H-JTAG 使用说明



H-JTAG twentyone http://www.hjtag.com/



Team MCUzone <u>http://www.mcuzone.com</u>

版本: Rev 1.0 更新日期: 2006.10.28 更新记录

Rev 1.0:

文档创建 2006-10-28 目录

- 第1章 介绍
 - 1.1 H-JTAG 介绍
 - 1.2 H-JTAG 安装
- 第2章 调试
 - 2.1 在 ADS1.2 中使用 H-JTAG 调试
 - 2.2 在 Realview2.2 中使用 H-JTAG 调试
 - 2.3 在 Keil 中使用 H-JTAG 调试
 - 2.4 在 IAR 中使用 H-JTAG 调试
- 第3章 编程
 - 3.1 AT91SAM7S64
 - 3.2 S3C44B0 公版
 - 3.3 LPC2132 测试板
- 附录A Wiggler 电路图
- 附录 B SDT JTAG 电路图

第1章 介绍

当前 ARM 的学习与开发非常流行,由于 ARM 的软件开发相对以前单片机而言更加复杂,硬件上的考虑也比较多,因此选择一个好的调试方法将可以使得开发的除错过程变得更加直接和简单。

现在市面上有很多可用于 ARM 调试的仿真器出售,然而其价格往往都比较贵。这些仿 真器一边都有其专用的软件和硬件,在速度和 flash 编程等方面有各自的优势。然而对初学 者而言,这些仿真器的成本都太高。而简易仿真器的出现,使得大家可以使用甚至自制 ARM 仿真器硬件。

有了调试器的硬件,还要加上调试代理软件,作为中介,将调试器前端软件(比如 AXD)的调试信息与目标板上的目标芯片交互,才能最终完成仿真的任务。目前,可以免费使用的简易 ARM 仿真器的代理软件很多,差别也比较大,主要表现在易用程度,目标器件支持,调试速度等方面。H-JTAG 作为近来新推出的简易 ARM 仿真器调试代理,其支持器件比较多,支持的调试器前端软件也比较多,特别是支持 keil,其调试速度也很有优势。

1.1 H-JTAG 介绍

H-JTAG 是由 twentyone 推出的一款免费调试代理软件。官方主页为:

http://www.hjtag.com/

目前的版本为 0.4.2 (2006 年 12 月 05 日), 支持下列特性:

- 1. 支持 RDI 1.5.0 与 1.5.1;
- 2. 支持 ARM7 与 ARM9 (包括 ARM9E-S 与 ARM9EJ-S);
- 3. 支持 thumb 与 arm 指令集;
- 4. 支持 little-endian 与 big-endian;
- 5. 支持 semihosting;
- 6. 支持 wiggler, sdt-jtag 以及用户自定义的简易调试器硬件接口;
- 7. 支持 WINDOWS 9.X/NT/2000/XP;
- 8. 支持 flash 器件的编程

1.2 H-JTAG 安装

根据上面给出的链接,通过 download 页面下载 H-JTAG 压缩文件包,展开即可获得 H-JTAG 的安装文件,如下图:



双击右侧安装程序即可开始安装。安装完成,点击 H-JTAG.exe 即可开始运行 H-JTAG。



程序运行的主界面如下:



上图是未找到目标器件的画面。快捷栏中的工具图标与菜单条中具有相同图标的菜单功能一 致,可以对照下面的菜单详细介绍:



退出程序:停止 H-JTAG 的运行。点击右上角的关闭按钮──只会使 H-JTAG 最小化到系统

托盘,而不会停止其运行。 2) Operations 菜单 ^{Operations} Flasher S ↓ Reset Target <u>g</u>位目标芯片 ○ Detect Target 探测目标芯片 × Kill Connection 斯开连接 复位目标芯片:可以使得目标器件复位

探测目标芯片:可以使得 H-JTAG 开始探测目标芯片;默认情况下,H-JTAG 在开始运行的时候会自动探测目标器件。如果探测不到目标芯片,会弹出下列对话框:

H-JTAG	Server
!	Unable to find target. Please make sure that the hardware is properly connected and powered up.

