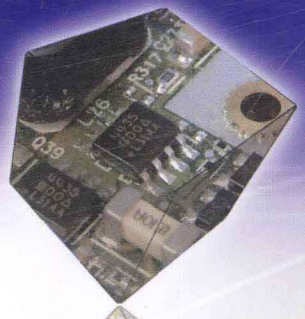
嵌入式技术与应用丛书
ARM AAE认证考试教材



ARM认证工程师应试指南

北京通联物网教育咨询有限公司　奚海蛟　谌　利　编著

- 本书面向ARM认证工程师考试，给出了考试的要点及其要求掌握的程度
- 本书内容基于ARM Cortex-A系列处理器和Linux操作系统
- 本书按照知识点展开描述，便于读者尽快适应ARM认证工程师考试

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

嵌入式技术与应用丛书
ARM AAE 认证考试教材

ARM 认证工程师应试指南

北京通联物网教育咨询有限公司　奚海蛟　谌　利　编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书是针对 ARM AAE 考试认证编写的，旨在为对参加 AAE 认证考试的读者提供有益的帮助，对汇编语言和 C 语言的程序员提供有用的信息。

本书分为 ARM 认证工程师学习指南和 Cortex-A 系列程序员指南两部分。在 ARM 认证工程师学习指南部分给出了 AAE 认证考试的要点及其要求掌握的程度；在 Cortex-A 系列程序员指南部分对第 1 部分给出的要点进行了详细论述，内容涵盖：ARM 简介，ARM 体系结构和处理器，工具、操作系统和开发板，ARM 寄存器、模式和指令集，汇编语言简介，ARM/Thumb 的统一汇编，浮点，NEON 简介，高速缓存，内存管理单元，内存访问排序，异常处理，中断处理，其他异常处理程序，引导代码，移植，应用程序二进制接口，性能分析，优化运行在 ARM 处理器的代码，编写 NEON 代码，多重处理器简介，SMP 架构考虑，并行软件，并行软件的问题，电源管理，安全性，虚拟化，big.LITTLE 简介，调试。

本书可供参加 AAE 考试认证的读者阅读，也可作为 Cortex-A 系列程序员的参考用书。

图书在版编目（CIP）数据

ARM 认证工程师应试指南/奚海蛟，谌利编著. —北京：电子工业出版社，2013.9
（嵌入式技术与应用丛书）
ISBN 978-7-121-21361-8

I. ①A… II. ①奚…②谌… III. ①微处理器－系统设计－工程技术人员－资格考试－自学参考资料
IV. ①TP332

中国版本图书馆 CIP 数据核字（2013）第 206662 号

责任编辑：田宏峰　　特约编辑：牛雪峰
印　　刷：三河市鑫金马印装有限公司
装　　订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
开　　本：787×1 092　1/16　印张：16.5　字数：422 千字
印　　次：2013 年 9 月第 1 次印刷
印　　数：4 000 册　　定价：49.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

ARM Accredited Engineer（AAE）是ARM提出的认证工程师计划。ARM为认证工程师计划提供了一个ARM系列知识的考试，旨在为全世界任何公司范围内的工程师提供一个“ARM认证工程师”水平及能力测试并授予资格认证的机会。AAE项目是为希望获得ARM架构专业知识可靠认证的工程师提供的唯一全球性认证项目，是评估ARM架构知识的统一标准，有助于招聘经理对求职者进行基准测试，亦有助于工程师从众多求职者中脱颖而出。ARM定义了一系列不同的考试，包括各种不同的学科领域和难度水平。

AAE认证主要考查ARMv7架构软件相关方面的知识，尤其是Cortex-A和Cortex-R部分，要求广泛了解ARM技术、侧重于应用处理器和实时处理器的一般嵌入式软件和系统开发人员。

本书就是针对AAE认证考试编写的，为使用ARM公司ARMv7-A体系结构的Cortex-A系列处理器的程序员介绍ARM技术。ARMv7是指体系结构的第7版本，A是对应用程序处理器的体系结构描述，包括Cortex-A5、Cortex-A7、Cortex-A8、Cortex-A9和Cortex-A15等处理器。

