Uboot调试

1. 介绍

ARM的bootloader有很多种，目前用的比较多的是开源的Uboot(universial boot）。booloader是处理从系统开机初始化硬件到开始到启动操作系统内核过程的一段程序，是系统最新运行的程序，与硬件有着很大的关系。

bootloader的主要工作是初始化系统的软硬件环境，使之满足系统启动的条件，并加载内核和文件系统的内存，将系统控制权转交给内核。一般来说，uboot可以分成两个阶段：

第一个阶段主要是初始化硬件、关闭看门狗(引导过程中不需要检查）、关闭中断（引导的时候中断禁止）、关闭cache（将第二阶段的image直接写入RAM，不需要cache），关闭MMU(bootloader使用实地址），设置CPU速率和时钟的频率，为stage2 准备RAM空间（大小和堆栈）等;

第二阶段主要是C语言程序，这个阶段主要是初始化硬件、检查系统内存映射、加载内核映像和文件系统映像、为内核启动设置参数，最后将控制权转交给内核入口程序等。

对Uboot进行编译，我们能得到u-boot-spl.bin,u-boot-spl,u-boot-spl.lds, u-boot-spl.map以及u-boot.bin,u-boot,u-boot.img,u-boot.lds,u-boot.map等。

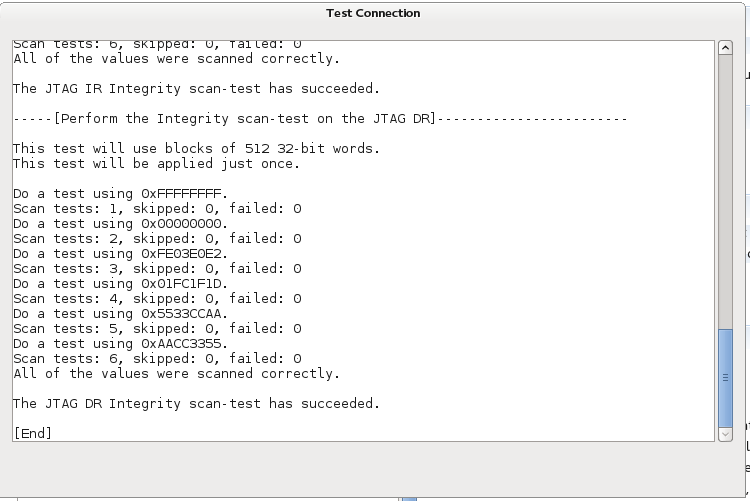
基于上面编译得到的文件，采用TI官方提供的CCS5.5对Uboot进行调试。

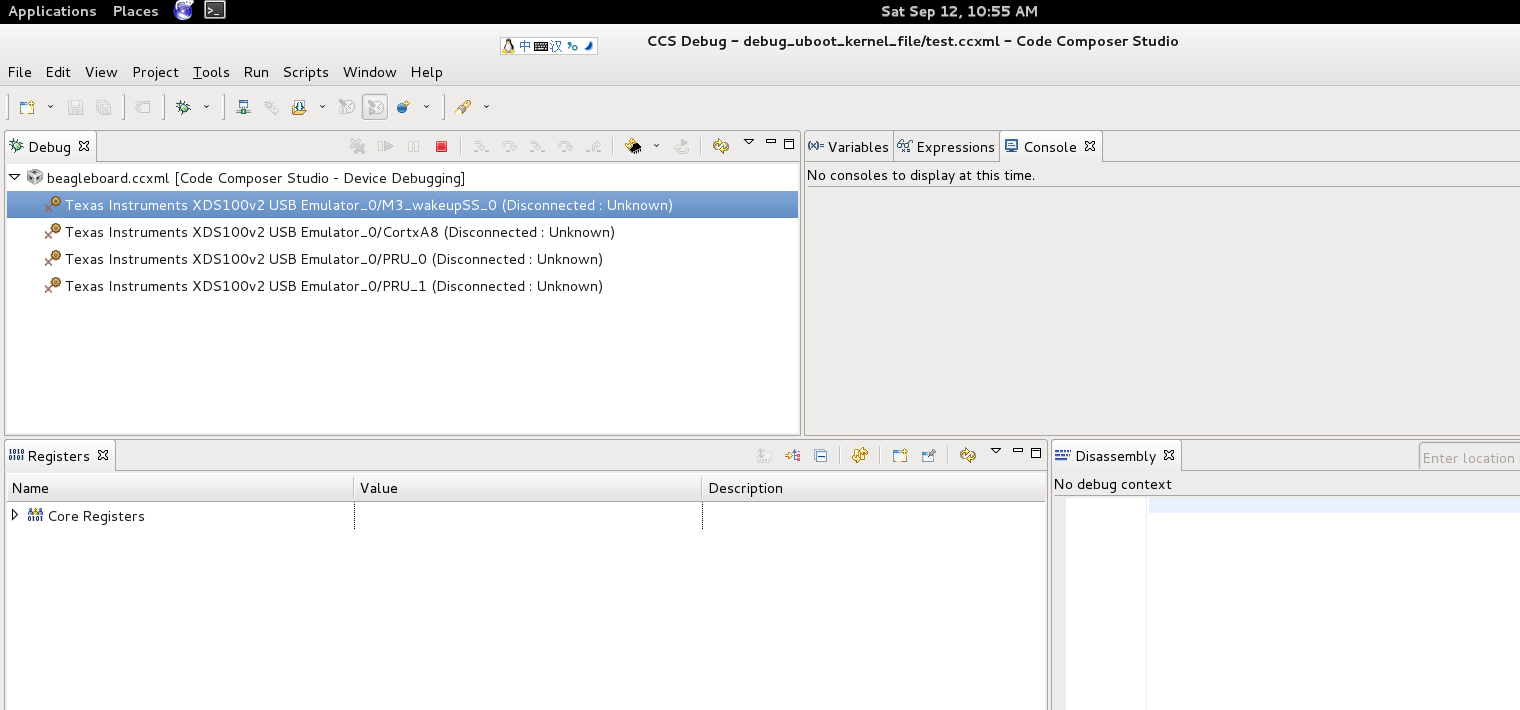
1. 调试前的准备

在开始调试前，准备一些必要的东西：

* 1. beaglebone/beaglebone black开发板;
  2. usb/Jlink连接线;
  3. Linux系统，已经装有TI的CCS;
  4. 已经编译好的Uboot;

接着，将开发板通过USB或者Jlink连接到电脑上，New->Target Configure File 建立一个目标配置文件，仿真器选择Texas Instruments XDS100v2 USB Emulator(选择对应的仿真器），在board or device中选择beaglebone（或者am3359），点击save，并test connect：

当出现上面的提示，表示连接成功。点击View->Target Configrue找到自己的配置，右键点击Launch Selected Configuration进入调试初期：