此时请检查硬件连接。

断开连接:在有调试前端软件(axd等)连接的情况下,可以断开与调试器前端的连接。 3) Flasher 菜单



运行 H-Flasher: 运行 H-Flasher 软件。该软件的具体功能将在第3章介绍。

自动下载:选中该功能后,调试前端软件在需要的时候可以自动下载程序到 flash。适用于 在 flash 中调试软件。这样在调试的时候就会自动下载代码到 flash,可以省掉一步手动下载 的过程。

4) Script 菜单



初始化脚本: 该功能类似 axd 提供的初始化脚本功能。可以在调试器开始运行的时候自动完成一些对器件的初始化操作。比如禁止 watchdog, 初始化 SDRAM 等。点击该功能将弹出如下窗口:

Ŀ	nit :	Script					
Г					(
	Idx	Cmd	Width	Address	Value		
							•
							•
							- -
							•
L						_	
	🔳 Er	nable Auto In	it	New L	.oad OK	С	ancel

点击 🗧 将会添加一条新的条目:

ldx	Cmd	Width	Address	Value	
1					
는 그는	1 14 4 1				

点击 cmd 将会添加一个具体的命令:



目前支持的只有设置内存(Setmem)和禁止 MMU(DisMMU)。 Width 窗口可以选择所访问内存的宽度:



应该按照实际内存访问的宽度来访问。

Address 可以设置欲进行内存的操作的地址。

Value 就是往内存地址写的值。

填写完成的一个条目如下:

Idx	Cmd	Width	Address	Value	
1	Setmem	32-Bit	0x80000000	0x01	
2					

点击 ቀ 可以再添加一条, 点击 🔶 可以删除一个条目, 点击 📤 与 🖊 可以调整条目间的顺

序。

下方的菜单与脚本的保存和运行有关:

🔲 Enable Auto Init	New	Load	OK	Cancel
--------------------	-----	------	----	--------

勾选了 Enable Auto Init,将会使得 H-JTAG 在运行时自动完成初始化操作。

点击 New 将清除所有的条目,Load 可以载入一个已经定义好的初始化文件。点击 OK 即可 保存但前的设置到一个初始化文件,方便下次加载。点击 Cancel 将不保存改变。 自动初始化:勾选了该功能等同于勾选了初始化脚本中的 Enable Auto Init。

5) Settings 菜单



JTAG 设置:此菜单可以对 JTAG 的硬件进行设置,包括所使用简易仿真器的类型,目前支持 wiggler,sdt 与自定义的 JTAG 连接。如果选择 JTAG 的硬件为自定义,则需要在右侧选择 JTAG 信号线对应的并口线。注意两个复位信号线 nTRST 与 nSRST 是一直可以自定义的。 必须根据硬件选择。Reset Signal Output 里可以选择两个复位信号的极性。下图没有选择 nSRST 信号,因此左侧复位信号取反中 nSRST 是禁止的。

Jtag Selection	Wiggler Pi	n Assi	gnment-	
Wiggler (Predefined)	TMS [Pin3	D1	¥.
Sdt Jtag (Predefined)	тск [Pin4	D2	w.
User Defined	TDI 🛛	Pin5	D3	v
Reset Signal Output	TDO [Pin11	Busy	Ψ.
Image: Image: Market with the second sec	nTRST	Pin2	DO	•
nSRST output inverted	nSRST	NO S'	rs RST	-

Wiggler 与 SDT JTAG 的硬件电路图可以参考附录 A 与附录 B。

端口设置:可以设置 PC 的并口,如下:



应根据 PC 的并口设置来选择,选择完成后可以运行 Port Testing,如果弹出如下错误:

Port	lesting	×
⚠	Selected port doesn't work!	