本书分为两个部分：ARM认证工程师学习指南和Cortex-A系列程序员指南。第1部分学习指南给出了AAE认证考试的要点及其要求掌握的程度。第2部分Cortex-A系列程序员指南针对第1部分给出的要点进行了详细的讲述。

本书汇集了各种信息与知识点，为想要开发最新Cortex-A系列处理器应用程序的程序员提供统一的指导，而且涵盖了硬件概念，如高速缓存和内存管理单元。本书的目的对于参加AAE认证考试的读者提供有益的帮助，对汇编语言和C语言的程序员提供有用的信息。

本书分为两个部分：ARM认证工程师学习指南和Cortex-A系列程序员指南。

第1部分学习指南给出了AAE认证考试的要点及其要求掌握的程度。

第2部分Cortex-A系列程序员指南针对第1部分给出的要点进行了详细的讲述。第2章～15章就AAE考试认证的要点进行了详细的讲解，内容包括ARM简介，ARM体系结构和处理器，工具、操作系统和开发板，ARM寄存器、模式和指令集，汇编语言简介，ARM/Thumb的统一汇编，浮点，NEON简介，高速缓存，内存管理单元，内存访问排序，异常处理，中断处理，其他异常处理程序；第16～30章就AAE考试认证提供更加高级的编程信息，内容包括引导代码，移植，应用程序二进制接口，性能分析，优化运行在ARM处理器的代码，编写NEON代码，多重处理器简介，SMP架构考虑，并行软件，并行软件的问题，电源管理，安全性，虚拟化，big.LITTLE简介，调试。

参与本书编写工作的还有：刘张辉、李政春、滕忠楠、李晓庆、付盈、乔林、吴飞、王秀文、王丽娜和陈晓冬。

由于时间仓促以及作者的水平，书中错误和不足在所难免，希望广大读者批评指正。

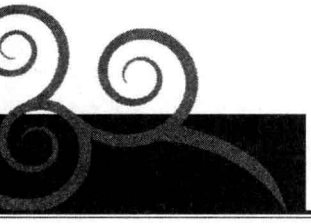
编　者

2013年8月

目　录

第 1 部分　ARM 认证工程师学习指南

第 2 部分　Cortex-A 系列程序员指南



第 1 部分

ARM 认证工程师学习指南

第1章

学习指南

1.1 ARM 认证工程师介绍

ARM 工程师认证项目为全世界的工程师和开发人员提供了一系列认证考试，使之有机会成为 ARM 认证的工程师。

这个项目定义了一系列不同范围内的考试，涵盖了不同的知识面和不同的难度等级。本章是关于 AAE 认证考试的第一部分测试，主要关注 ARMv7-A 和 ARMv7-R 架构系列的软件方面。

本章旨在向读者解释如何使用“Cortex-A 系列程序员指南”（后面简称“程序员指南”）并参考入门级大纲（可从 ARM 网站下载）的详细要求来学习知识点，本章需要和“程序员指南”配合阅读，解释了大纲每个部分的范围以及怎样找到这些大纲范围的必要的详细信息。要求学习本章和大纲的读者已经具备了相当的 C 语言和汇编语言编程，以及嵌入式系统和微处理器的相关知识。

本章仅作为实际应用实验，在真实 ARM 平台上进行软件开发的课程的有益补充，假定读者已熟悉 ARM 的软件开发工具，且主要是基于 ARMv7-A 和 ARMv7-R 架构的处理器以及其他由此类架构派生出来的 ARM 处理器系列。

1.2 ARM 认证工程师大纲概述

读者需要熟悉 AAE 入门级大纲的文档（可从 ARM 网站下载，文档编号 AEG0052C）。

大纲详细提供了一系列详细的知识考点，并使用颜色标注了不同的难度级别，读者应该牢固掌握标注绿色的知识点，清楚了解标注黄色的知识点。

本章将一步步地解释大纲，并会提供每个知识点在“程序员指南”中对应的位置，并提供“程序员指南”没有覆盖的考试要求的背景知识。要注意的是，初学者无须按照它的顺序来学习新的知识，但是一般需要先熟悉 ARM 的指令、寄存器和模式，然后学习中断控制器及相关硬件。

1.3　大 纲 详 述

在本节，学生可以了解每个知识点的详细要求。引用大纲的文字采用斜体，紧跟着正体的文字将解释大纲的内容，包含如何找到相关信息，以及一些必要的额外信息。

1.3.1　实现

1. ARM 处理器家族

（1）什么是多内核

Candidates should be familiar with the available processors from ARM and know which of these may be used in multiprocessor configurations.

