作者原文的名字叫利用 DS-5 旗舰版中的 ARMv8-A 模型启动 Linux, 我觉得名字起的太长, 蛋疼, 就顺手改了, 原文地址在这, 下面是中文翻译。

ARM 最近推出的 DS-5 旗舰版添加了对最新 ARMv8-A 架构和相关 ARM 处理器的支持。安装包中包含 一个 ARMv8 (FVP) 模型,让你可以在没有 ARMv8 硬件之前就可以进行软件开发。我将在本文中向 你说明让 Linux 在 FVP 上运行的步骤。

首先,如果你是一个 DS-5 的新手, DS-5 中提供了一些裸机例子,你可以使用这些样例来确认一 切配置正确。你应该能从 DS-5 的 File 菜单中用 Import... → General → Existing Projects into Workspace 将 Fireworks 的例子从\\DS-5\examples\Bare-metal_examples_ARMv8.zip 中导 入。这个例子中包含了 Debug 配置,你可以通过 Debug 控制面板在 FVP 上运行这个例子。这是一个 预配置的例子,你可以用它作为第一步来了解 GUI,寄存器视图等等。



如果一切正常,我们可以开始将 Linaro 的 Linux 安装到 FVP 上。最新的 ARMv8-A 的 Linux 分支可 以在这里找到,基本每个月都会有更新。如果你只是想在 FVP 上启动 Linux,你可以使用预编译 的文件。你需要获得内核文件 img.axf 和对应的文件系统文件。下面是 2 个对应的文件系统文件,需要在主机上解压才能使用。

vexpress64-openembedded_minimal-armv8-GCC-<version_and_date>. img. gz 是一个最小的文件 系统(~80MB)

vexpress64-openembedded_lamp-armv8-GCC-<version_and_date>. img. gz 是一个全功能的文件系统(~430MB)

我们可以独立启动,也可以使用 debugger 启动 FVP,步骤几乎一致。对于独立启动,我建议创建一个批处理文件,运行下面的命令行。需要注意到把下面的 Kernel 和文件系统文件位置改成你对应的位置:

<path_to> \ DS-5 \ SW \models\ BIN \ FVP_VE_AEMv8A.exe \

-a $\phi = a$ (path_to) img. axf $\$

- "/ <unzipped_filesystem_image> img" \

--parameter motherboard.mmc.p_mmc_file = \

--parameter motherboard.mmc.card_type =的 eMMC \

--parameter motherboard.smsc_91c111.enabled =true

--parameter motherboard.hostbridge.userNetworking =true

—parameter motherboard. hostbridge. userNetPorts = "5555 = 5555, 8080 = 8080, 22 = 22" 有关这些选项的说明,请参阅最新 FVP 文档。从各种机器测试来看,大约需要 60-120 秒开机的最 小文件系统映像,约 4-7 分钟启动了完整的 LAMP 文件系统。

🚅 Telnet localhost	
root@genericarmv8:~# ls hello.c hello.cc index.html	
root@genericarmv8:~# root@genericarmv8:~# pwd	
/home/root root@genericarmv8:~#	
root@genericarmv8:~# ls hello.c hello.cc index.html	
rootlgenericarmv8:~# rootlgenericarmv8:~#	-

对于从 DS-5 调试器启动,从 Run Debug Configurations...,然后点击左上角的 New launch configuration 按钮。添加配置的名称,然后找到 VE_AEMv8。我发现很容易通过在 Filter 窗口 中输入 VE_AEMv8 找到目标,并且这些工具会自动跳转到它。展开树形目录到 Linux Kernel Debug,然后选择 Debug ARMAEMv8-A_MP_SMP,连接到包含四个 CPU 的模型。

Name: Blog Example
🖇 Connection 🛛 🖟 Files 👫 Debugger 🎡 OS Awareness 🚧 Arguments 🚾 Environment
Select target Select the manufacturer, board, project type and debug operation to use. Currently selected: ARM FVP / VE_AEMv8x4 / Linux Kernel debug / Debug ARMAEMv8-A_MP_SMP
VE_AEMv8
ARM FVP VE_AEMv8x4 Bare Metal Debug Debug ARMAEMv8-A_MP_0 Debug ARMAEMv8-A_MP_1 Debug ARMAEMv8-A_MP_2 Debug ARMAEMv8-A_MP_3 Debug ARMAEMv8-A_MP_SMP
然后,你需要粘贴上面批处理文件中的配置到 模型参数 窗口。

Connections
Linux Kernel Debug SMP Model parameters -a img.axf --parameter motherboard.mmc.p_mmc_file="C:\work\workspaces\AEMv8\Linaro_OpenEm

点击调试按钮,然后选择 Connect Only。点击 Debug 连接调试器,点击 go,让系统和前面一样 启动。你现在也可以用调试器控制系统执行(启动/停止等)。 如果你已经重新编译包含调试信息的内核(预构建映像不包含调试信息),你可以停止目标,现 在使用调试控制面板中的加载功能

🗈 Load File 💿 💌
Load Type Add Symbols File 🔹
Loads additional debug information into the debugger.
>rkspaces\AEMv8\Linaro_1403\vmlinux ▼ File System Workspace
Load Offset
Enable on-demand loading
OK Cancel

或从 CLI 输入: add-symbol-file <path_to>\vmlinux 调试器现在将显示在 FVP 上运行的所有线程的状态:



请注意,以下所有操作需要使用上述全功能的文件系统(或者你已经建立了自己一个相似系统)。

你可以创建一个远程系统查看文件系统,通过打开远程系统窗口中,创建一个(右击)→ New Connection → General → Linux 连接,设置主机名设置为 localhost,并给该连接的任意连接 名称(下图中的 FVP Linux)。 然后单击下一步,然后在配置部分中选择 ssh.files。 单击 Finish(完成)。 然后,你将能够扩展文件系统视图(如果有窗口弹出,设置用户名作为 root)。

New Connection		
Remote Linux System	Connection nation	
Parent profile:		
Host name:	LOCALHOST	-
Connection name:	FVP Linux	
Description:		
Verify host name Configure proxy settings		
?	< <u>B</u> ack <u>N</u> ext > <u>F</u>	inish Cancel
Files Define subsystem informa	ation Properties	
☐ ftp.files ☑ ssh.files	Property Valu	Je
Available Services		
2 Ssh / Sftp File Sen 2 ⊗∳ SSH Connector Se E SSH Settings	rvice	
Description Work with files on remote	systems using the Secure Shell (ssh) pro	itocol.
?	< Back Next >	inish Cancel

📕 Remote Systems 🐹
FVP Linux Sttp Files
Ny Home
A Post
4 品 /
b 🔲 bin
boot
> dev
⊳ 🗀 EFI
b interview i
b in home
> 🧰 lib
b iost+found
þ 🧰 media
þ 🧰 mnt
opt
proc
þ 🧰 run
b in sbin
b is sys
> 👛 tmp
b 🗀 usr
> 🧰 var
Image: Shell Processes
🛃 Ssh Shells
你

从这个视图中,你也可以在 ssh 的终端单击鼠标右键,在 DS-5的 GUI 中启动一个终端窗口,从而 省去了来回跳转到 telnet 窗口。

最后,你也可以使用 DS-5 在 FVP 上调试应用程序。 我在 FVP 上新建的 hello.c,用 gcc -g hello.c进行编译生成 a. out。然后,我启动一个 gdbserver 的会话 "gdbserver: 5555 a. out" (你会发现这个端口是在启动模式时指定的模型参数)。然后我就可以创建一个应用程序调试配置,如下,调试应用。

Jerminals 🕱	
See FVP Linux 🖾	
Last login: Mon Apr 21 10:55:05 2014 root@genericarmv8:~# gcc -g hello.c root@genericarmv8:~# gdbserver :5555 a.out Process a.out created; pid = 1528 Listening on port 5555	*
	Ŧ

-

<u>N</u> ame:	FVP Applica	tion Debug	1		
⊲⊳ Co	onnection	🗟 Files 🕯	🎓 Debugger] 🍩 OS Awareness 🛛 🕬= Arguments 🛛 🌌 Environment 🗋		
Sel	ect target				
Sele	ect the manuf	acturer, bo	ard, project type and debug operation to use. Currently selected:		
Lin	ux Applicatio	n Debug / /	Application Debug / Connections via AArch64 gdbserver / Connect to already	running application	
	Filter platform	ıs			
	▲ Linux App	lication De	bug		
	⊿ Applic	ation Debu	g		
	⊿ Co	nnections	via AArch64 gdbserver		
		Connect t	o already running application		
		Download	and debug application		
		Start gdbs	erver and debug target-resident application		
	⊳ Co	nnections	/ia gdbserver		
DS-	5 Debugger w	vill connect	to an already running gdbserver on the target system.		
C-1					
	nnections				
		Address:	localhost		
gdł	oserver (TCP)	Port:	5555		
			V Use Extended Mode 🛛 🛛 Terminate gdbserver on disconnect		
			-		
				Apply	
🌞 Del	oug Control 🛛	3			
		Ē	💐 💥 💥 🦓 🔍 ▼ 🖓 ▼ 🕨 💷 🏊 👁 📌 📬 🔍 🕗	9- ~	
🗖 🖉	FVP Applica	tion Debu	g connected		
۵	🗁 Active T	hreads			
	🔺 🍓 Thre	ad 1528 #1	L stopped on breakpoint #1		
	= m	nain			
		:0x0000007	FB1EB9D20		
Þ	🗁 All Threa	ius			
- 📜	FVP Applica	tion Debug	g connected		
	No OS Supp	oort			

另外,如果应用程序是在 PC 上生成的,您可以使用"下载和调试"选项,用调试器通过远程系统的连接将应用程序下载到 FVP,。

Connection	🗟 Files 者	🌾 Debugger	OS Awareness	(X)= Arguments	Kan	
Select target						
elect the manuf	acturer bo	ard project t	ne and debug on	eration to use. Cu	rently selected:	
Linux Applicatio	n Debug / /	Application D	ebug / Connection	ns via AArch64 gd	bserver / Download and debug app	plication
					5 11	
Filter platforn	ns					
⊿ Co	nnections	/ia AArch64 g	dbserver			-
	Connect t	o already run	ning application			
	Download	and debug a	pplication			
	Start gdbs	erver and deb	oug target-resident	application		
⊳ Co	nnections	via gdbserver				
Applic	ation Debu	g with Rewin	d Support			
LOGIC PD						
⊳ LSI						
b Marvell						-
S-5 Debugger v	vill downloa	ad your applie	ation to the target	system and then	start a new gdbserver session to de	ebug the application. This configuration
equires ssh and	gdbserver (on the target	platform.	·	5	5 11 5
Connections						
	FVP Linux					
RSE connection		[
RSE connection						V Use KSE Ho
RSE connection	Address:	localhost				
RSE connection gdbserver (TCP)	Address: Port:	localhost 5555				
RSE connection	Address: Port:	5555 Use Extend	ded Mode			