AT91系列ARM芯片的开发技术

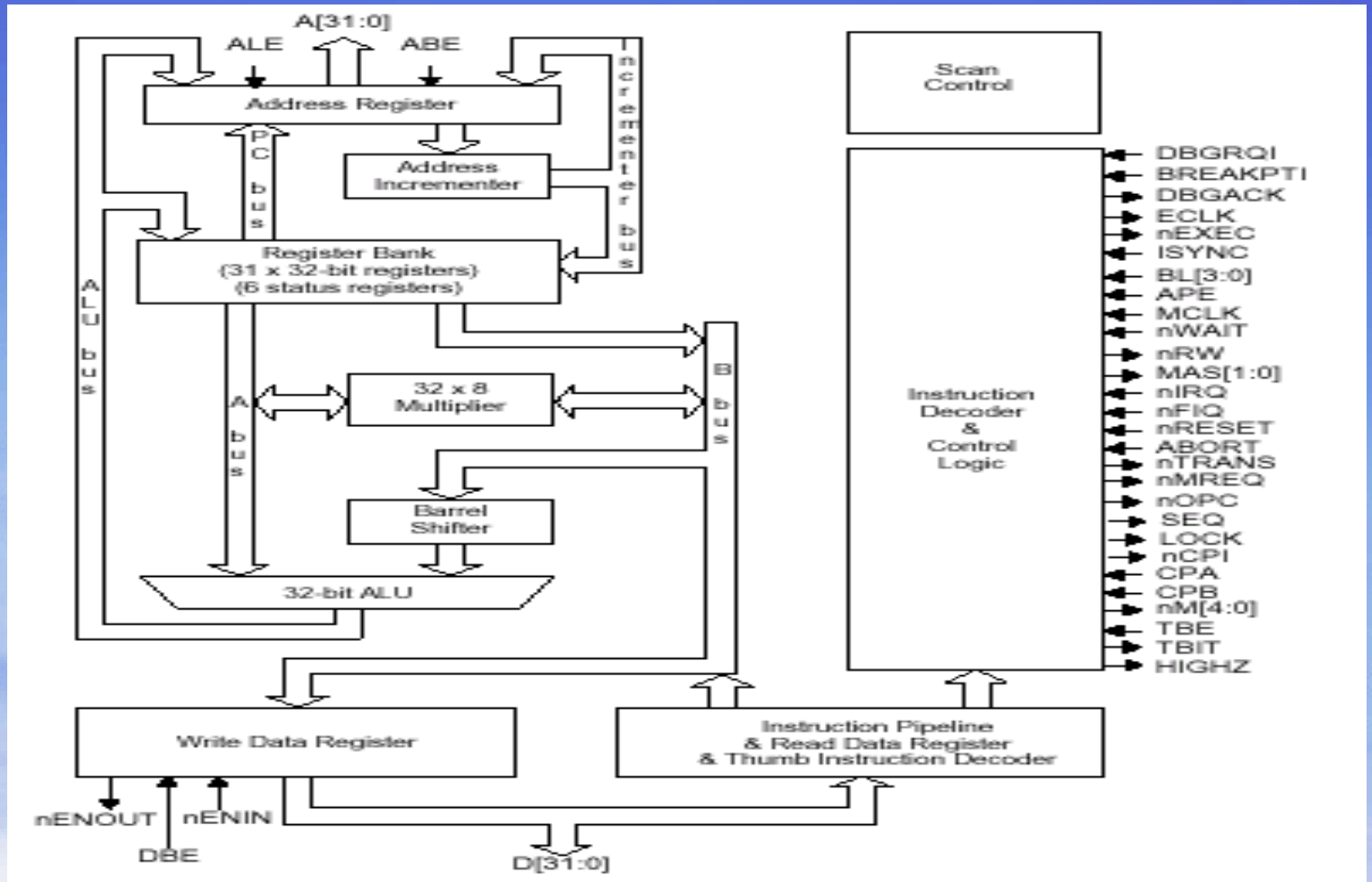
- 深圳市英蓓特信息技术有限公司

http://www.embedinfo.com

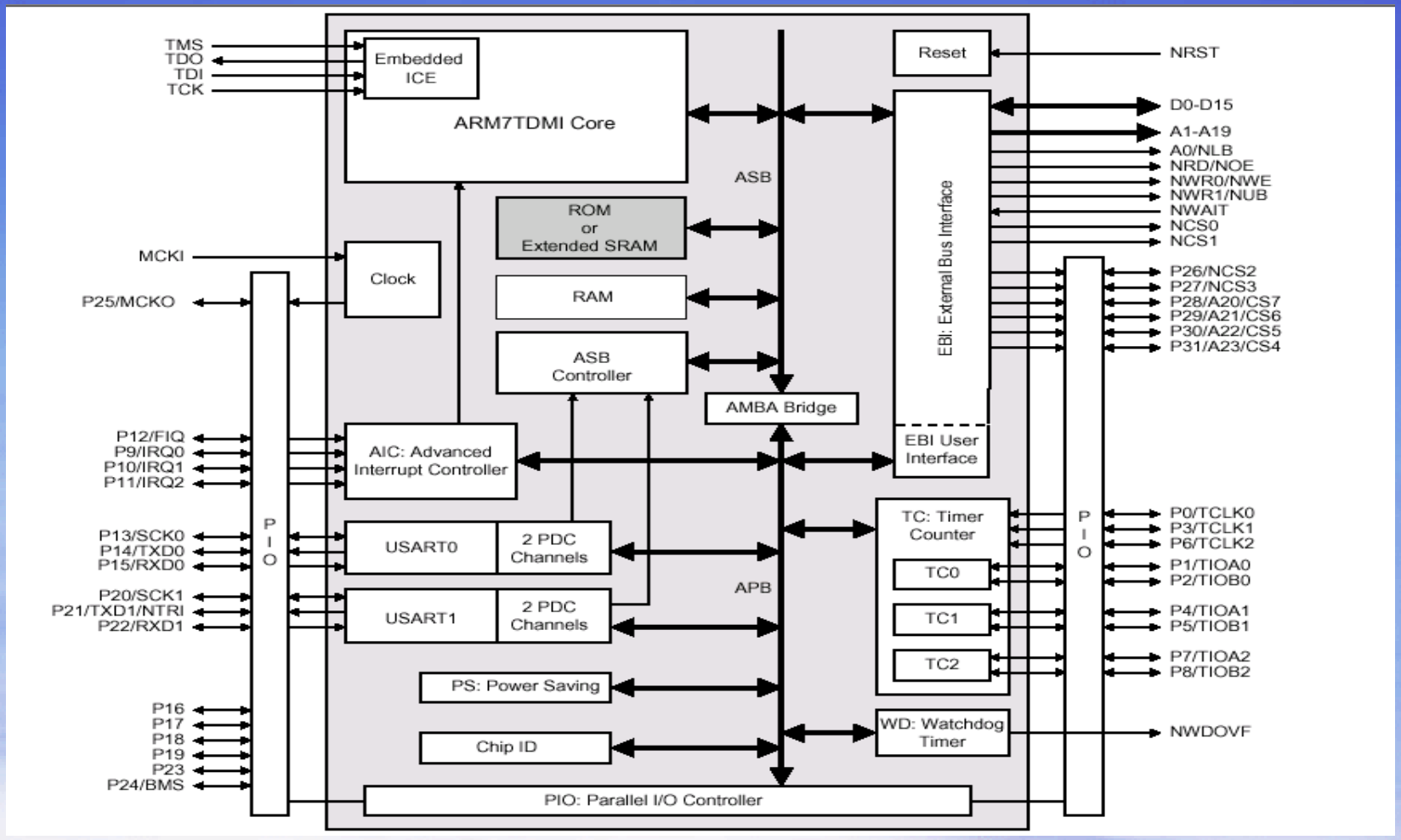
主要内容

- 硬件最小系统
- 硬件设计概述
- 硬件调试过程
- 调试接口定义
- 软件调试流程
- 相关配置文件示例
- 启动代码详解
- 高级调试技术

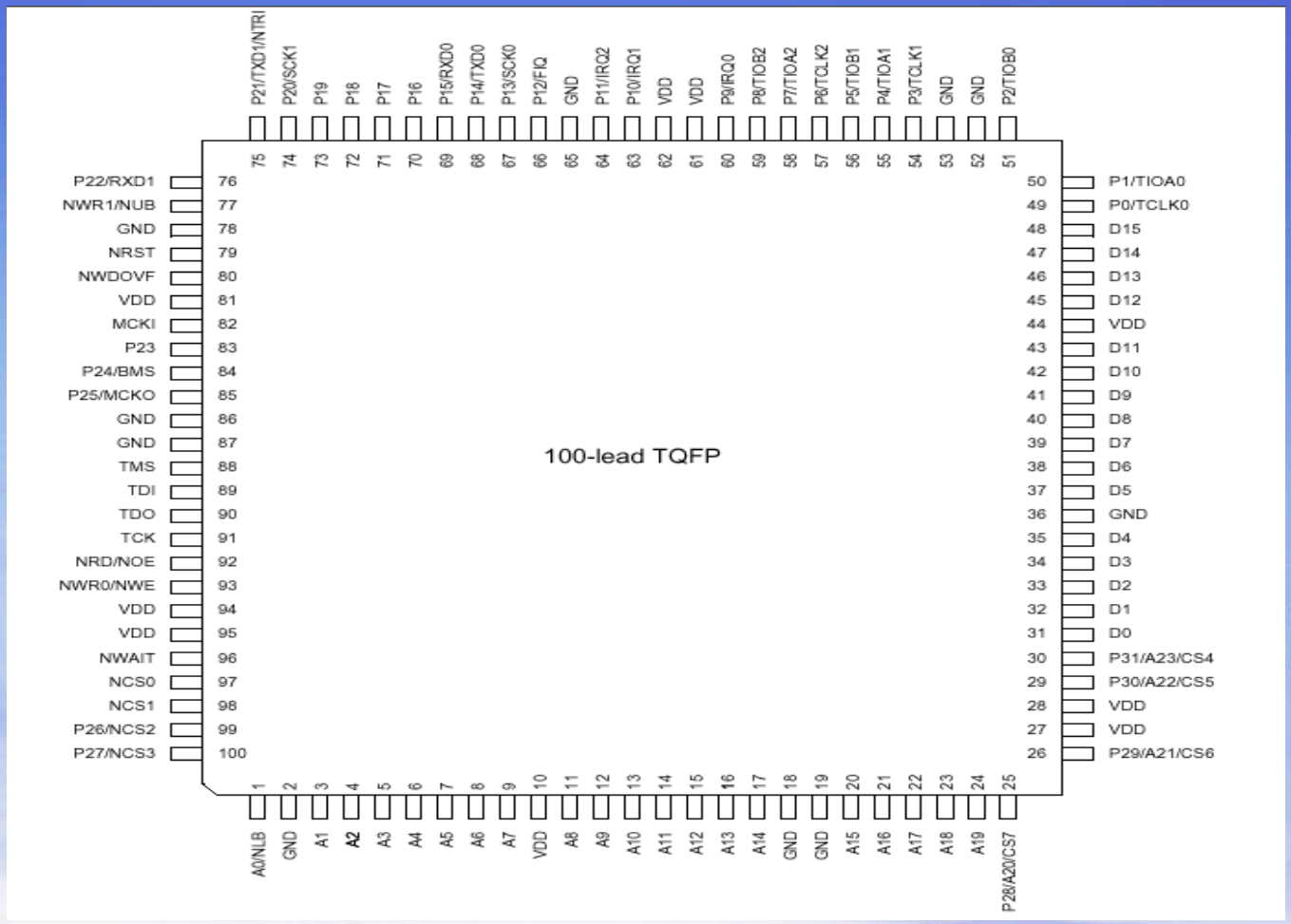
ARM内核框图


A[31:0]
ALE
ABE
Address Register
Incrementer bus
PC bus
Address Incrementer
Register Bank
(31 x 32-bit registers)
(6 status registers)
ALU bus
A bus
B bus
32 x 8
Multiplier
Barrel
Shifter
32-bit ALU
Scan
Control
Instruction
Decoder
&
Control
Logic
DBGRQI
BREAKPTI
DBGACK
ECLK
nEXEC
ISYNC
BL[3:0]
APE
MCLK
nWAIT
nRW
MAS[1:0]
nIRQ
nFIQ
nRESET
ABORT
nTRANS
nMREQ
nOPC
SEQ
LOCK
nCPI
CPA
CPB
nM[4:0]
TBE
TBIT
HIGHZ
Write Data Register
Instruction Pipeline
& Read Data Register
& Thumb Instruction Decoder
nENOUT
nENIN
DBE
D[31:0]

40800框图


TMS
TDO
TDI
TCK
Embedded ICE
ARM7TDMI Core
Reset
NRST
D0-D15
A1-A19
A0/NLB
NRD/NOE
NWR0/NWE
NWR1/NUB
NWAIT
NCS0
NCS1
ASB
ROM or Extended SRAM
RAM
ASB Controller
AMBA Bridge
EBI: External Bus Interface
EBI User Interface
MCKI
Clock
P25/MCKO
P26/NCS2
P27/NCS3
P28/A20/CS7
P29/A21/CS6
P30/A22/CS5
P31/A23/CS4
P12/FIQ
P9/IRQ0
P10/IRQ1
P11/IRQ2
AIC: Advanced Interrupt Controller
P13/SCK0
P14/TXD0
P15/RXD0
USART0
2 PDC Channels
P20/SCK1
P21/TXD1/NTRI
P22/RXD1
USART1
2 PDC Channels
APB
TC: Timer Counter