考生需要熟悉 ARM 的处理器，并且了解其中哪些是用于多处理器配置的。

本书 3.4 节描述了一些常见的 ARM 处理器，3.5 节更详细地描述了 Cortex-A 系列的处理器，尤其需要详细了解表 3-3。

（2）连接硬件

Candidates should be aware of the concept of coherency hardware, what it does, why it is required and what are the benefits of using it.

考生需要了解连接硬件的含义，它是做什么的，为什么需要它以及它的优势是什么。

虽然这部分内容是在大纲的开始部分提及的，但它属于难度较大的知识点。连接硬件是在多核系统中必须使用的。要理解它是如何工作的以及它是如何影响软件开发的，需要具备对多核系统以及缓存知识的了解。这部分内容位于第 10 章（缓存）及第 22 章（介绍多核系统），连接硬件是在 23.1 节介绍的。

（3）通用中断控制器 GIC

通用中断控制器 GIC 在“编程手册”的 14.2 节介绍，考生应该同时参考 23.3 节（多内核系统中断处理）来学习。GIC 常用于 Cortex-A5 多内核、Cortex-A9 多内核以及 Cortex-R7 多内核处理器中，它也可作为可选部件存在于 Cortex-A7 和 Cortex-A15 内核中。

（4）架构版本归属

Candidates must be able to identify which ARM processors conform to which Architecture versions. Architectures from ARMv4T onwards are included, with the exception of ARMv7-M and ARMv6-M architectures.

考生需要能够识别具体的 ARM 的处理器是属于哪个处理器架构版本的。ARMv4T 架构之后的都需要了解。

表 3-2 提供了相关的详细信息

（5）性能

Candidates must be able to distinguish between applications, real-time and microcontroller

profiles, be aware of the distinguishing features between them and understand the typical target market/applications of each.

考生需要能够区别 ARM Cortex 的应用、实时和微控制器三大系列的区别，理解它们之间的不同特性以及所针对的典型目标应用和市场。

《Cortex-A 系列程序员指南》，顾名思义，它只介绍了 Cortex-A 系列，并没有提供很多关于 Cortex-R 和 Cortex-M 系列的知识。Cortex-A 系列处理器主要应用于需要 MMU 内存管理单元和缓存工作的操作系统的应用场合，要求拥有高性能的处理能力。

Cortex-R 系列处理器主要面向实时应用，它面向需要硬实时反应能力的嵌入式系统。这说明它的特性要求有快速、确定的中断响应，有紧密耦合的存储器（TCM）位于处理器的局部快速总线上以提供快速响应的代码和数据，有奇偶校验或者 ECC 校验机制来保证错误的检测和修正。Cortex-R 系列处理器使用内存保护单元 MPU 代替 Cortex-A 系列中的内存管理单元 MMU 来进行内存的管理和保护。

Cortex-M 系列处理器面向低成本和功耗敏感的微控制器和混合信号系统。例如一些终端设备，包括智能电表、人机交互设备、自动和工业控制系统、大型家用电器、消费类产品和医疗仪器等。这类领域应用的关键是要求微量代码（更好的代码密度），易于使用以及节能。例如，Cortex-M0+内核是 ARM 处理器中能耗最低的内核，可达到 mW/MHz。Cortex-M 系列处理器仅支持 16 位和 32 位的 Thumb 指令集，通常不含缓存。

2. 指令周期时序

Candidates should be aware of the effects of pipeline and memory system on instruction execution timing.

