基于 AM335x 的 U-Boot/SPL 的 CCS 调试

秦耀明 TI 通用嵌入式处理器技术支持

摘要

在移植 AM335x 的 U-Boot/SPL（Second Program Loader）的过程中难免遇到问题。利用仿真器在线调试，可快速准确定位问题。本文主要介绍如何用 CCS 与仿真器调试基于 AM335x 的 U-Boot/SPL。

内容

图

1 AM335x 的 Linux 启动过程以及 U-Boot/SPL 调试代码的准备

1.1 AM335x 的 Linux 的启动过程

AM335x 的 Linux 的启动包括 ROM Code，SPL，U-Boot 和 Kernel 四个阶段：

1.1.1 ROM Code

ROM Code 是固化在芯片内部的代码，根据 SYSBOOT 引脚的配置，确定启动模式，初始化相应的外设，读取下一阶段启动文件，即 SPL 文件。

在 AM335x Technical Reference Manual的第 26 章 Initialization 中，对 ROM Code 和启动方式有详细描述。本文以 SD 卡启动为例介绍的调试方法，同样可适用于启动方式包括 Non-XIP memory 以及外设启动。（在 XIP memory 启动方式中，启动代码的内存空间和其他方式差异较大，在此不作介绍。）

1.1.2 SPL 和 U-boot

SPL 和 U-Boot 是启动的两个阶段。U-Boot 的加载和运行所需内存远大于 AM335x 片上 SRAM 所能提供的空间（约 110K bytes），而 ROM Code 代码不可编程，故无法支持各种 DDR 内存，所以将其分为两个阶段：SPL 和 U-Boot。

SPL 由 ROM Code 加载到片上 SRAM 中，负责完成最小系统，包括 ARM core，时钟系统，DDR，外设/存储设备等的初始化，在 DDR 中加载并运行 U-Boot 文件。

U-Boot 会在 SPL 系统配置的基础上，进一步配置包括网口，USB，Nand Flash 等外设或存储设备，并从中读取和加载 Kernel，具体在参考文献中有详细描述。

1.1.3 Kernel

Kernel 是 Linux 启动的最后一个阶段，完成整个 OS 系统的搭建。

1.2 U-Boot/SPL 调试代码的准备

1.2.1 U-Boot/SPL 代码

U-Boot/SPL 的代码在一个代码文件夹中，通过编译宏来分别编译。目前 TI U-Boot/SPL 代码发布主要有两个渠道：

- git://arago-project.org/git/projects/U-Boot-am33x.git

  可以获取最新的代码，包含最新的 bug 修正。

- 通过 TI 官网的EZSDK发布：

  EZSDK 是正式发布的软件包，经过全面测试，性能稳定，U-Boot/SPL 在 board-support 目录中。可以选择 EZSDK 作为开发的基础代码。

1.2.2 U-Boot/SPL 的编译

为了便于用 CCS 进行调试， 在编译上需要注意两点，其一，加入调试信息，就是为了加入 symbol 等信息；其二，去掉编译器的性能优化编译选项，因为优化后的代码执行顺序相对 C 代码可能会有调整，不便于单步调试。

针对这两点，需要修改 Uboot/SPL 的 config.mk 文件：

- 在 CFLAG 和 AFLAG 中加入调试编译选项(-g)，从而加入调试信息：

```
311 ALL_AFLAGS = $(AFLAGS) $(AFLAGS_$(BCURDIR)/$(@F)) $(AFLAGS_$(BCURDIR)) -g

312 ALL_CFLAGS = $(CFLAGS) $(CFLAGS_$(BCURDIR)/$(@F)) $(CFLAGS_$(BCURDIR)) -g
```

- 去掉 CFLAG 中的编译选项， -O2（U-Boot 中默认是-O2）

```
61 HOSTCFLAGS  = -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer
```

编译过程可以参考 U-Boot 编译指南。

1.2.3 可执行文件

SPL 生成的 image 在文件夹 335x/U-Boot-am33x/am335x/spl 中，

- u-boot-spl 作为带有调试信息的 image，由于大约有 500K bytes，所以不能直接下载到片上 SRAM（109K bytes），所以只能用于导入 symbol。
- u-boot-spl.bin 包含有调试信息但没有头信息，调试时可将其通过 JTAG 下载到片上 SRAM 上。

- u-boot-spl.map 是 SPL 的内存映射信息，可以找到各函数的入口地址。
- am335/U-Boot-am33x/MLO 是 SD 卡中所存储的，用于启动的 image。

针对不同的启动方式，ROM Code 对 SPL 是否包含头信息有不同的要求，头信息中主要包含下载到片上 SRAM 的地址以及 image 的大小，具体可以参考这里。因此，以 SD 卡方式启动时，相应的 MLO 带有头信息。而 CCS 无法解读该头信息，所以用 CCS 方式下载 SPL image 时应选用没有头信息的 u-boot-spl。

U-Boot 生成的 image 在文件夹 U-Boot-am33x/am335x 中，具体如下：

- u-boot.img 为第二阶段启动所使用的 image。
- u-boot 包含有调试信息，属于 ELF 格式，是调试时需要的 image。
- u-boot.map 这个文件里面存储了 U-Boot 的内存映射信息，可以找到各函数的入口地址。

