处理 3 个传感器的情况，其中每个传感器都有单独的一组校正系数（IRSlope IROffset IRMax）。
- 使用调试器来估计执行 **Convert** 函数所需的时间。
- Cortex M 处理器支持浮点数运算。开发一个本实验的浮点数版本，并开发一个方法来进行测试。比较两个版本的精度和执行时间。

3.8 接下来是哪些模块？

我们将通过接下来几个实验来创建控制机器人所需的部件。输入/输出是嵌入式系统的重要部件。以下模块将在本模块的基础之上建立：

模块 4) 介绍 C 语言并开发机器人所需的一些函数
模块 5) 开始搭建机器人，包括电池和电源转换
模块 6) 学习如何使用微控制器的引脚进行输入和输出
模块 7) 学习使用有限状态机来控制机器人
模块 8) 用微控制器与开关和 LED 进行交互。这将扩展输入输出的功能并增加系统复杂度。

3.9 您应该已经学会

本环节中我们来回顾一下本模块中您应该已经学会的重要概念：

- 理解处理器在执行过程中是如何使用寄存器的
- 明白 RAM 与 ROM 的区别，以及软件如何使用这二者
- 使用汇编语言进行加减乘除运算
- 理解微控制器中常数是如何储存的
- 学会如何使用条件分支汇编语句
- 使用调试器进行单步调试和变量查看
- 进行函数测试

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources 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.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

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