考生需要了解流水线和内存系统对指令执行时间的影响。

流水线在“程序员指南”的 5.4 节有一些介绍，分支预测在 5.5 节介绍。这些都提供了一些流水线架构相关的基本知识。

了解 ARM7TDMI 的三级流水线（取指、译码、执行）的含义可以帮助理解流水线对执行周期时序的影响。ARM7TDMI 技术参考手册给出了这个流水线，这让理解流水线的概念变得更容易。

例如，在 ARM7TDMI 上分支跳转会导致 2 个周期的损失，因为分支跳转发生在流水线的执行阶段，此时位于取指和译码阶段的指令将被冲刷掉，流水线被重新按新的指令流填满，因此在跳转后的第一条指令执行前将产生 3 个周期的延迟。

同理，一旦理解了简单的流水线，“加载-使用损耗”的概念就很明了了，它用于描述当寄存器加载一个存储空间的值时的状况。如果接下来的指令需要使用这个寄存器的值，那么它需要等待该寄存器完成数据的加载（并到达处理器流水线的相关级，可能通过专用通道到达）。编译器（或者汇编器）会避免这种情况发生，它一般会试图把使用加载数据的指令从加载指令处分离开。

1.3.2 软件调试

1. 标准调试手段

（1）软件断点和硬件断点的区别

Candidates should be aware of the differences and limitations between Hardware and

Software breakpoints. They should understand the usage of watchpoint units in configuring these two types of breakpoint.

考生需要清楚硬件和软件断点的区别以及各自的限制，需要了解在配置这两种类型的断点时观察点单元的使用。

这部分在 30.1 节有详细介绍。在较早的处理器（如 ARM7TDMI）中，并没有配备专用的断点指令 BKPT，而是使用设置观察点单元来查看特定的位模式的，调试器可以在 RAM 的代码空间中设置无限量的软件断点。调试器会读取需要放置断点指令的操作码，并保存到主机上。这条指令然后被一个特定的数据位替代（通常对应的是一个无效指令）。通常，这需要处理器的数据缓存被清空并且使指令缓存无效，以确保从处理器数据端写入的特定数据可以被取指逻辑正确访问。在较新的处理器上，包括所有的 Cortex 系列处理器，则采用了 BKPT 指令来代替原来的需要放置断点的操作码。当断点被移除时，原本存储在主机上的原操作码会被写回。与此对应的是，硬件断点可以被设置在任何类型的存储器上（包括 RAM 和 ROM），但它的数量则由硬件限制。

在学习这部分内容时，建议使用真实的调试器来理解它的使用和限制，获得一些实际的应用经验，这可以比书本上的解释提供更多的有益背景知识。

（2）监控模式与停止模式调试

Candidates should be aware of the differences between debugging in each of the two modes and when one would be chosen over the other e.g. choosing monitor mode if interrupts must continue to be serviced while debugging.

考生需要了解调试的两种模式之间的区别以及何时选用哪种模式。例如，需要在调试过程中仍旧响应中断时，则应选择监测模式来调试。

这一部分在 30.1.1 节有介绍。

（3）矢量捕捉

Candidates should be aware of the function of this feature, why and when it is typically used.

考生需要了解矢量捕捉的功能，为什么以及何时需要使用它。

矢量捕捉是调试器捕获处理器异常中断的方法，早期开发时在异常中断服务句柄被取指之前捕获到处理器的异常中断。许多 ARM 的处理器都由矢量捕获硬件来完成这一任务。而在另外一些处理器，如 ARM7TDMI，调试器可以采用在异常矢量表的适当位置设置断点的方法来进行中断矢量捕获。

（4）判别异常触发原因（如 DFSR、SPSR、Link 链接寄存器）

Candidates should understand how to determine the actual cause of an exception by using additional resources, which might include special registers that contain fault status. For example, using the DFSR, DFAR to determine cause of a Data Abort or using the SPSR and LR to locate and retrieve SVC comment field.

考生应当理解如何使用额外的资源来判别触发一个异常中断的原因，这些资源通常包括含有错误状态的特殊寄存器，如使用 DFSR、DFAR 来判别数据中止的原因，采用 SPSR 以及 Link 寄存器来定位并获取 SVC 系统调用的参数位。

当调试一段软件时，通常有必要了解为何特定的处理器异常会发生。为此 ARM 处理器提供了一系列的寄存器来提供有用的信息。在异常中断中，当前异常模式的链接寄存器（LR）会给出主程序中最靠近触发异常的指令的位置。通常触发异常的指令的位置是 LR 寄存器的

值减去 4 或者 8（详见 13.2 节）。类似地，SPSR 寄存器的 mode 位可以给出在进入异常模式之前处理器所处的模式。

对于特定的模式，还有一些额外的信息。在取指异常或数据中止异常后，CP15 寄存器的错误状态寄存器（FSR）和故障地址寄存器（FAR）是值得注意的，它们可以给出异常中止发生的原因（如一个外部存储器错误或者该地址的转换表项无效）以及产生异常中止的内存访问操作的地址。

注意

ARMv6-M 和 ARMv7-M 架构在异常处理模型上与其他的 ARM 内核有所不同，但在基本的考试中这些并不要求掌握。

（5）跟踪

Candidates should be aware of what Trace is, what on-chip and off-chip components are required to make it work, what it is used for. Also that it is non-intrusive.