TC0
TC1
TC2
P0/TCLK0
P3/TCLK1
P6/TCLK2
P1/TIOA0
P2/TIOB0
P4/TIOA1
P5/TIOB1
P7/TIOA2
P8/TIOB2
PS: Power Saving
Chip ID
WD: Watchdog Timer
NWDOVF
P16
P17
P18
P19
P23
P24/BMS
PIO: Parallel I/O Controller
P I O
P I O

AT91M40800


100-lead TQFP
P22/RXD1 76
NWR1/NUB 77
GND 78
NRST 79
NWDOVF 80
VDD 81
MCKI 82
P23 83
P24/BMS 84
P25/MCKO 85
GND 86
GND 87
TMS 88
TDI 89
TDO 90
TCK 91
NRD/NOE 92
NWR0/NWE 93
VDD 94
VDD 95
NWAIT 96
NCS0 97
NCS1 98
P26/NCS2 99
P27/NCS3 100
P21/TXD1/NTRI 75
P20/SCK1 74
P19 73
P18 72
P17 71
P16 70
P15/RXD0 69
P14/TXD0 68
P13/SCK0 67
P12/FIQ 66
GND 65
P11/IRQ2 64
P10/IRQ1 63
VDD 62
VDD 61
P9/IRQ0 60
P8/TIOB2 59
P7/TIOA2 58
P6/TCLK2 57
P5/TIOB1 56
P4/TIOA1 55
P3/TCLK1 54
GND 53
GND 52
P2/TIOB0 51
P1/TIOA0 50
P0/TCLK0 49
D15 48
D14 47
D13 46
D12 45
VDD 44
D11 43
D10 42
D9 41
D8 40
D7 39
D6 38
D5 37
GND 36
D4 35
D3 34
D2 33
D1 32
D0 31
P31/A23/CS4 30
P30/A22/CS5 29
VDD 28
VDD 27
P29/A21/CS6 26
A0/NLB 1
GND 2
A1 3
A2 4
A3 5
A4 6
A5 7
A6 8
A7 9
VDD 10
A8 11
A9 12
A10 13
A11 14
A12 15
A13 16
A14 17
GND 18
GND 19
A15 20
A16 21
A17 22
A18 23
A19 24
P28/A20/CS7 25

40x系列CPU

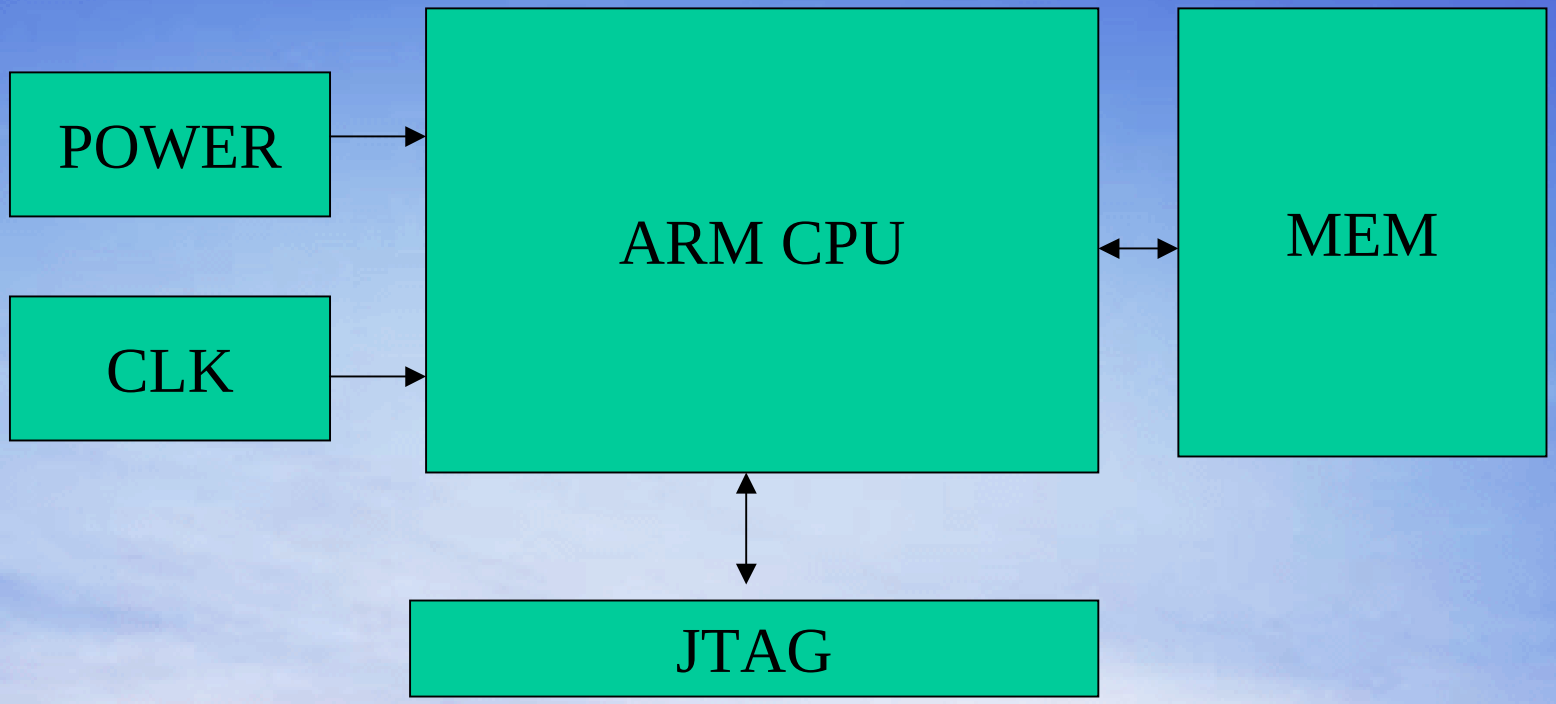
Microcontroller	Primary SRAM Bank	Secondary SRAM Bank	ROM
AT91M40800	8K Bytes	–	–
AT91R40807	8K Bytes	128K Bytes	–
AT91M40807	8K Bytes	–	128K Bytes
AT91R40008	256K Bytes	–	–