2 调试环境的准备

调试环境主要包含 3 个部分，仿真器，集成调试环境和开发板。

2.1 仿真器（Emulator）

目前支持 AM335x 的仿真器的型号比较多，有 XDS560v2, XDS510，XDS100v2, XDS100v3, J-Link 等，XDS560v2 调试性能较好，本文的调试将采用该仿真器。

2.2 集成开发环境（IDE）

支持 AM335x 集成开发环境有 CCS，IAR，这里采用 CCS 。

同时，由于 U-Boot/SPL 是在 Linux 中编译，其路径是 Linux 下的绝对路径，所以 Linux 版本 CCS 可通过 image 中的调试信息找到对应的源码，而 Windows 版本的 CCS 不能识别出其绝对路径，需要通过手动找到源码，但是找到一个文件的源码后，CCS 会根据相对路径找到其他文件。因此，本文调试采用 Linux 版本的 CCS 5.2.0 下载地址。

2.3 开发板

AM335x 的开发板中，BeagleBone，Starter Kit和ICE板上已集成 XDS100v2，直接用 USB 线连接即可使用，而GP EVM和IDK上设计有CTI JTAG接口。

这里选择 GP EVM 和 Spectrum Digital XDS560v2 作为调试硬件平台。

3 CCS 调试 Uboot/SPL 的具体步骤

下面正式开始 CCS 的调试。调试的过程主要分为导入 U-Boot/SPL 工程， CCS 连接 AM335x，代码调试等几个部分。

3.1 导入 CCS 代码

在 CCS 中，Menu File -> Import，选择 Makefile 方式导入，如图 1 所示：

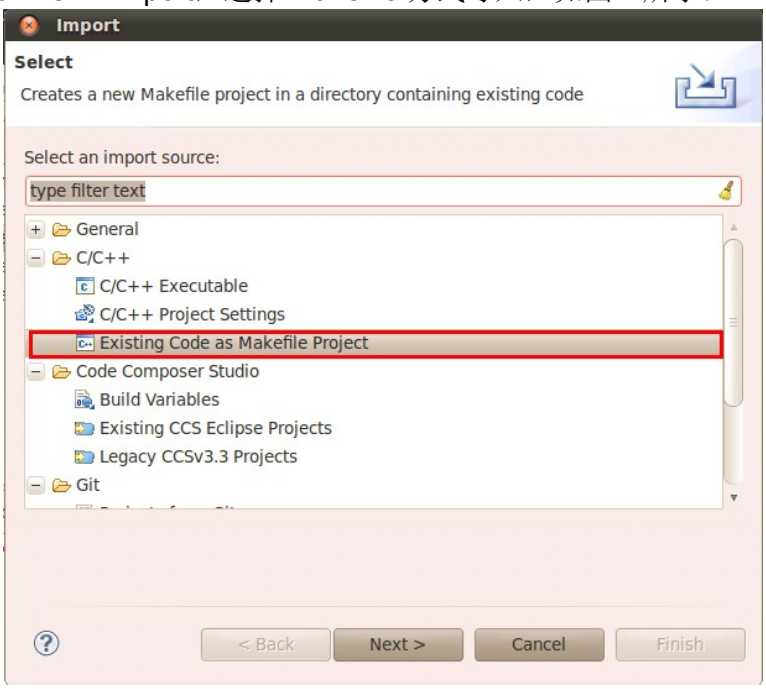

图1 导入工程界面

在 ezsdk 中，U-Boot/SPL 所对应的 Makefile 的具体路径如下：

```
/home/sitara/ti-sdk-am335x-evm-05.05.00.00/board-support/U-Boot-2011.09-psp04.06.00.08
```

可同时导入 U-Boot 和 SPL 代码。(导入代码时会提示"RSTC Support" 提示窗口，点击 NO 即可)

3.2 CCS 连接 AM335x

主要分成仿真器的连接，ARM core 连接两部分：

3.2.1 仿真器的连接

- 硬件连接

  在 GP EVM，将仿真器与其 baseboard 的 J2 口相连。

- 软件连接

  CCS 中，Target 描述仿真器的连接。 首先，配置 Target，包括仿真器和 SOC 配置两部分:

  – View -> Target Configurations 。

  – 点右键选择 New Target Configuration，创建 target。

  – Connection 中选择仿真器型号，在 Board or Device 中选择 AM335x，并保存。

  配置结果如图 2 所示：

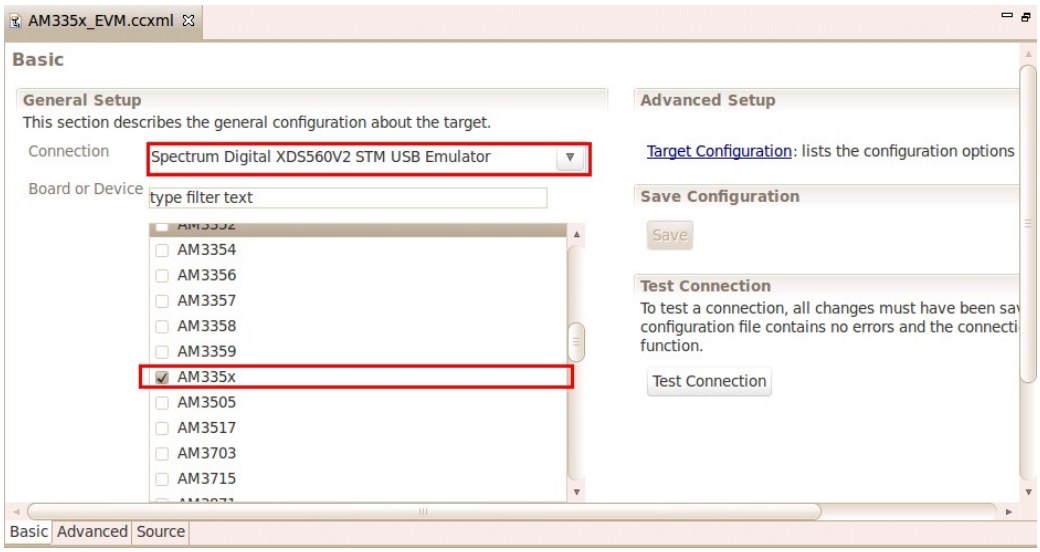

图2 仿真器配置界面

完成配置后，选中 Target Configurations 中已配置好的 target，AM335x_EVM.ccxml， 在右键菜单中选择 Launch Selected Configuration，Launch 成功后如图 3 所示：

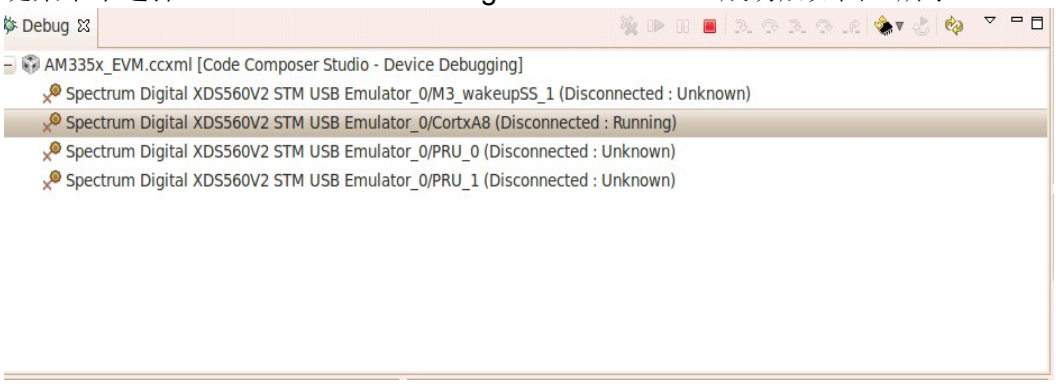

图3 仿真器Launch成功界面

3.2.2 ARM Core 连接

在 Debug 窗口中，右键点击 CortxA8 core，选择 Connect Target。连接成功后，如图 4 所示：

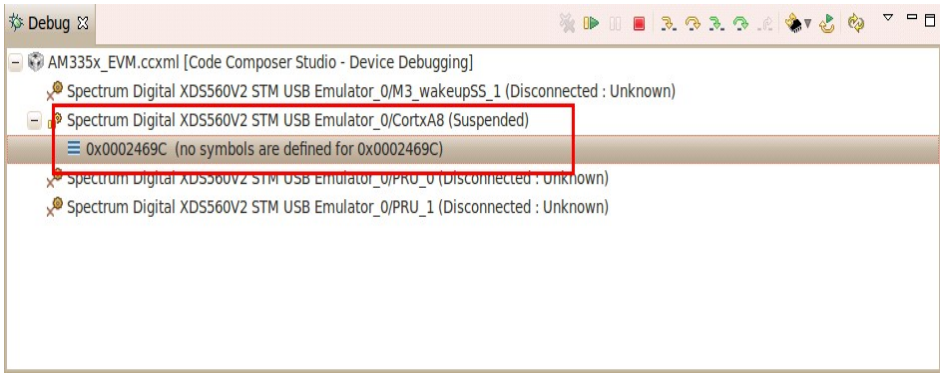

图4 CortxA8 连接成功界面

3.3 SPL 的调试

调试 U-Boot/SPL 的方式有两种：方法一，把 image 通过 JTAG 下载到片上 RAM 或者 DDR 中，然后导入 symbol，重置 PC 指针到 image 的入口处，进行调试；方法二，把 image 烧到 SD 卡或者其他启动存储器上，启动板子，通过 JTAG 停住 CortxA8 的 PC 指针，导入 symbol，重置 PC 指针到 Image 的入口处，进行调试。

下面具体步骤中说明这两种方式如何操作：

3.3.1 下载 SPL image 到 AM335x 中

- AM335x 以 SD 卡方式启动。

  MLO 由 ROM Code 读取到片上 RAM 中。

- SPL image 选择通过 CCS 下载到片上 RAM。

  此时，不要插带有 SPL image 的 SD 卡。鼠标左键选择 CortxA8 core， 然后在 CCS 菜单中， Tools -> Load Memory，选择 u-boot-spl.bin，如图 5 所示：