则应该检查端口设置。

目标器件设置:此菜单可以选择目标器件的类型与 endian 的类型。处理器类型一般选择自动探测即可,如果实在探测不出,也可以试试手动指定。目标器件的对齐模式可以选择小端对齐或者大端对齐。选择完成点击 OK 之后,H-JTAG 即会开始探测目标器件。

æ	cessor Auto	Variant Detect	
•	ARM	(7TDMI	-
arg	get En	idian —	
Targ ©	get En Little	idian Endian	

目标器件管理:此菜单可以管理目标器件。

Add New	ID Code	
ID Code	j0x	- 4
Processor		
Delete Ex	isting ID Code-	
ID Code		- ×

分为添加新的 ID 号以及删除已存在的 ID 号。如果目标器件的 ID 不在目前的处理器列表里, H-JTAG 将不能识别其处理器类型。使用该功能即可添加其 ID,并为其指定一个处理器类型, 那么下次 H-JTAG 再次探测到该 ID 时,就会指出其处理器类型并进行相应的处理。删除与 此过程相反。

TAP 配置:此功能用来配置目标的 TAP。

TDI >>	BYPASS	>>	ARI	vi >:	BYP	4SS	>> TDO
AP Num:	0 ÷		1	-	0	•	
Reg Len:	0 ÷	IZ	Auto	-	0	•	

一般不用设置,但对于 TAP 特殊的目标器件必须手动设置。比如 STR912 就必须按照下图 设置,才能正确使用:



6) Options 菜单



禁止 Semihosting: 禁止使用 semihostiong 功能。

禁止向量中断捕捉:一些调试软件会为异常向量(swi, data abort 等)保留断点,这会占用系 统提供的断点资源。而 ARM7 系列只能提供两个硬件断点。如果勾选这个功能,在 flash 中 调试的时候将不会有余下的断点资源用于单步等。因此一般应该勾选该选项。

自动 reset: 勾选此功能将使得调试软件在必要时可以直接复位目标芯片。

报告数据异常:勾选此功能后,当处理器在调试过程中产生数据异常,H-JTAG 将会报告异常。

7) Help 菜单



H-JTAG 主页:点击后即可打开 H-JTAG 主页。 关于 H-JTAG:将会显示版本信息及作者的联系方式,如下图:



通过里面的链接可以方便的连接到 H-JTAG 主页,以及向作者反馈信息。

第2章 调试

H-JTAG 的主要功能是作为调试代理,使得调试器前端软件可以通过其与简易的 JTAG 通信,完成目标调试的任务。

由于支持 RDI 接口, H-JTAG 支持很多市面上流行的调试软件。下面将分别介绍在流行 IDE 下如何使用 H-JTAG 进行调试,硬件平台使用 mcuzone 的 S64-DEK 2.0 和 wiggler。

2.1 在 ADS1.2 中使用 H-JTAG 调试

使用 wiggler 连接 PC 和目标板,连接好开发板的 USB 线到 PC,运行 H-JTAG,如果连接正确,H-JTAG 将发现 ARM7TDMI 核心(上面步骤以下简称为:正确连接硬件),如下图:

H H-JTAG Server	
File Operations Flasher Script Setting	gs Options Help
₩ ٩ × F 5 5 0 J	0 🔁 🖸
▲ Market A Mar	、 对话框。
如果在奋性尖望中显示为 UNKNOW, 或者出现如下 H-JIAG Server Unable to find target. Please mak is properly connected and powered 	N 山住: E sure that the hardware up.