硬件最小系统

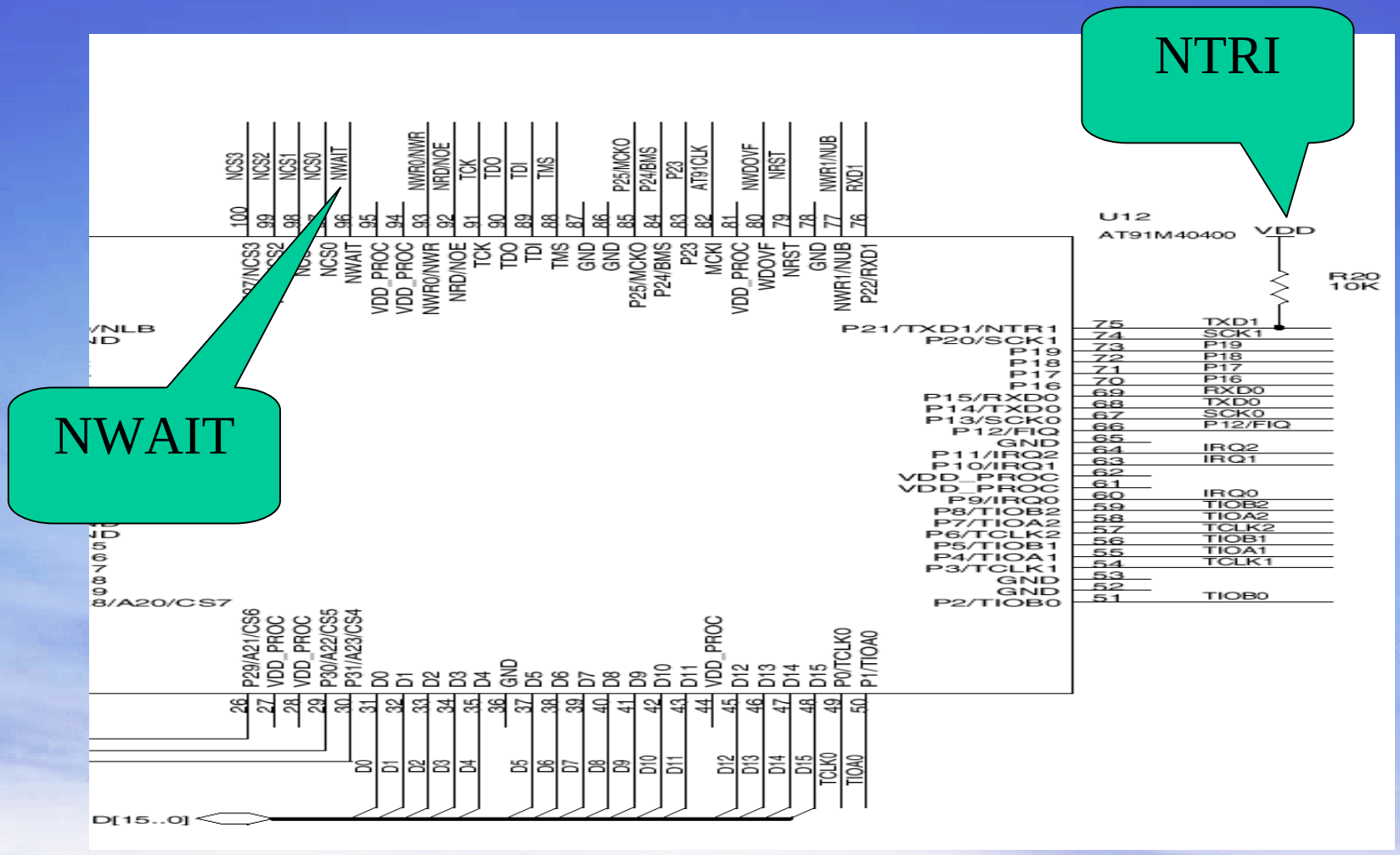
对ARM CPU来讲，可以调试的最小硬件系统，包括电源、CPU芯片，晶振、存储器（外部的或者内部的）和JTAG调试接口。

一般，为了直观，可以连接一两只LED管，指示CPU的工作状态。

硬件最小系统



需要注意的管脚

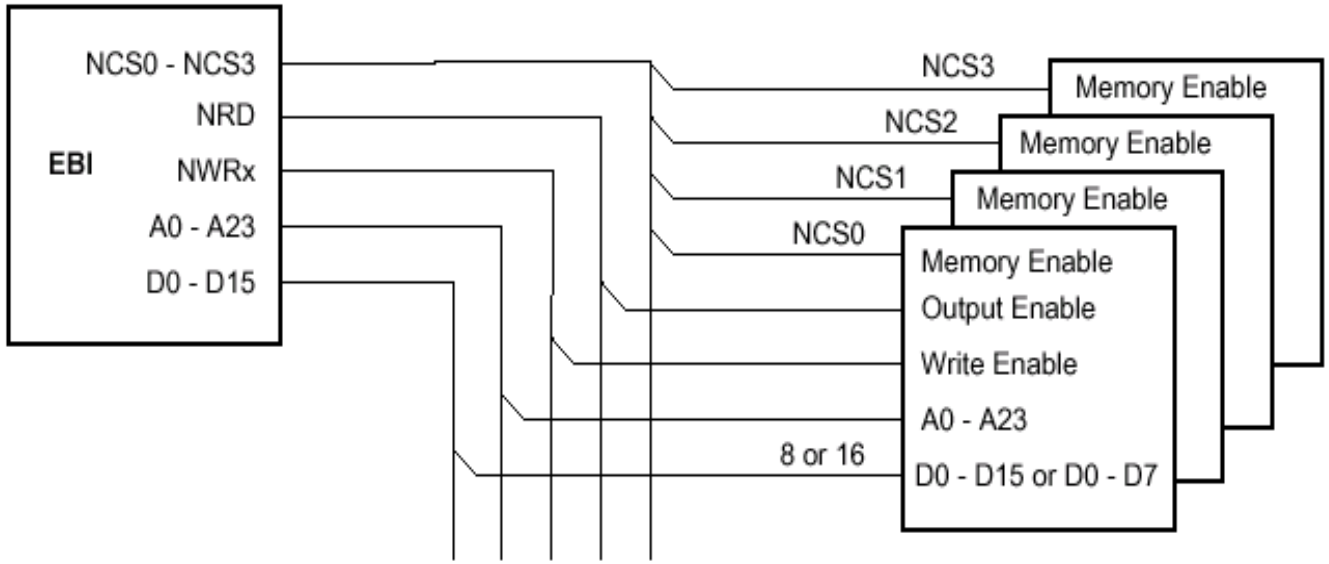


需要注意的管脚

❖ NTRI	Tri-state Mode Select	1
❖ NWAIT	Wait Input	1
❖ JTAGSEL	Selects between ICE and JTAG	0
❖ nTRST	Not JTAG reset	1