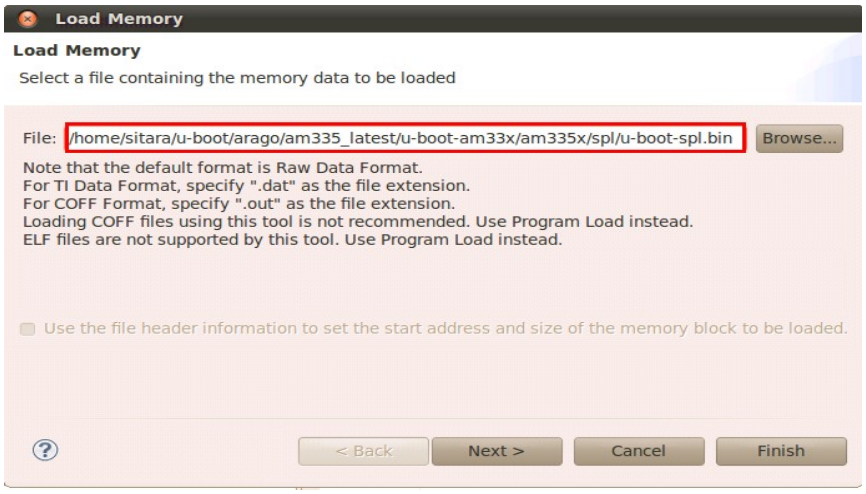

图5 加载 SPL 界面

如前所述，CCS 将下载不包含头信息的 u-boot-spl.bin。因此，需要为 CCS 指定加载的地址，该地址由 ROM code 指定，为 0x402F0400。软件上，在 include/configs/am335x_evm.h 中由宏 CONFIG_SPL_TEXT_BASE 定义。此外，U-Boot/SPL 代码都是运行在 ARM（32bit）模式下，所以 Type-size 也需设为 32bit。设置界面如图 6：

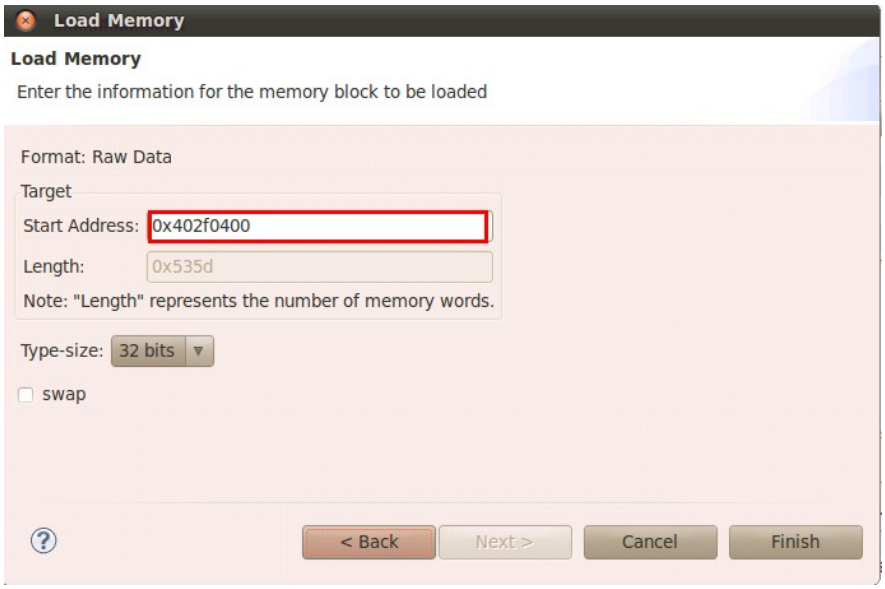

图6 加载SPL界面-2

最后点击 Finish，SPL 就会被 load 到片上 RAM 中了。如果在 Console 窗口中有错误信息，需要把 SPL image 重载一遍。

3.3.2 加载 symbol

加载 RAW image 后，需加载 symbol，建立运行代码和 CCS 中源码的联系。可通过 Run -> Load -> Load Symbols，选择带有 symbol 信息的 U-Boot-spl，加载 symbol 信息，界面如图 7：

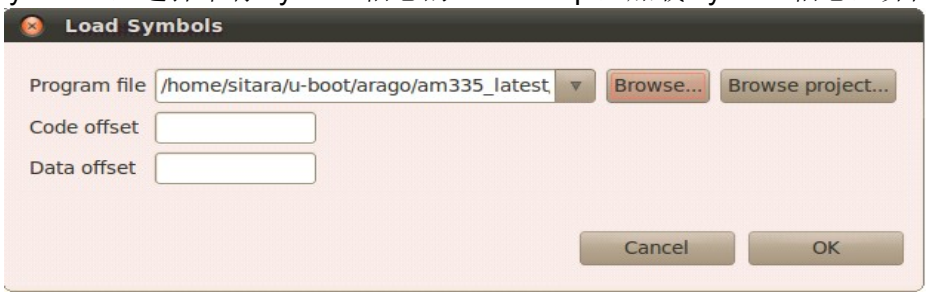

图7 SPL加载symbol界面

3.3.3 设置 CortxA8 core 到 ARM 状态

ARM core 启动后，默认在 Thumb（16bit）模式下，需切换到 ARM（32bit）模式。在 View->Registers 中展开 CPSR 寄存器，把 T 位设置为 0 即可。界面如图 8：