即说明 H-JTAG 未能与目标器件建立连接,应该检测硬件连接,直到出现正确的连接画面。

运行 AXD,首先要添加 H-JTAG 为新的 target。点击 Optionsà Config Target,如下图



点击 Add

ARM ADI ARMUL	1 1	D:_Dsetup\\gateway.dll D:_Dsetup\\armulate.dll	1.0.0.7 1.4.0.206	Remove
Banyan-UE Multi-ICE	1	D:_Dsetup\ARM\banyan\Banyan.dll D:_Dsetup\\Multi-ICE.dll	1.8.7 2.2.0.1095	
				Configur
	and a second	a target environment from the above	list or odd a	

添加 H-JTAG 安装目录下的 H-JTAG.dll,

打开		?
查找范围(I)	: 🔁 H-JTAG 🗾 🗲	🔁 📸 📰 •
🗀 FC onfig		
FDevice		
HContig		
H-JTAG. dl	0	
	描述: H-JTAG	
又1千名(图):	文件版本: 0.4.0.0	打开(0)
文件类型(T)	创建日期: 2006-10-16 summer最爱tdy21:51 大小: 112 KB	取消

点击 "打开", H-JTAG 即会被添加为一个新的 target, 效果如图:

	Juneitus			0.2
Target	RDI	File	Version	Add
ARM ADI ARMUL Banyan-UE	1 1 1	D:_Dsetup\\gateway.dll D:_Dsetup\\armulate.dll D:_Dsetup\ARM\banyan\Banyan.dll	1.0.0.7 1.4.0.206 1.8.7	Remove
H-JTAG	1	D:_Dsetup\ARM\H-JTAG\H-JTAG.dll	VO. 4. 0	Ronomo
Multi-ICE RealMonitor	1 1	D:_Dsetup\\Multi-ICE.dll D:_Dsetup\\RealMonitor.dll	2. 2. 0. 1095 1. 0. 1. 1341	<u>Save</u> As

H-JTAG Debug Interface for ARM In-Circuit Emulation.

点击 OK 即可使用 H-JTAG 作为调试代理。如果出现如下对话框:

Restart
Configure
Co <u>n</u> nect mode
Quit

说明连接有问题。

当 AXD 连接上 H-JTAG 之后, H-JTAG 下方的状态栏会有详细的显示,包括调试器前端软件的名称,RDI 版本等,如下图:



此时 AXD 也处于就绪状态,通过点击 Fileà Load Image...载入欲调试的 axf 文件即可



则说明断点设置过多,原因在于 ARM7 系列只有两个硬件断点,而在 flash 中调试时只能使用硬件断点,因此断点数量有限制。一般而言,系统将会保留一个硬件断点供单步调试时使用,因此实际所能使用的断点仅剩一个。需要注意的是,默认情况下,AXD 会为异常向量保留断点,这将造成在实际调试的时候没有任何断点可用,解决方法有两个:

一是在 AXD 中点击 Options à Config Processor...,点击 Clear All 去掉所有。

Processor Properties- ARE	TDEL ? 🔀
Vector catch R U S P D I F Clear All Set All	OK Cancel
🔲 Enable Comms Channel view:	Help
🖵 Semihosting	
C Std semihosting Vector	
C DCC semihostingHandler	
Semihosting SWIs	
ARM semihosting	
Thumb semihosting	

二是在 H-JATG 中直接禁止所有的向量捕捉,如下图:



勾选 Disable Vector Catch 即可。

2.2 在 Realview2.2 中使用 H-JTAG 调试

正确连接硬件后,运行 RVD,点击 Target Connect to Target...,在弹出的连接控制窗口 中点击右键,如下图:

	[]]]]]]]]]]]]]]]]]]]		
+ Server + loca	Collapse All Expand All Connection Properties Add/Remove/Edit Devices Select Board File	÷s et simulator nost col (serial port)	

点击 Add/Remove/Edit Devices...即会弹出下列窗口:

 \mathbf{X}

🖪 RDI Target List

Name	Version	Description	
ਤਿ⊊ੇ Remote_A ਤੋਂ ਜ਼ੀ ARMulator	v1.2 v1.4	Angel debug protocol (serial port) ARM instruction set simulator	
	C		

选择 Add Dll..., 选择添加 H-JTAG.dll,

A Select RD	I DLL			
查找范围(I):	🗀 h-jtag		• 🗢 🗄) 💣 🎹 -
しています。 我最近的文档 夏雨	FConfig FDevice HConfig Target Temp			
2000 100 100 100 100 100 100 100 100 100	Contraction of the	大小: 160 KB 文件: TargetDrv.dll,	TargetInfo	