存储器接口的例子

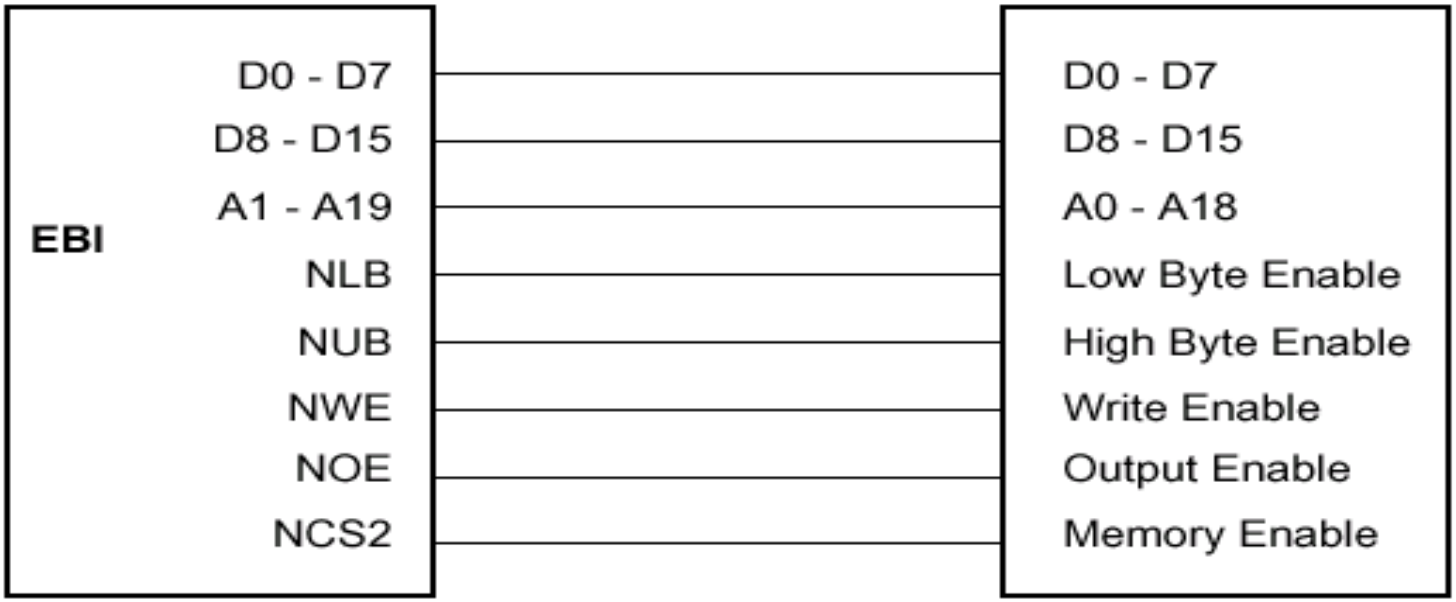
Figure 7. Memory Connections for Four External Devices



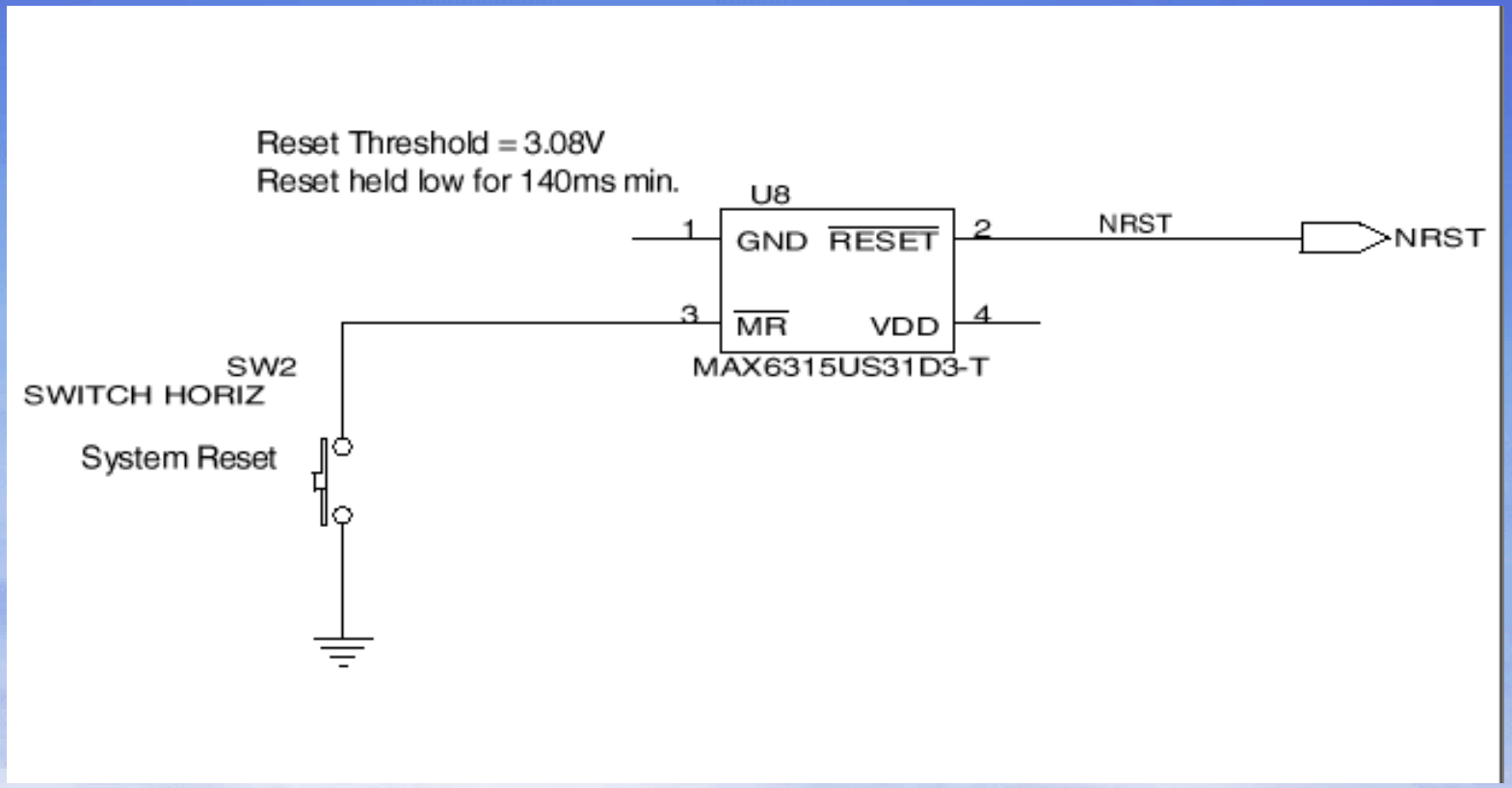
Note: For four external devices, the maximum address space per device is 16M bytes.

存储器接口（16位）

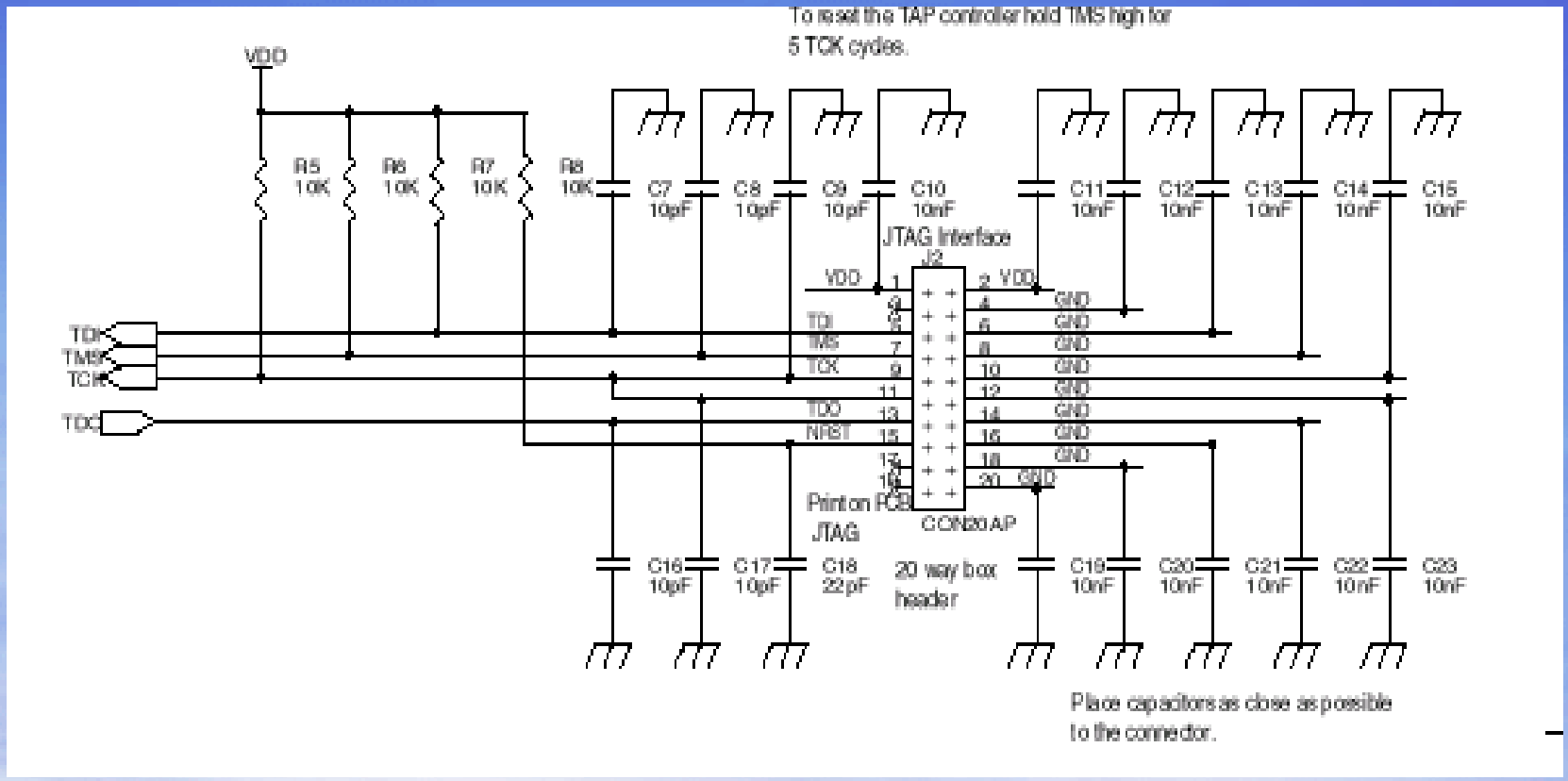
Figure 10. Memory Connection for a 16-bit Data Bus