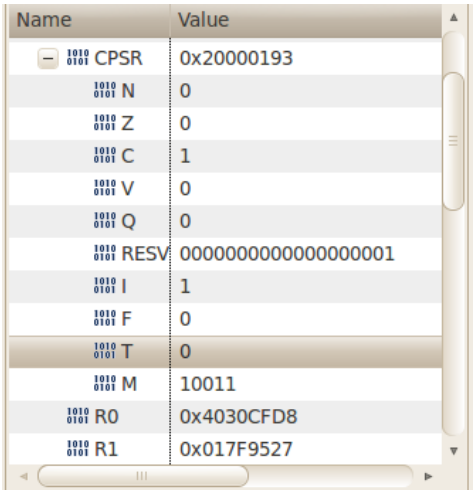

图8 ARM 模式配置界面

3.3.4 SPL 的单步调试

- 设置 PC 到 SPL 的入口处。

  加载完 SPL image 后，PC 指针根据加载方式不同而指向不同地址，需将 PC 指向 SPL 入口处，从而 SPL 在 CortxA8 core 上从开始处运行。入口地址即加载 SPL image 地址，为 0x402f0400，在 View->Registers 中设置 PC 指针的值即可, 界面如图 9：

图9 SPL 起始地址配置

查看反汇编窗口（Disassembly），可得 PC 指针设置完成后的界面如图 10：

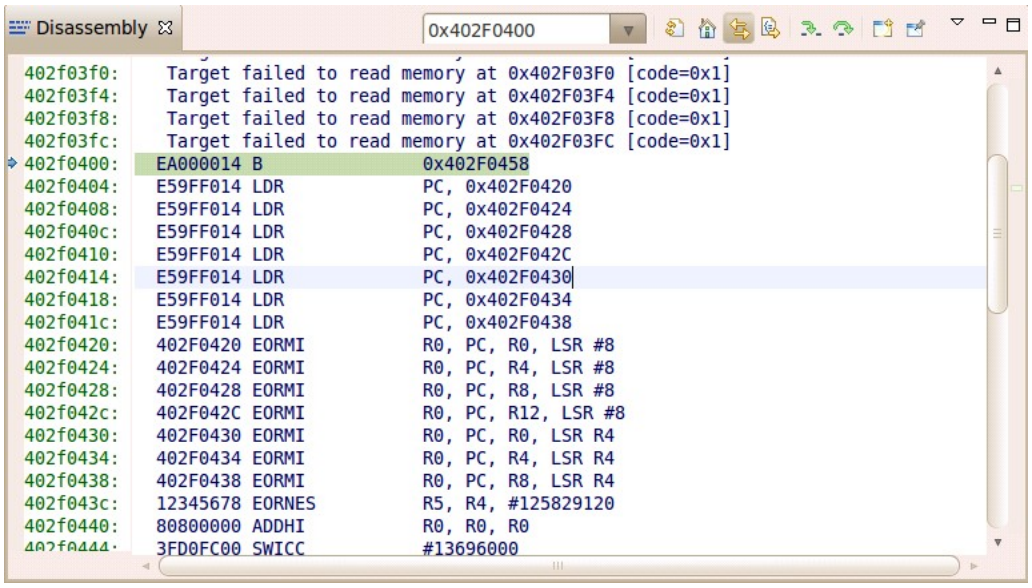

图10 SPL 加载成功反汇编窗口

如上图所示，只有汇编窗口，而编辑窗口侧没有显示源码。这是由于开始执行的代码在 /arch/arm/cpu/armv7/start.s 中， 为汇编代码。

- 单步运行。

  点击 Debug 窗口(view->debug)上 tool bar 中的汇编单步按钮，界面如图 11 所示，就开始调试了。

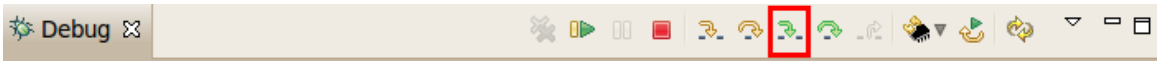

图11 Deubgger工具栏界面