为 H-JTAG 设定名字即描述,比如直接叫做 H-JTAG。

Enter a name and a connection list:	description for the new entry in the
Short Name (examp	le - "Dual 7TDMI"):
HJTAG	
Description (example	e - "Multi-ICE with two ARM7s"):
Description (example	e - "Multi-ICE with two ARM7s"):
Description (example HJTAG	e - "Multi-ICE with two ARM7s"):

点击 OK 之后,新的 RDI 目标即被添加到 RVD 中,如下图:

<u>K</u> elp					
Name	Description				
ARM-A-RR + %H-JTAG + %ARMulator ARM-A-RR + %Remote_A %Server + %localhost	ARM Ltd. RDI targets H-JTAG ARM instruction set simulator ARM Ltd. RDI targets Angel debug protocol (serial port) Connection Broker Simulator Broker				

点开 H-JTAG 左侧的"+"号,勾选下方的 ARM,即可与 H-JTAG 建立连接。

😫 Connection Cor	ntrol (summer\rvdebug.brd)	
Help		
Name - ARM-A-RR - CH-JTAG ARM + CARMulator - ARM-A-RR + CRemote_A - Server + Clocalhost	Description ARM Ltd. RDI targets H-JTAG ARM on localhost ARM instruction set simulator ARM Ltd. RDI targets Angel debug protocol (serial port Connection Broker Simulator Broker)
	ARM7TDMI 0×3F0F0F0F	
Ready	TVS RDI 1	. 5. 0x0000000

连接完成后的 H-JTAG 的状态栏将显示连接状态。

点击 Targetà Load Image...,或者直接在下图中点击,即可加载欲调试的 image 文件。



加载调试文件后 RVD 的窗口如下:



可以在源代码中设置断点并运行。可以查看变量,内存,寄存器值。

REPORTORIAN	95> = MARELARE-A	-KR (Osattac	hed]					
Lile Idit Fice T	arget Braject Build	Dabug Icols Hel	ly Ly6L-m + L √	C (R IF 00 00 0		a di x		
Tile sain.c	Fiad	17 0 -1135	Eine	-				
and to have for the						4	크쾨	
<pre>(mail and any (vers) (mail and (vers) if it is a start (vers) f seath Main(vers) seath M</pre>	-500000/ -500000/ -0007777400+0x07 -0007777400+0x07 -0007777400-0x07 -0007977400-0x07 -0007977400-0x07 -0007977400-0x07 -00079777400-0x07 -00079777400-0x07 -00079777400-0x07 -00079777400-0x07 -00079777400-0x07 -00079777400-0x07 -00079777400-0x07 -00079777400-0x07 -00070000/ -0007777400-0x07 -00070000/ -000777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -0007777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -00077777400-0x07 -0007777400-0x07 -0007777400-0x07 -000777777 -00077777777777777777777	15 0 0 19 19 19 19 19 19 19 19 19 19 19 19 19					Type # AFM - Thome - Victor - Source - Source	Value FC-00x001001A8 use_ANAS.act Hange+Symbols cluttos a. cRot. Sarado ts from Jmage
+ + Dom / Sto Ana	inic (sembloots /	(+		100	+ + Process	(Map/
Name Void delay: void delay: void Hein/vo void Hein/v	Value Va	A Bass	Value 400097 cBcb	Clicktinos Clicktinos	(s 1)	× 90 0007 × 0007 × 00000 × 0000 × 0000 × 00000 × 0000 × 00000	A13D E1 00 0004 B3 00 0001 R5 00 0000 R7 00 0000 R1 0000 R1 00 0000 R1 00 0000 R1 00 0000 R1 00 0000 R1 00	00000002 0000000 0000000 0000000 0000000
×				all new particular		Laurine Con	C	
D atep								0
Stop>	o (bus (netos /se	OM /Log /						110
A start of the sta	V. V. W. Ward	and and a						