右键点击CortexA8的connect target，进入调试模式。

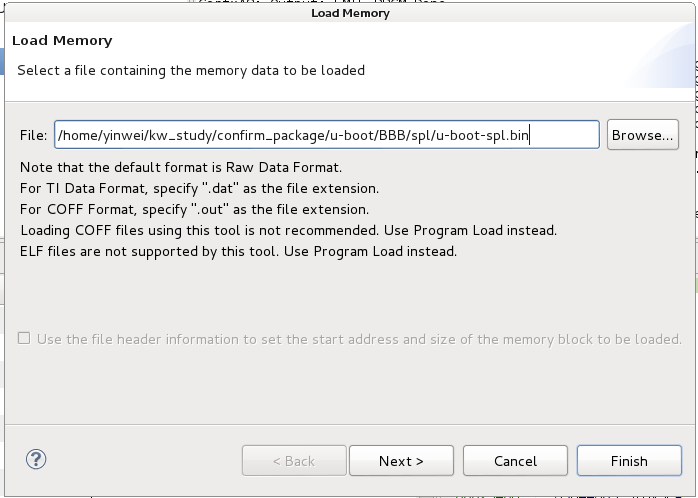
下面分两节讲解怎么进行SPL和Uboot的调试。

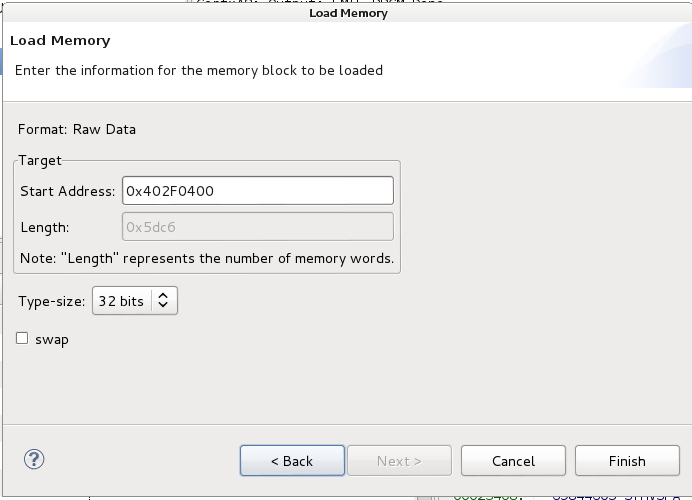
1. SPL调试

SPL是bootloader的第一阶段程序，他的主要作用是初始化硬件环境，并将控制权转给u-boot，从他的代码可以看出他是通过start.S中进行bl board\_init\_r ，再通过检查spl\_image.os来判断u-boot映像是否已经在ram中，然后调用jmp\_to\_image\_no\_args(),从而将控制权交给u-boot。

uboot编译后，在根目录下会生成一个spl目录，里面有u-boot-spl.bin(可执行文件）,u-boot-spl（调试信息）等。

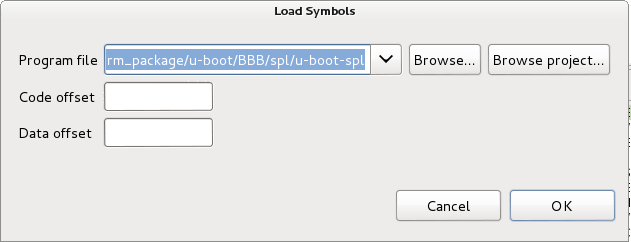
点击tools->Load Memory,得到下面的对话框：

点击Browse输入生成的u-boot-spl.bin文件，next>:



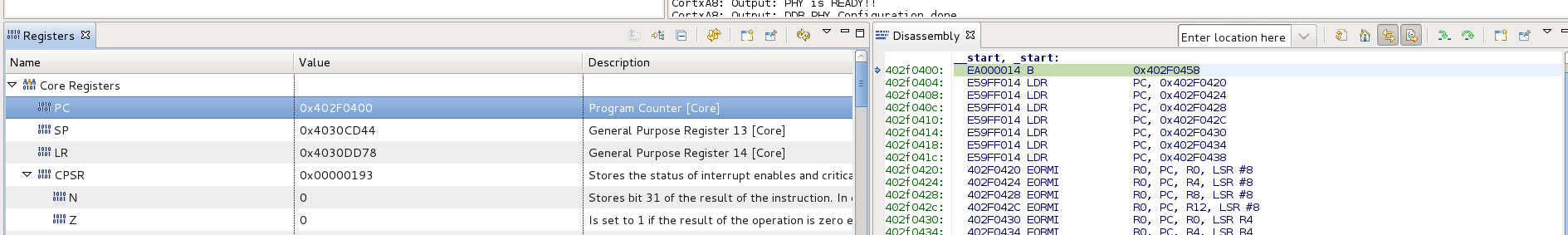
按照上面进行输入，0x402F0400是spl在RAM中的起始地址，将Type size选为32bit（TI am3359是32位的），点击Finish。

接着载入调试symbol，点击Run->Load->Load Symbol：

Browse找到调试信息文件u-boot-spl,code offset 和Data offset 不用输入，点击OK。

修改：

* 1. CPSR状态寄存器的T标志位为0，不是则设置;
  2. 修改PC值为0x402F0400,进入SPL的起始地址;

从上图可以看出，已经停在了SPL的入口地址，从这里开始，就可以进行单步调试。

1. Uboot调试

在进行Uboot调试前，必须将SPL加载运行，因为Uboot要加载到DDR上的话，首先就得对DDR进行初始化才行。

Uboot的调试和SPL的调试相似。Uboot编译后生成的U-boot文件是ELF格式的，自身带有调试信息，因而，在加载执行文件和调试信息的时候可以都Load U-boot就行了。有一点不同的是Uboot的阶段涉及到代码的重载，在加载调试信息的时候，要设置code/data offset,这个offset可以在arch/arm/lib/board.c里的gd->relocaddr得到。