如图 12 所示，点击单步运行按钮，可以看到 PC 指针向前运行。说明已经可以成功进行调试了。

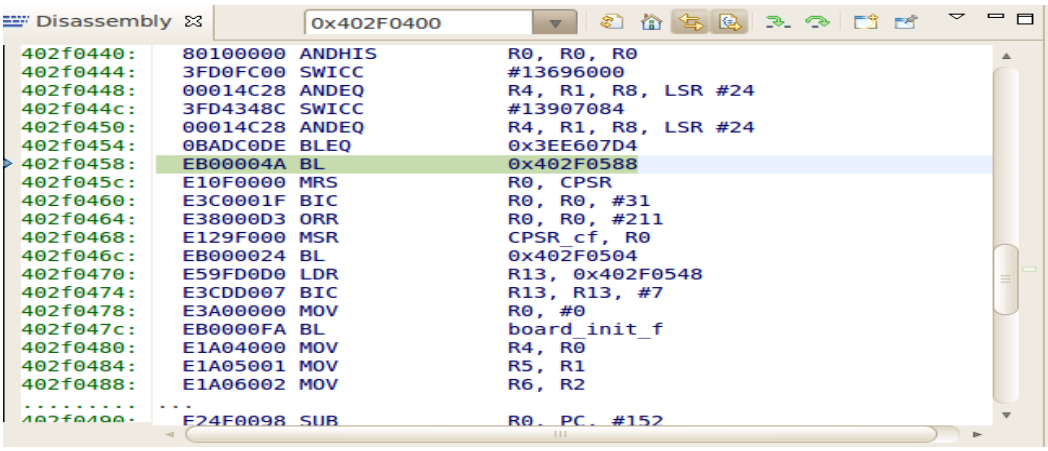

图12 单步调试界面

- 设置断点。

  在汇编代码中设置断点时，不可直接在源文件中设置断点。可根据 map 文件中的映射地址，在汇编窗口中查找该地址，然后设置断点。而在 C 语言中，可直接在源文件中设置断点，以 arch/arm/lib/spl.c 为例，如图 13 所示：

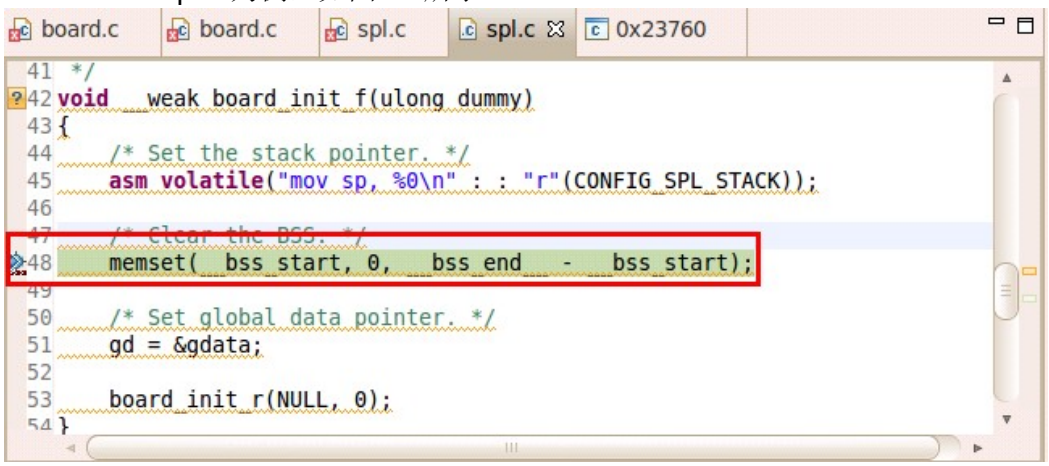

图13 断点设置成功界面

至此已能完整的运用 CCS 对 SPL 进行调试。

3.4 U-Boot 的调试

U-Boot 的调试过程和 SPL 调试过程类似，有以下几点不同：

- 加载 U-Boot 前，需先加载并运行 SPL。

  从 AM335x 的启动过程可知，U-Boot 运行在 DDR 中，故需要先运行 SPL 初始化 DDR 等相关模块。

- 选择 ELF image 进行调试。

  编译 U-Boot 中生成的 image 中，u-boot 为 ELF 格式， 包含加载地址， symbol 等信息。可在 CCS 菜单 Run-> Load-> Load Program 进行加载， CCS 会根据 ELF 中包含的加载地址（0x80800000）加载， 如下图所示：