如果在设置断点的时候出现:

a2

S64Init();	Error BOO18 (Board): H/W Breakpoint limit	Regi	ster			
* (small mad int	reached					
* (unsigned int		RO	00000000	Rl	00000000	
* (unsigned int		R2	00000004	R3	00000002	
* (unsigned int		R4	00000001	R5	00000000	
- fonorgied mo	OK	R6	00000000	R7	00000000	
while (1)	Landau and L	R8	00000000	R9	00000000	
f (1)		R10	00000000	R11	00000000	
*(unsime)	tht *)0xFFFFF434=0x01:	R12	00040000	SP	00203F50	
delav():	,	LR	001001E8	PC	001001E8	
*(unsigner	int *)0xFFFFF434=0x02;	CPSR	600000DF			
delay();		NZCV	FIQ IRQ S	TATE M	IODE	
* (unsigned	1 int *)0xFFFFF434=0x04;	0170	TTA DIA M	ow, s	.ve	
delay();		4 1	Core Deb	ug /		
*(unsigned	<pre>int *)0xFFFFF430=0x07;</pre>	-				
delay();						

即说明硬件断点数量已用尽,应清除一些不用的断点后再试。

2.3 在 Keil 中使用 H-JTAG 调试

在 keil 中配置调试器使用 RDI 接口,如下图:

)evice Target	t Output Listing C/C++	Asm Linke	r Debug	Vtilities		
♥ Use Simulator Settings ■ Limit Speed to Real-Time		Settings 🧖 🛽	[se: RDI VLIN	Interface Driver 💌 K ARM Debugger	▼ Settings	
🔽 Load Appli Initializatio	cation at Sta 🛛 💆 Run to m	main() 🔽 I Ini	ULIN Load RDI tializati	K Cortex-M3 Debugger Interface Driver on	main()	
		Edit	emapROMst	art. ini 👘	Edit	
 ✓ Breakpo ✓ Watchpo ✓ Memory 	ints 🔽 Toolbox ints & P. Display	5	✔ Breakpo ✔ Watchpo ✔ Memory	pints 🔽 Toolbox pints Display		
CPU DLL:	Parameter:	Driv	ver DLL:	Parameter:		
SARM. DLL	ARM. DLL -cAT91SAMTS		SARM. DLL			
Dialog DLL:	Parameter:	Dial	Log DLL:	Parameter:		
DARMATS DLL -p91SAM7S64		TAR	TARMATS.DLL -p91SAM7S64			

选择完成后,点击右侧的 Settings,

RDI Interface Driver Setup		
Browse for RDI Driver DLL		
D:\Dsetup\H-JTAG\H-JTAG.dll		
Debug Cache Options Cache <u>C</u> ode Cache <u>M</u> emory	Configure EDI Driver	
	OK Cancel	<u>H</u> elp

选择并添加 H-JTAG.dll, 点击 OK。

点击 debug, keil 即会与 H-JTAG 建立连接,并开始调试。同时 H-JTAG 下方的状态栏也会显示连接的状态,如下图:



则说明断点设置过多。

2.4 在 IAR 中使用 H-JTAG 调试

在工程设置中选择 Debugger 配置页面, Driver 选择 RDI:

tegory:				Factory Settings
eneral Options	Setup Download	Rytra Ontion	e Plugine	-
./L++ Lompiler Issembler	Deceb Dowittoad	Extra option	is i i ugins	
Custom Build	Driver	V	<u>R</u> un to	
uild Actions	RDI	•	main	
inker				
Simulator	-Setup macros-			
Angel		file		
IAR ROM-monitor				
J-Link/J-Trace				
Macraigor RDI	-Device descript	tion file —		
Third-Party Driver	🧮 Override de	efault		
	-			

然后在 RDI 的配置卡中选择 H-JTAG.dll 的安装位置:

Options for node	fproject1"	X
Category: General Options C/C++ Compiler Assembler Custom Build	RDI M <u>a</u> nufacturer RDI driver	Factory Settings
Build Actions Linker Debugger Simulator Angel IAR ROM-monitor	\ARM\H-JTAG\H-JTAG ☐ <u>A</u> llow hardware reset ☐ <u>E</u> TM trace	IG. dll Note Use the RDI menu to specify additional driver settings. (This menu is available after the RDI Juint to the location
Macraigor RDI Third-Party Driver	Log RDI <u>c</u> ommunication \$TOOLKIT_DIR\$\cspycomm.l.	Catch exceptions Reset Data FIQ Undef Prefetch SWI IRQ
		OK Cancel

以上过程即可完成配置。以下就是调试的步骤,这里不在多说。注意在调试前运行 h-jtag, 并正确找到硬件即可。

第3章 编程

H-JTAG 的另一个有用的功能就是 flash 编程。这对于片上 ram 较小需要在 flash 上调 试的 ARM 控制器和有外部 flash 的 ARM 处理器来说,提供一种廉价但却有效的代码下载方 式。

3. 1 AT91SAM7S64

下面以 MCUZONE 的 S64-DEK 为例说明如何编程 flash。

首先连接好板子的 USB 线,这样就会给板子上电。然后连接好 wiggler 与开发板。运行 h-jtag,将会发现 ARM7 内核,如果不能找到,请检查连接。找到内核后,运行 h-flasher,

Flas	sher	Script	Settin
Ē	Star	t H-Flas	her
	Auto) Downloa	.d 🎽

选择正确的 flash 类型,这里选择 AT91SAM7S64。



由于这是内部整合的 flash,所以不需要过多的设置, Memory Config 和 Init Script 可以不用修改:



H-Flasher						
lew Load Save Save A Program Wizard	s Opti <mark>>> Ini</mark>	ons Exit t Script	About			
 Flash Selection Memory Config Init Script Programming H-Flasher Help 		Cmd	Width	Address	Value	*

在 Programming 页面进行实际的编程操作。

H-Flasher				
New Load Save Save	As Option:	s Exit About		
Program Wizard	>> Prog	ramming - AT91SAM7S6	4	
 Flash Selection Memory Config Init Script 	Flash: Target:	Unchecked Unchecked		Check
Programming	Туре:	Auto Flash Download	•	Program
👎 H-Flasher Halp	Src File: Dst Addr:	[
	From:	Entire Chip	•	Erase
	To:	Entire Chip	-	Blank

点击左边的 check, h-flash 将会按照以上的设置查找目标芯片,并给出相应的信息:

rogram Wizard	>> Prog	ramming - AT91SAM7S64		
Flash Selection Memory Config	Flash: Target	AT91SAM7S64 0x27090540 ARM7TDMI Little-Endian		Check
Init Script Programming	Type:	Plain Binary Format	•	Program
H-flasher Help	Dst	0x000000		
	From:	Entire Chip	•	Erase
	To:	Entire Chin	T	Blank

以上就是正确连接上后,找到信息。此时点击下方的 Erase,擦除芯片:



点击 Blank 可以检查 flash 是否擦除完成。

H-Flasher	
00:00:50 Flash memory is b	olank !
	Close

在中间的 Program 选项卡中选择目标文件的类型,再设置好目标文件的路径,下方的 Dst 设置为 0,也就 flash 的地址。然后点击 Program:

Programmed and verified suc	cessfully.
00:01:70 100% 16 KB/s	Size = 27.4 KE

以上就是编程完成的画面。

选择 H-Flasher Help 将得到一些帮助信息,提示了 flash 编程的步骤。

🔁 H-Flasher	
New Load Save Save	As Options Exit About
Program Wizard	>> H-Flasher Help
 Flash Selection Memory Config 	To program flash device, please follow the steps listed below:
3 Init Script	2. Config memory of target
Programming	3. Fill the script for initialization
H-Flasher Help	4. Program/Erase/Check Blank To view the following helps, please expand (double click) the left item: - Help on flash selection - Help on memory config - Help on init script - Help on programming