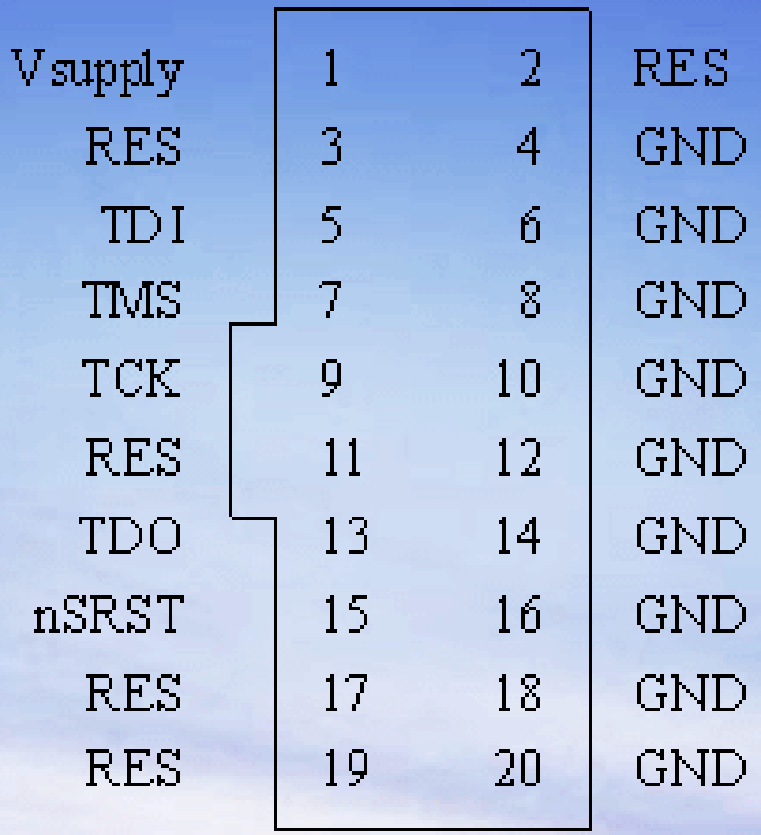
复位电路


Reset Threshold = 3.08V
Reset held low for 140ms min.
U8
1
GND
RESET
2
NRST
NRST
3
MR
VDD
4
MAX6315US31D3-T
SW2
SWITCH HORIZ
System Reset

JTAG接口电路


To reset the TAP controller hold TMS high for 5 TCK cycles.
VDD
R5 10K
R6 10K
R7 10K
R8 10K
C7 10pF
C8 10pF
C9 10pF
C10 10nF
C11 10nF
C12 10nF
C13 10nF
C14 10nF
C15 10nF
JTAG Interface
J2
VDD
TDI
TMS
TCK
TDO
NRST
GND
TDI
TMS
TCK
TDO
Print on PCB JTAG
CON20AP
20 way box header
C16 10pF
C17 10pF
C18 22pF
C19 10nF
C20 10nF
C21 10nF
C22 10nF
C23 10nF
Place capacitors as close as possible to the connector.

JTAG接口定义



Vsupply	1	2	RES
RES	3	4	GND
TDI	5	6	GND
TMS	7	8	GND
TCK	9	10	GND
RES	11	12	GND
TDO	13	14	GND
nSRST	15	16	GND
RES	17	18	GND
RES	19	20	GND