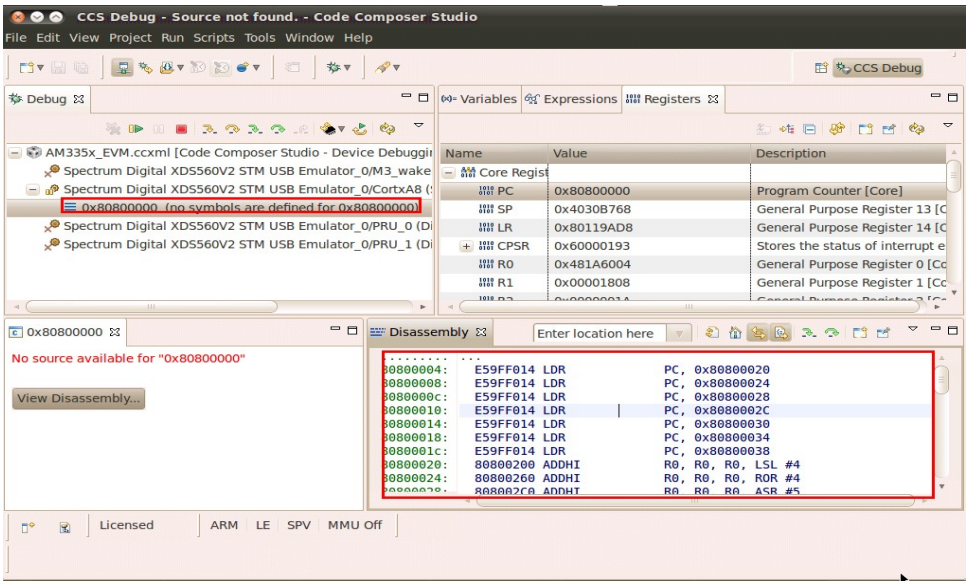

图14 U-Boot加载成功界面

如图 14 所示， PC 直接指到 0x80800000 地址了，也就是 U-Boot 的起始地址。CCS 从 ELF 文件头信息中得到该地址，定义在 CONFIG_SYS_TEXT_BASE（include/configs/am335x_evm.h）中。

在 EZSDK 5.06 之前，该地址定义在 0x80100000，此后，为了支持 NOR flash 等启动，将该地址修改至 0x80800000。

- U-Boot 代码的重载（code relocation）。

  代码重载是由开源 U-Boot 所设计的，相应的，symbol 也需重新载入。 在 board_init_f() 中调用 relocate_code()进行代码重载，重载地址通过 gd->relocaddr 参数传递。因此，可在 arch/arm arch/arm/lib/board.c 中 447 行，relocate_code()函数调用处设置断点，并运行至此，如图 15：

图15 U-Boot 重载调用 bol

查看变量 addr 的值（view->variables）可得重载的地址 0x9FF4E000。在 EZSDK5.06 之前该地址为 0x9FF88000。此地址可作为加载 symbol 的偏移地址。重载 symbol，可在 CCS 的菜单中， Run -> Load -> Add Symbols, 同样选择 ELF 格式 u-boot， data/code offset 即为偏移地址，如下图所示：

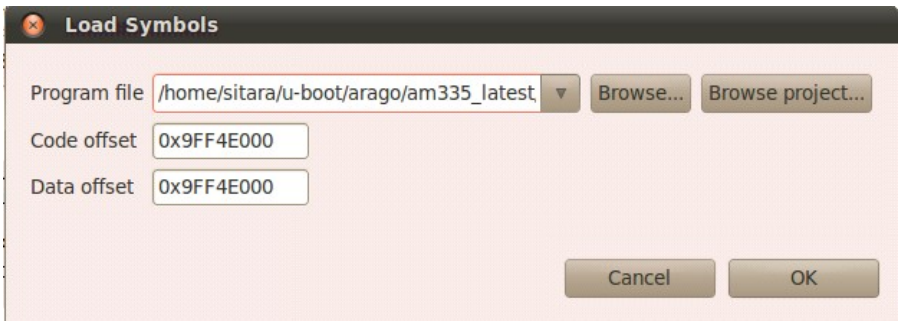

图16 U-Boot 重载 symbol 界面

若需在代码重载后代码中设置断点，需 load Symbols 后，在相应源码中设置断点，这里以在 arch /arm/lib/board.c 中 510 行设置断点为例，如图 17 所示：

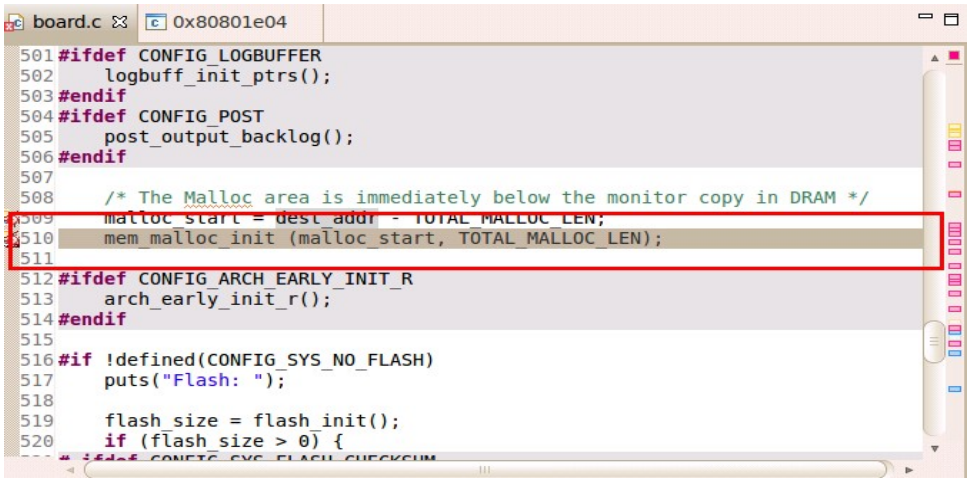

图17 U-Boot 重载 symbol 后设置断点成功界面

4 总结

关于用 CCS 与仿真器对 AM335x 的 U-Boot/SPL 的调试就介绍完了。这里介绍的方法，包含了 CCS 与仿真器调试的基本原则，不仅可用于 U-Boot/SPL 调试，也可用于 Starterware，Kernel 等调试。