考生需要了解什么是跟踪，它正常工作需要哪些片上和片外模块，它是用来做什么的，以及它是属于非侵入式的。

这些都在 30.2 节中介绍。

（6）交叉触发

考试对这部分不做要求

（7）物理调试接口

Candidates should be aware of available options for making physical connections to the target for the purpose of run-control debug (e.g. JTAG, SWD)

考生需要理解在线调试所需要的连接到物理目标系统的物理连接选项（如 JTAG、SWD）。

这部分在第 30 章介绍。JTAG 是一个广泛使用的工业标准，它在物理上通常需要至少 5 个信号引脚来连接主机计算机和 ARM 目标系统板。串行调试口 SWD 是一个拥有类似调试功能的 2 脚调试口，并且它通常只在较新的处理器上才有。而从处理器抓取跟踪信息，则需要比控制代码执行大得多的带宽，因为每个周期都会有很多的 32 位地址和数据信息产生，因此专用的跟踪端口需要更多的引脚。此外，跟踪信息也可以存储在片上的缓冲区中，采用较少物理引脚的慢速调试口来读取这些跟踪信息。

（8）调试访问内存

Candidates should be aware of how a debugger uses the processor to access memory and be aware of how memory translation affects this. They should also understand the potential impact of cache maintenance caused by debug operations on the state of the memory system.

考生需要了解调试器是如何使用处理器来访问内存的，并且理解内存转换如何影响这种访问，还需要理解对内存系统状态的调试操作可能带来对缓存内容维系的潜在影响。

调试器通常需要显示调试目标内存（或者外设）的内容并进行修改，它通常是通过处理器执行加载或者存储指令来实现的，然而有些系统则可以通过内建的调试系统单元来直接对内存进行写（或读）操作，不需要处理器本身参与内存操作。

一个良好的调试器会减少调试动作对系统的影响。例如，它通常会尽量减小调试时对缓存的修改。如果调试器在调试窗口上显示一块内存区域的内容但调试器必须使用处理器来读

取内存时，它会尽量采用非缓存方式来操作，从而确保之前缓存中的内容不会被冲刷掉。程序员必须了解（当内存管理单元 MMU 被使能时）调试显示使用的是虚拟地址而非物理地址，而且缓存中的内容可能与外部存储器的内容不一致。

如果内存访问是直接通过片上系统的调试单元（而非处理器）来进行的，那么访问使用的就是物理地址而非处理器的虚拟地址转换，而且它是直接访问内存的（绕过处理器的缓存）。

（9）半主机

Candidates should be aware of semi-hosting, understand what its function is and how it can be used during development. They should also be aware of the invasive nature of using this feature and how it may affect timing etc. They should also be aware of the danger of using semi-hosting when no debugger is connected and how to retarget the standard C library.

考生需要理解半主机的功能是什么，在开发过程中应如何使用；需要了解这种机制的侵入式特性以及它对时序等方面的影响；还需要了解当目标板没有连接调试器时使用这一机制的风险，以及它如何重定向到标准 C 库。

半主机是一种让 ARM 目标板上的代码可以使用主机计算机上调试器提供的部分资源的机制。

这些资源可以包含键盘输入、显示输出、硬盘 I/O。例如，程序员可以使用这种机制提供的标准 C 库函数，如 printf()和 Scanf()来使用主机上的终端屏幕和键盘。开发的硬件通常不包含全功能的输入/输出设备，而半主机则可以让主机提供这些资源。