上方菜单栏中的主要功能在于加载和保存 flash 编程的设置,方便多次使用同一设置。

3.2 S3C44B0 公版

44B0 的公版使用了外置的 flash, 大都是 SST 的产品。下面以一块装配了 SST39VF1601 的板子为例说明编程方法。

正确连接硬件后运行 h-jtag:



运行 h-flasher 后选择 flash 类型:

er Lood Save Save An	Options Exit About		
Program Wizard	> Flash Selection		
 Flack Selection Hencey Config Init Script Programming H-Flacker Help 	SST 39SF010A SST 39SF020A SST 39SF040 SST 39SF040 SST 39VF040 SST 39VF040 SST 39VF040 SST 39VF040 SST 39VF040 SST 39VF1602 SST 39VF1602	8	Vanda: SST Paetia: SST39AF1601 Type: NDR Flash Secto: 512 Size: 2.MB ID: 0x234800BF Widt: 16-BR

Memory Config 中设置如下:

Flash 的起始地址为 0(根据 44B0 公版定义), RAM 起始地址设置为 S3C44B0 内部 RAM 的起始地址。

Init Script 中需要设置如下:

🖥 H-Flasher						
New Load Save Save A: Program Wizard	s Optio >> Init	ons Exit Script	About			
1 Flash Selection	Idx	Cmd	Width	Address	Value	
 Memory Config Init Script Programming H-Flasher Help 	1	Setmem	32	0x01C00000	0	:

需要向 0x01C00000 处写入 0 (32 位写操作),禁止 WDT。

在 Programming 选项卡中点击 check:

H-Flasher				
ew Load Save Save Program Wizard	As Uption	s Exit About ramming - SST39VF1601		
 Flash Selection Memory Config Tait Semist 	Flash: Target	SST39VF1601 Ox234B00BF ARM7TDMI Little-Endian		Check
Friedrick Script Programming	Type:	Plain Binary Format	•	Program
👎 H-Flasher Help	Src Dst	0x000000		
	From:	Entire Chip	•	Erase
	To:	Entire Chip	-	Blank

可以看到读出的 flash 芯片的 ID 与处理器核心类型。

选择好文件路径及类型后设置好 Dst 位置为 0,即可点击 Program 进行编程:

Programmed and verified succes	sfully.
00:01:30 100% 21 KB/s	Size = 27.4 KE

3.3 LPC2132 测试板

由于 LPC 使用了 ARM7TDMI-S 内核,注意 LPC 的 RTCK 需要下拉才能进入 JTAG 模式,以下给出编程的步骤。

正确连接硬件后运行 h-jtag:



选择器件类型:

	- opennie were mere		
togram Wizard	>> Firsh Selection		
Flash Selection	LPC2124		Vendor : PHILIPS
Menory Config	LPC2129 LPC2131		PartNo LPC2132
Init Script	1202132		Type: On-Chip Flash
Programing	LPC2136		Sector 9
N-Flasher Help	LPC2138 LPC2141		Size 64 HB
	LPC2142 LPC2144		ID: 0x00029F11
	LPC2146		Width: 8-Bit
	LPC2194		
	LPC2212 LPC2214		
	LPC2292 LPC2292		
	A WYWTY	*	

器件检测:

rooram Wizard	>> Prog	ramming - LPC2132		
Flash Selection Nemory Config	Flash: Targat	LPC2132 0#0002FF11 ARMTIDMI-S Little-Endian		Check
Init Script Programing	Type:	Flain Binary Format	۲	Prope
N-Flasher Help	Src Dut	0x000000		
	Pr on :	Entire Chip	•	Erste
	Ia:	Entire Chin	-	Flask

以下的编程步骤与其它相同。对于 LPC2000 系列有个特殊的选项。点击菜单中的 Optinos:



勾选这个选项将在编程时自动生成 LPC2000 系列所需要的向量之和。

附录A Wiggler 电路图