参考文献

1. http://processors.wiki.ti.com/index.php/AM335x_U-Boot_User%27s_Guide
2. http://processors.wiki.ti.com/index.php/JTAG_Adapters
3. http://processors.wiki.ti.com/index.php/JTAG_Connectors
4. http://processors.wiki.ti.com/index.php/U-boot_Debug_in_CCSv5

重要声明

德州仪器(TI) 及其下属子公司有权根据 JESD46 最新标准, 对所提供的产品和服务进行更正、修改、增强、改进或其它更改， 并有权根据 JESD48 最新标准中止提供任何产品和服务。客户在下订单前应获取最新的相关信息, 并验证这些信息是否完整且是最新的。所有产品的销售都遵循在订单确认时所提供的TI 销售条款与条件。

TI 保证其所销售的组件的性能符合产品销售时 TI 半导体产品销售条件与条款的适用规范。仅在 TI 保证的范围内，且 TI 认为 有必要时才会使用测试或其它质量控制技术。除非适用法律做出了硬性规定，否则没有必要对每种组件的所有参数进行测试。

TI 对应用帮助或客户产品设计不承担任何义务。客户应对其使用 TI 组件的产品和应用自行负责。为尽量减小与客户产品和应 用相关的风险，客户应提供充分的设计与操作安全措施。

TI 不对任何 TI 专利权、版权、屏蔽作品权或其它与使用了 TI 组件或服务的组合设备、机器或流程相关的 TI 知识产权中授予 的直接或隐含权限作出任何保证或解释。TI 所发布的与第三方产品或服务有关的信息，不能构成从 TI 获得使用这些产品或服 务的许可、授权、或认可。使用此类信息可能需要获得第三方的专利权或其它知识产权方面的许可，或是 TI 的专利权或其它 知识产权方面的许可。

对于 TI 的产品手册或数据表中 TI 信息的重要部分，仅在没有对内容进行任何篡改且带有相关授权、条件、限制和声明的情况 下才允许进行复制。TI 对此类篡改过的文件不承担任何责任或义务。复制第三方的信息可能需要服从额外的限制条件。

在转售 TI 组件或服务时，如果对该组件或服务参数的陈述与 TI 标明的参数相比存在差异或虚假成分，则会失去相关 TI 组件 或服务的所有明示或暗示授权，且这是不正当的、欺诈性商业行为。TI 对任何此类虚假陈述均不承担任何责任或义务。

客户认可并同意，尽管任何应用相关信息或支持仍可能由 TI 提供，但他们将独力负责满足与其产品及在其应用中使用 TI 产品 相关的所有法律、法规和安全相关要求。客户声明并同意，他们具备制定与实施安全措施所需的全部专业技术和知识，可预见 故障的危险后果、监测故障及其后果、降低有可能造成人身伤害的故障的发生机率并采取适当的补救措施。客户将全额赔偿因 在此类安全关键应用中使用任何 TI 组件而对 TI 及其代理造成的任何损失。

在某些场合中，为了推进安全相关应用有可能对 TI 组件进行特别的促销。TI 的目标是利用此类组件帮助客户设计和创立其特 有的可满足适用的功能安全性标准和要求的终端产品解决方案。尽管如此，此类组件仍然服从这些条款。

TI 组件未获得用于 FDA Class III（或类似的生命攸关医疗设备）的授权许可，除非各方授权官员已经达成了专门管控此类使 用的特别协议。

只有那些 TI 特别注明属于军用等级或“增强型塑料”的 TI 组件才是设计或专门用于军事/航空应用或环境的。购买者认可并同 意，对并非指定面向军事或航空航天用途的 TI 组件进行军事或航空航天方面的应用，其风险由客户单独承担，并且由客户独 力负责满足与此类使用相关的所有法律和法规要求。

TI 已明确指定符合 ISO/TS16949 要求的产品，这些产品主要用于汽车。在任何情况下，因使用非指定产品而无法达到 ISO/TS16949 要求，TI不承担任何责任。

	产品		应用
数字音频	www.ti.com.cn/audio	通信与电信	www.ti.com.cn/telecom
放大器和线性器件	www.ti.com.cn/amplifiers	计算机及周边	www.ti.com.cn/computer
数据转换器	www.ti.com.cn/dataconverters	消费电子	www.ti.com/consumer-apps
DLP® 产品	www.dlp.com	能源	www.ti.com/energy
DSP - 数字信号处理器	www.ti.com.cn/dsp	工业应用	www.ti.com.cn/industrial
时钟和计时器	www.ti.com.cn/clockandtimers	医疗电子	www.ti.com.cn/medical
接口	www.ti.com.cn/interface	安防应用	www.ti.com.cn/security
逻辑	www.ti.com.cn/logic	汽车电子	www.ti.com.cn/automotive
电源管理	www.ti.com.cn/power	视频和影像	www.ti.com.cn/video
微控制器 (MCU)	www.ti.com.cn/microcontrollers		
RFID 系统	www.ti.com.cn/rfidsys		
OMAP应用处理器	www.ti.com/omap		
无线连通性	www.ti.com.cn/wirelessconnectivity	德州仪器在线技术支持社区	www.deyisupport.com

邮寄地址： 上海市浦东新区世纪大道 1568 号，中建大厦 32 楼 邮政编码： 200122