硬件设计注意事项（一）

- CPU复位信号的处理
- 模式管脚的正确连接
- JTAG复位信号的上拉
- JTAG输入信号的上拉
- 信号串扰

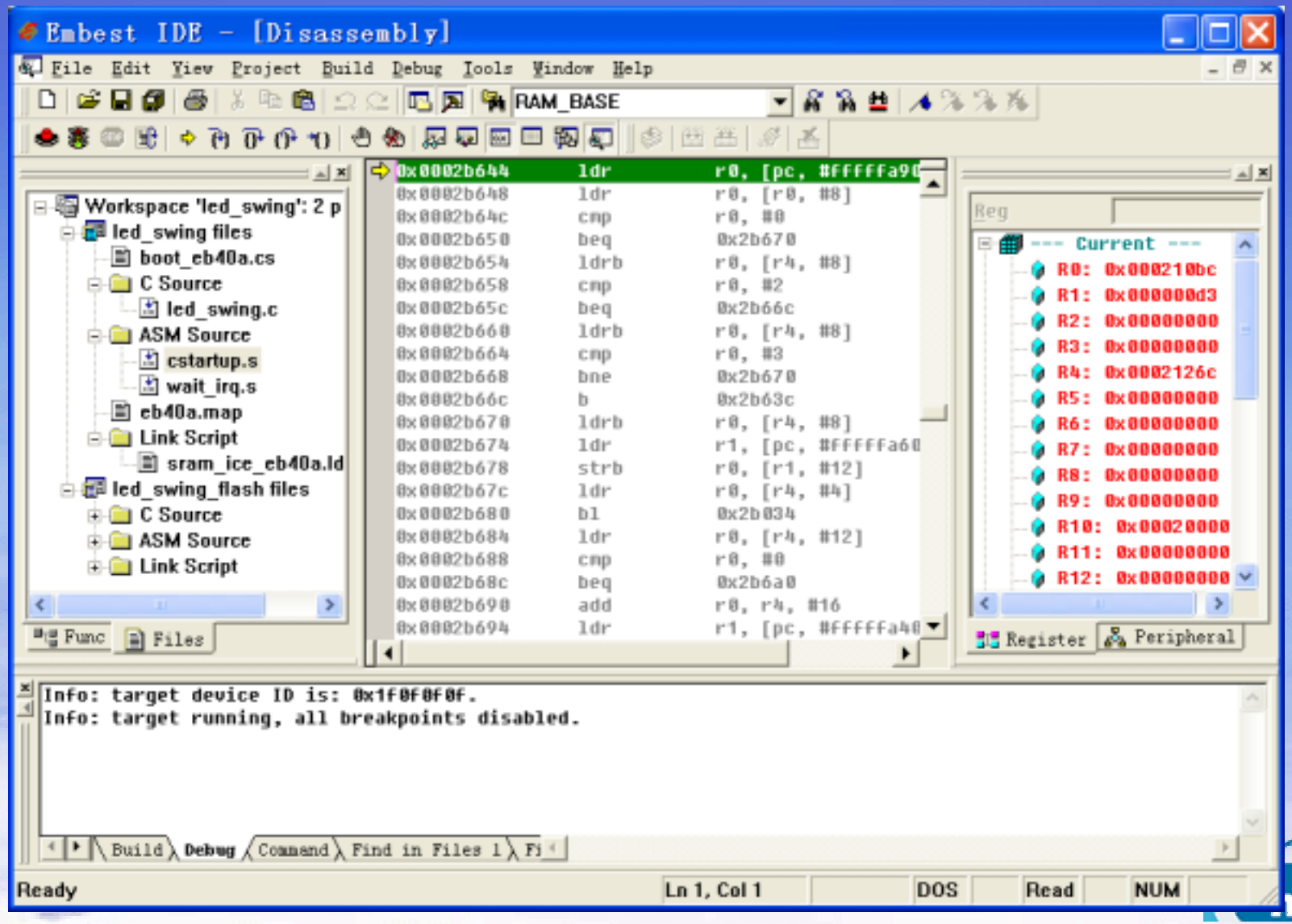
硬件设计注意事项（二）

- 多电平电源的设计
- 外部器件的接口电平
- 存储器总线接口
- 存储器等待时间nWAIT

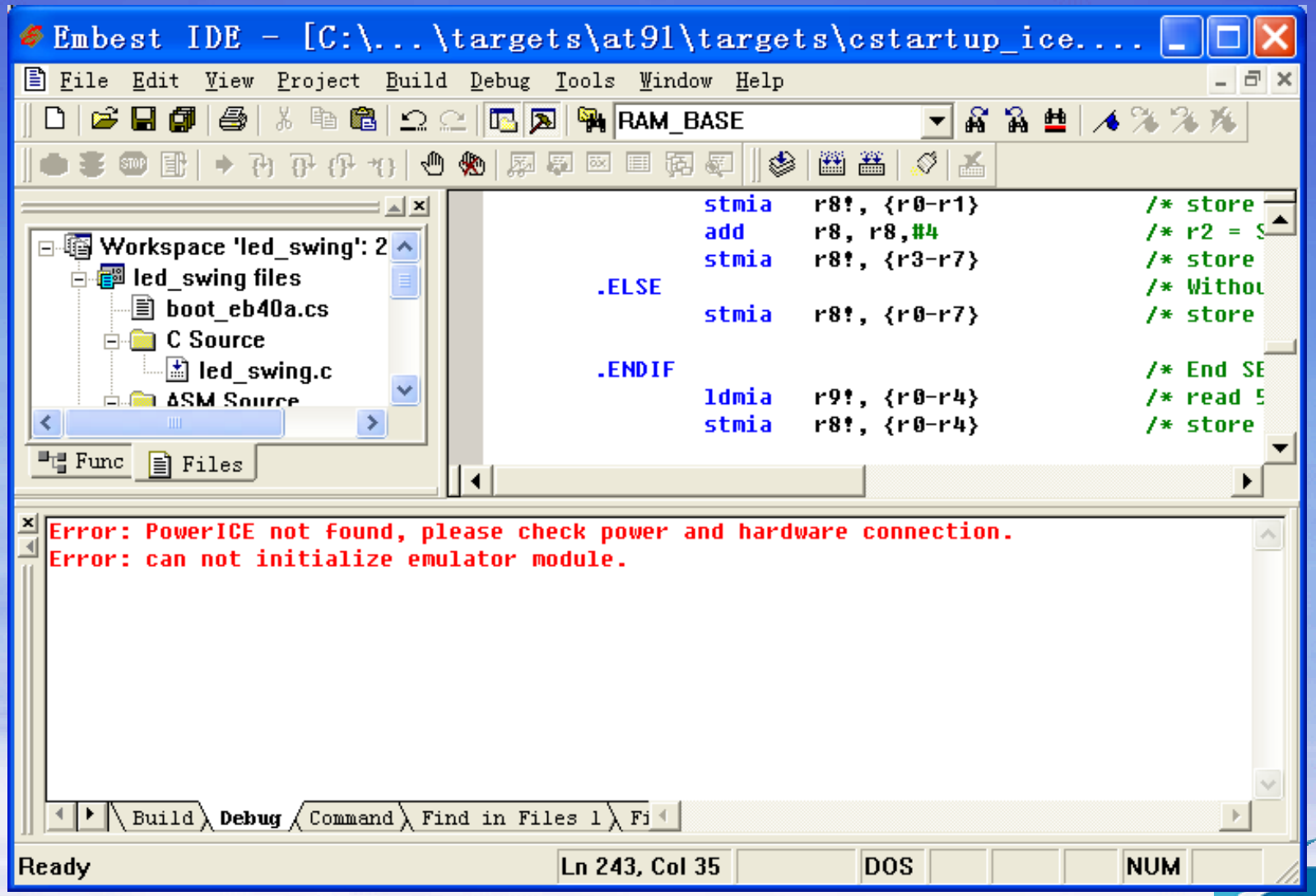
仿真器与CPU连接

- 连接之前，核对电气特性要求
- JTAG接口同JTAG仿真器硬件连接，利用相应的调试工具观察CPU状态
- 确定CPU与调试工具可以正常通信
- 调试工具可以检查和控制CPU的运行