半主机是通过一系列专用的软件指令产生异常来实现的。应用程序使用适当的半主机异常调用——调试中介来响应异常处理。调试中介会提供到主机所需的通信。

半主机的接口一般是通过 ARM 提供的调试单元作为中介来实现的。ARM 的工具使用 SVC 0x123456（ARM 状态）或者 SVC 0xAB（Thumb 状态）来表示半主机调试函数。

当然，如果脱离了开发环境，运行着调试器的主机并不会连接到目标系统上。此时开发人员有必要将任何使用到半主机的库函数重定向，如 fputc()函数。这意味着需要使用能够输出字符到指定设备的代码来替代使用 SVC 调用半主机的库函数代码。

2. 标准调试技术

（1）调用栈

Candidates must understand what a call stack is and how it may be used in debugging.

考生需要理解什么是调用栈以及它在调试时应如何使用。

应用程序代码主要使用堆栈来进行参数传递、保存局部变量和保存返回地址。每个函数压入堆栈的数据都被组织为一个“堆栈帧”，当调试器停止处理器时，它可以分析堆栈中的数据，为程序员提供“调用栈”，这是从最顶层函数调用到当前子函数之间每一层级的调用关系，它可以在调试时让用户非常方便地了解整个调用路径，了解为何程序会运行到当前的位置。

为了能够重建调用栈，调试器必须能够确定堆栈的哪些项包含了返回地址的信息。如果编译的时候包含的话，这些信息一般会存在于“调试器信息”（DWARF 调试表）中，或者从一个由程序压入堆栈的“帧指针”链表获得。当然，代码必须使用帧指针。如果这两种类型的信息都没有，那么这个调用栈无法被重建。

在多线程任务应用中，每个线程都有自己的堆栈，因此调用栈的信息也只和它对应的线程相关。

（2）单步执行

Candidates must understand what single stepping is and how it may be used in debugging. They should understand the difference between Step-In and Step-Over when single-stepping.

考生需要理解什么是单步执行以及如何在调试中应用，它需要理解在单步执行时 Step-In 和 Step-Over 的区别。

单步执行指的是调试时调试器可以控制执行部分代码，每一次执行一条指令。Step-In 和 Step-Over 的区别可以从函数调用中理解。当采用 Step-Over 来调试函数时，整个函数会作为一步来执行，让程序员可以直接执行整个函数而不需要进入函数里面去单步执行。Step-In 表示进入函数里面去单步执行函数体。

（3）开始/停止

Candidates must understand what start/stop debugging is and how it may be used.

考生需要理解什么是开始/停止调试以及如何使用它。

开始的含义就是按下调试器的“开始”按钮，处理器退出调试状态并重新进入正常执行状态（从当前的程序指针处开始执行），直到程序遇到某个让它停止的因素为止，这些因素通常是一个断点、观察点或者捕捉向量事件或者是从外部调试器及其他系统阻塞所产生的调试请求。

开始/停止的调试模式与不能停止代码执行的调试的系统形成鲜明的对照。在一些嵌入式系统中（如汽车引擎控制系统），在进行系统调试时是不能简单地让处理器停止执行的。

（4）打印 printf

Candidates must understand how printf may be used in code instrumentation (they may also be expected to understand what other kinds of instrumentation might typically be used).

考生需要理解如何在代码仪器化中使用 printf（还需要理解其他可用的仪器化函数）。

这是一个最基本的调试技术，通常在所有的处理器架构中使用，用于在代码中插入指令来输出一些数据，例如来显示程序的指令流或者某一时刻一些关键变量的值。

（5）裸机代码和应用代码

Candidates should be aware of the difference in debug options between applications running on OS-based or bare metal system (e.g. Ethernet, GDB server, ICE).

Example: Candidates should know that it is necessary to run a server program on the target system in order to debug an OS-based application.

考生需要了解基于操作系统的应用层软件调试和基于裸机代码的系统调试方法的差别（如以太网、GDBServer、ICE）。

例如，考生需要知道在目标板上调试一个基于操作系统的应用程序必须运行一个调试服务程序。

在 30.4 节介绍了这方面的内容。

（6）RAM/ROM 调试

Candidates should be aware of the limitations of debugging code running in different types of memory (e.g. breakpoints and image load).

Example: Candidates should know that it is not possible to set an unlimited number of breakpoints when executing code in ROM. More advanced candidates would understand why this is the case.