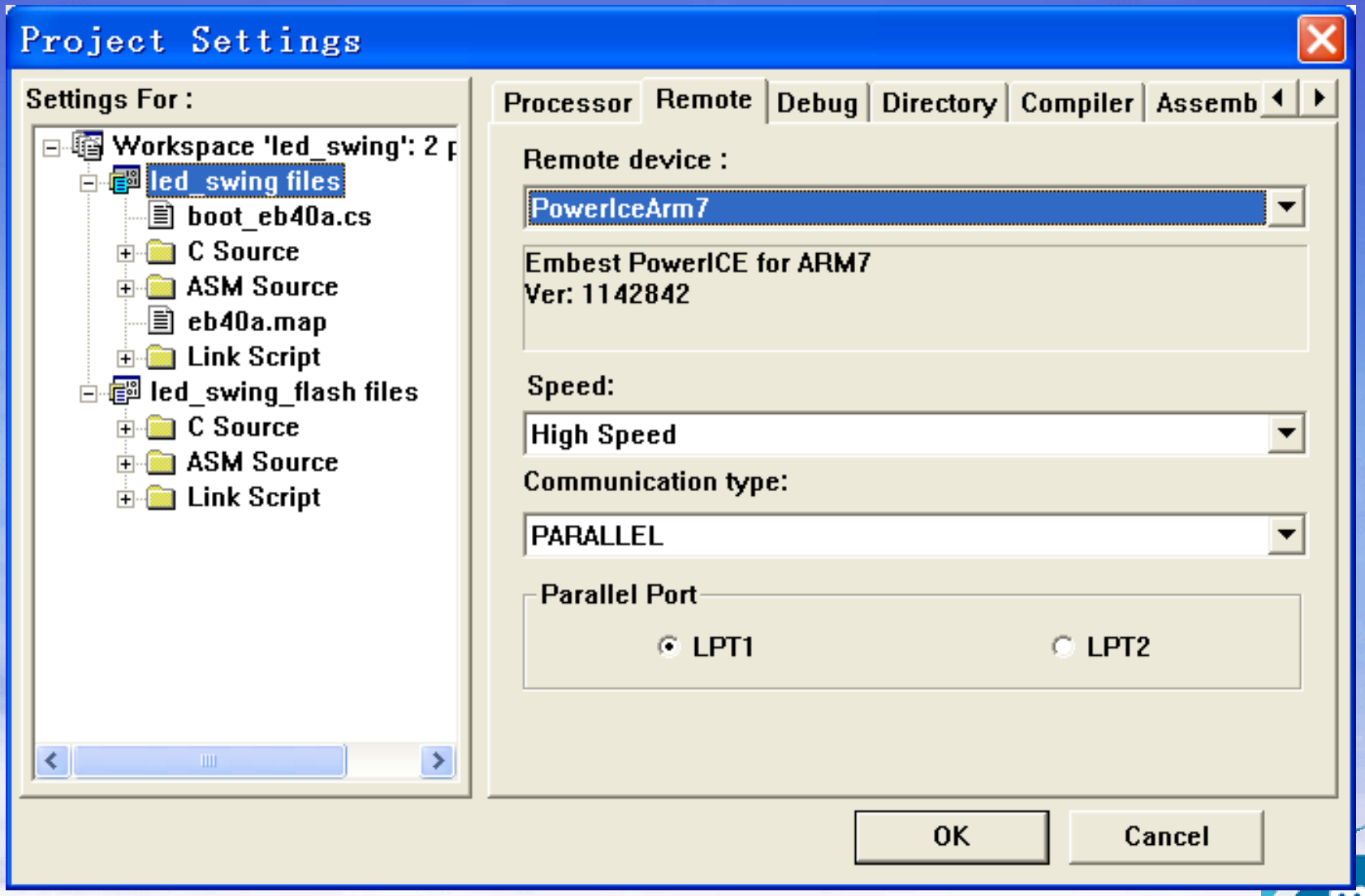
确定CPU与调试工具可以正常通信



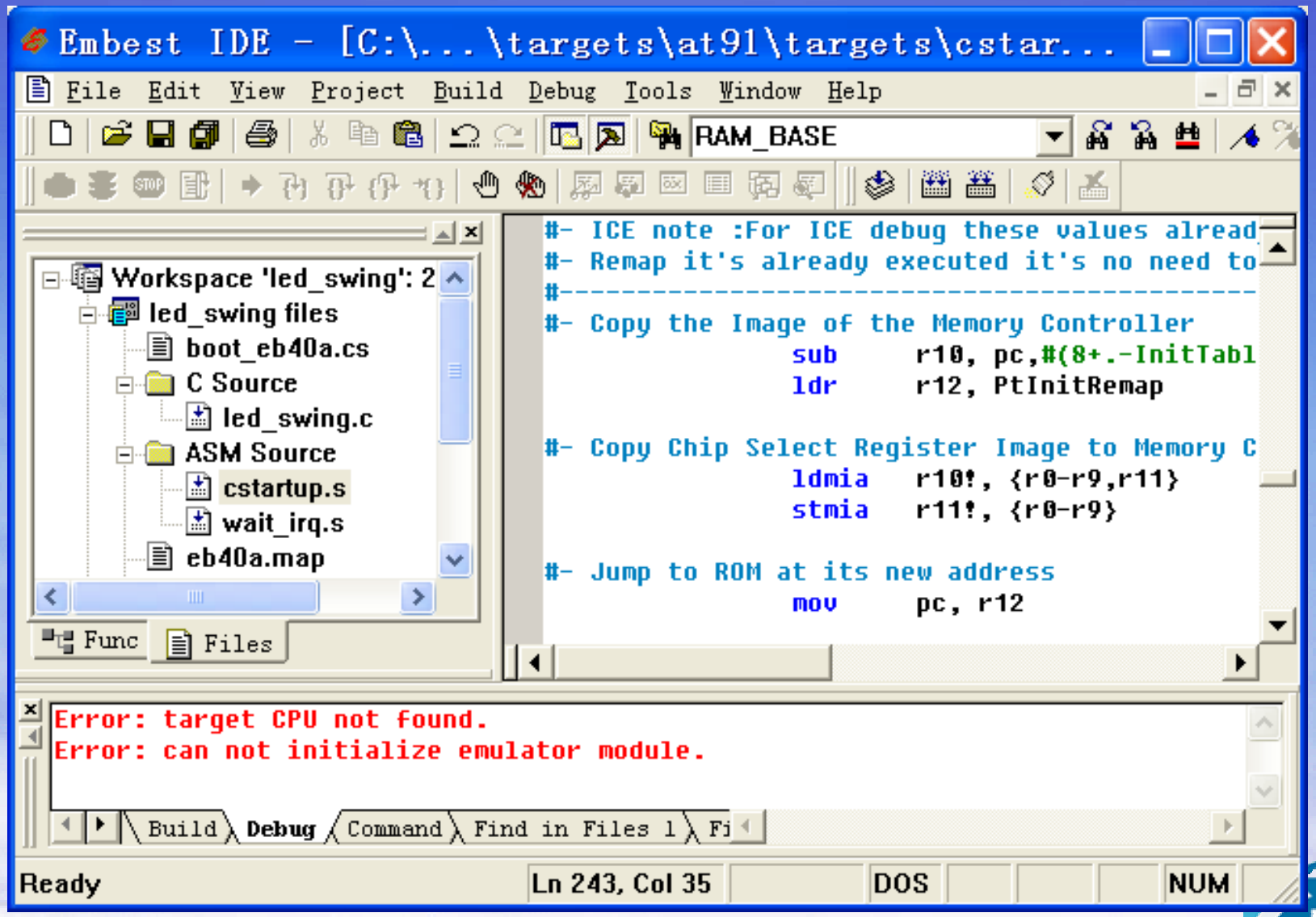
仿真器出错提示信息


Embest IDE - [C:\...\targets\at91\targets\cstartup_ice....
File Edit View Project Build Debug Tools Window Help
RAM_BASE
Workspace 'led_swing': 2
led_swing files
boot_eb40a.cs
C Source
led_swing.c
ASM Source
Func
Files
stmia r8!, {r0-r1} /* store
add r8, r8,#4 /* r2 = S
stmia r8!, {r3-r7} /* store
.ELSE /* Withou
stmia r8!, {r0-r7} /* store
.ENDIF /* End SE
ldmia r9!, {r0-r4} /* read S
stmia r8!, {r0-r4} /* store
Error: PowerICE not found, please check power and hardware connection.
Error: can not initialize emulator module.
Build
Debug
Command
Find in Files 1
Fi
Ready
Ln 243, Col 35
DOS
NUM

Remote设置对话框



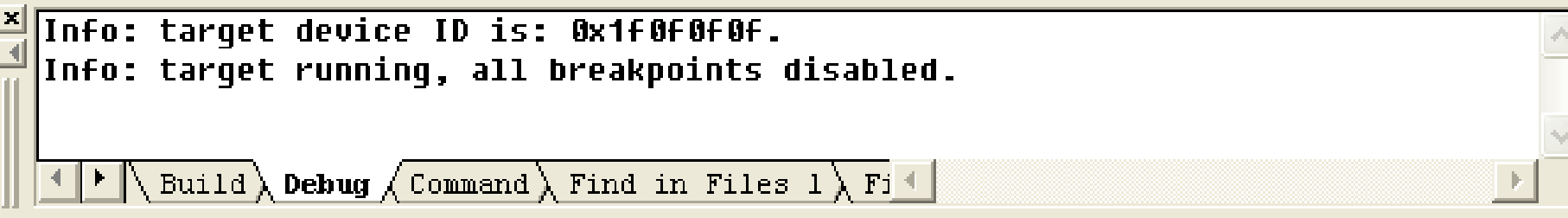
JTAG出错提示信息


Embest IDE - [C:\...\targets\at91\targets\cstar...
File Edit View Project Build Debug Tools Window Help
RAM_BASE
Workspace 'led_swing': 2
led_swing files
boot_eb40a.cs
C Source
led_swing.c
ASM Source
cstartup.s
wait_irq.s
eb40a.map
Func
Files
#- ICE note :For ICE debug these values alread
#- Remap it's already executed it's no need to
#- Copy the Image of the Memory Controller
sub r10, pc,#(8+.-InitTabl
ldr r12, PtInitRemap
#- Copy Chip Select Register Image to Memory C
ldmia r10!, {r0-r9,r11}
stmia r11!, {r0-r9}
#- Jump to ROM at its new address
mov pc, r12
Error: target CPU not found.
Error: can not initialize emulator module.
Build
Debug
Command
Find in Files 1
Ready
Ln 243, Col 35
DOS
NUM

JTAG出错提示的可能原因

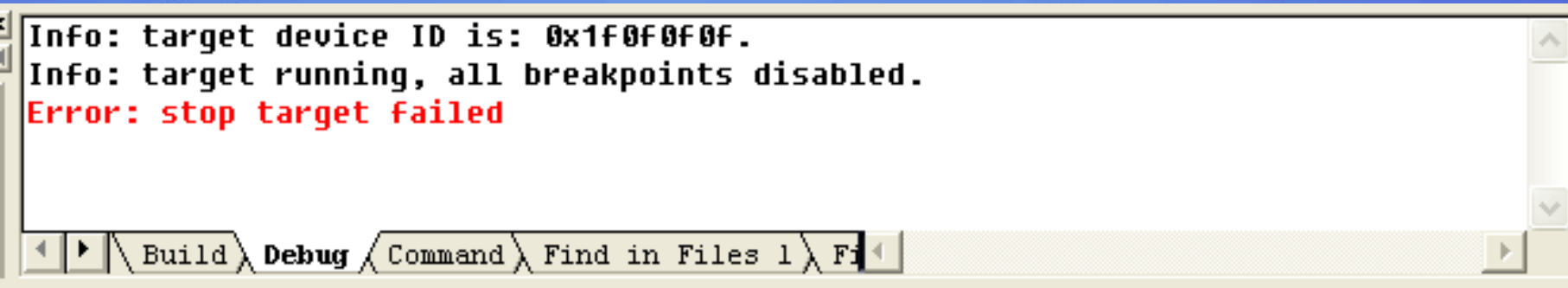
- ❖相关管脚电平（NTRI、JTAGSEL、nTRST）
- ❖CPU电源
- ❖JTAG接口定义、接口电平

CPU与调试工具正常通信时的提示信息



调试工具可以检查和控制CPU的运行

调试工具控制CPU的运行失败时的提示信息



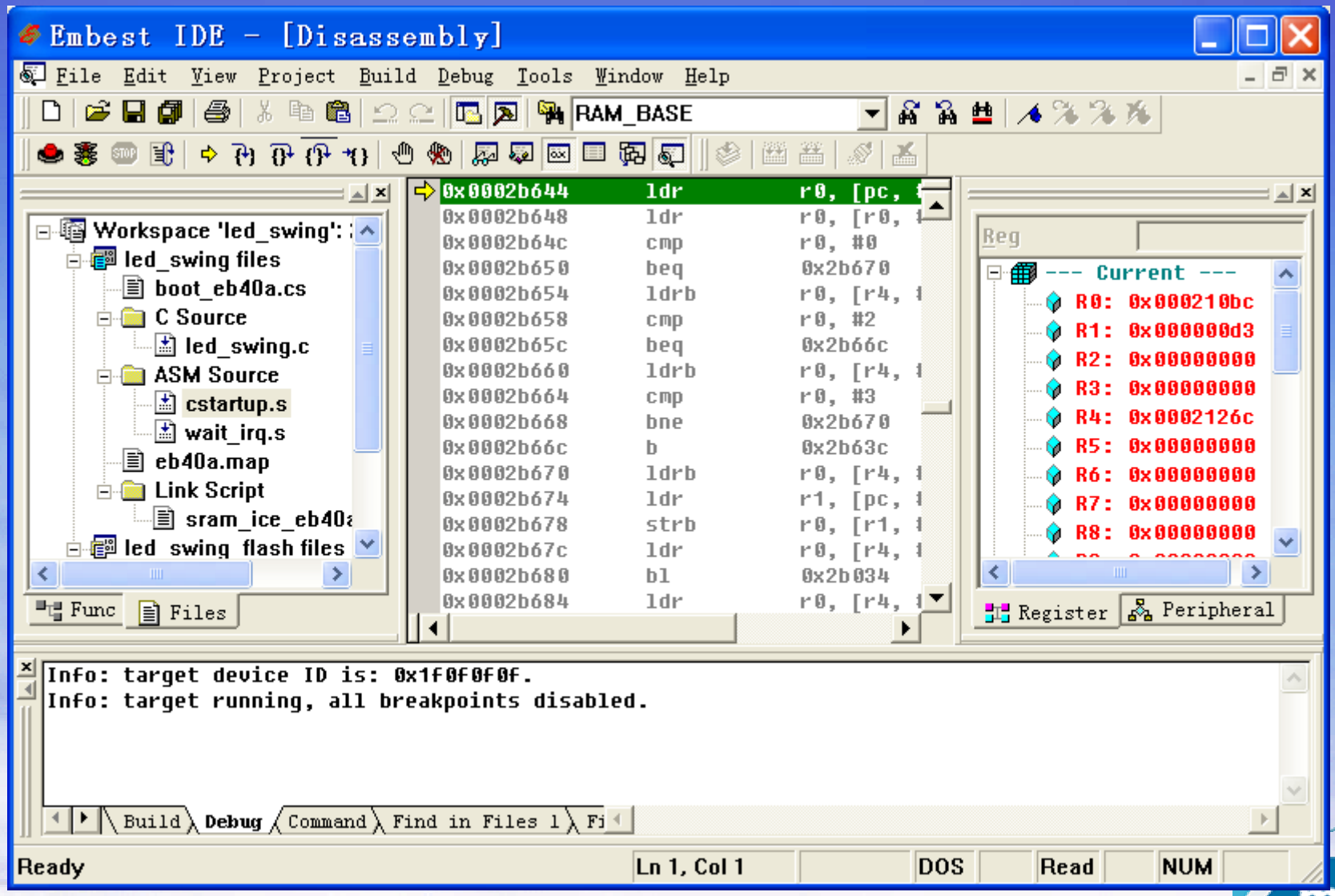
调试工具控制CPU的运行失败的可能原因

- ❖相关管脚电平（NWAIT）
- ❖晶振
- ❖CPU供电电压、电流
- ❖仿真器设置（ARM7/ARM9）

检查和控制CPU失败的可能原因列表

错误提示	Target CPU not found	Stop target failed
可能原因	相关管脚电平（NTRI、JTAGSEL、nTRST）	相关管脚电平（NWAIT）
	CPU供电电压、电流	CPU供电电压、电流
	JTAG接口定义、接口电平	晶振
		仿真器设置（ARM7/ARM9）

调试工具成功检查和控制CPU的运行时的界面



进一步验证硬件连接的正确性和稳定性

- 访问CPU内部寄存器
- 访问芯片内部的外围寄存器
- 在内部RAM运行简单的测试程序

REMAP


地址
命令执行前
命令执行后
大小
0xFFFFFFFF
片内外设
片内外设
4M Bytes
0xFFC00000
0xFFBFFFFF
保留
外部芯片
（最多 8 片）
0x00400000
0x003FFFFF
片内 RAM
保留
1M Bytes
0x00300000
0x002FFFFF
片内器件保留
片内器件保留
1M Bytes
0x00200000
0x001FFFFF
片内扩展 RAM
片内扩展 RAM
1M Bytes
0x00100000
0x000FFFFF
NCS0 所选器件
片内 RAM
1M Bytes
0x00000000

设置（配置）EBI

- 利用调试工具正确配置EBI
- 尝试访问RAM区
- 检查数据总线与地址总线的正确性

EBI设置（配置）内容

- 存储区类型
- 存储区大小和起始地址
- 访问等待周期
- 访问时序
- 数据总线宽度
- 其他需要设置的内容

设置（配置）EBI的方法

- 启动代码完成
- 手工实现
- 利用命令脚本

存储控制器

EBI Chip Select Register

Register Name: EBI_CSR0 - EBI_CSR7
Access Type: Read/Write
Reset Value: See Table 6
Absolute Address: 0xFFE00000 - 0xFFE0001C
Offset: 0x00 - 0x1C

31	30	29	28	27	26	25	24
BA							

23	22	21	20	19	18	17	16
BA				–	–	–	–

15	14	13	12	11	10	9	8
–	–	CSEN	BAT	TDF			PAGES

7	6	5	4	3	2	1	0
PAGES	–	WSE	NWS			DBW	

存储控制器

EBI Memory Control Register

Register Name: EBI_MCR
Access Type: Read/Write
Reset Value: 0
Absolute Address: 0xFFE00024
Offset: 0x24

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	DRP	–	ALE		

存储控制器

EBI Remap Control Register

Register Name: EBI_RCR
Access Type: Write only
Absolute Address: 0xFFE00020
Offset: 0x20

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	RCB

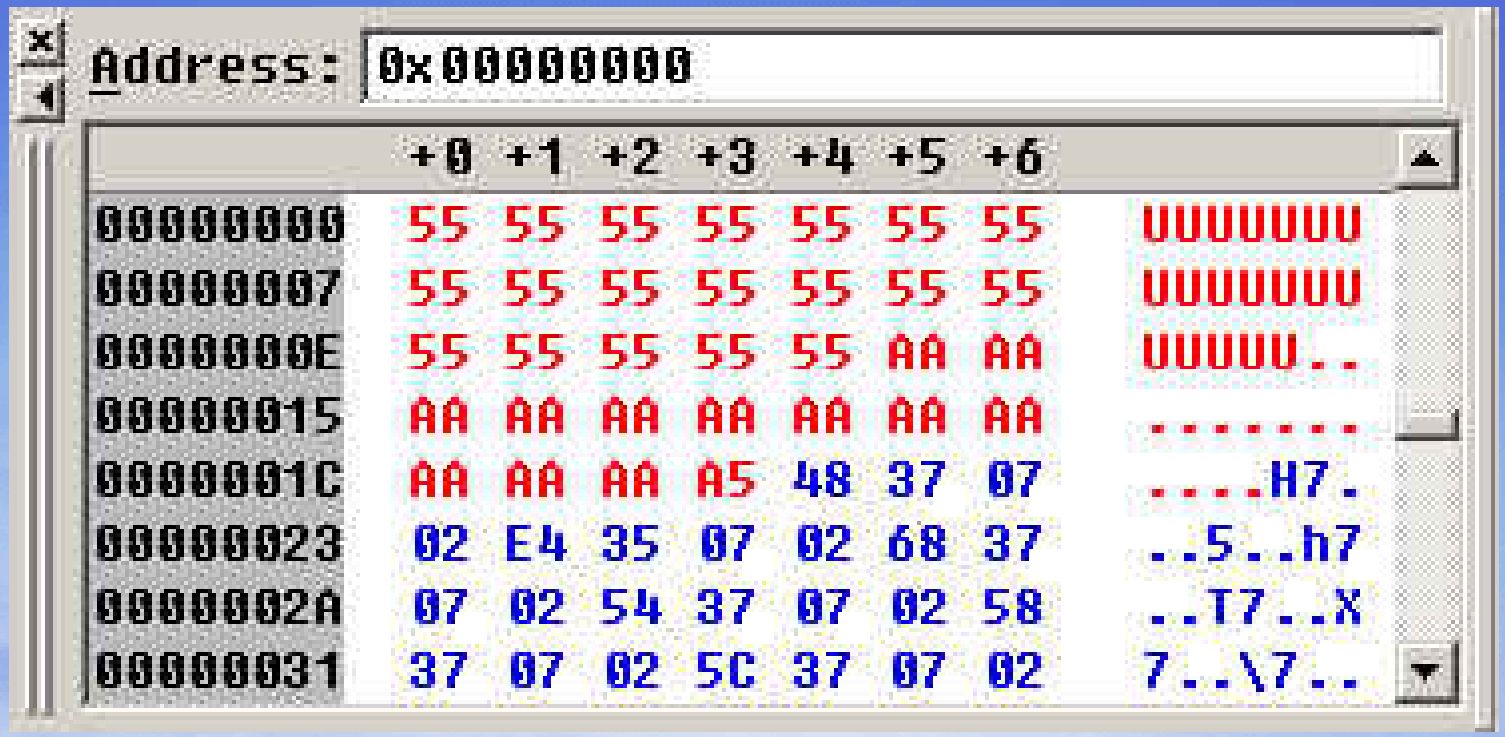
- **RCB: Remap Command Bit**
  0 = No effect.
  1 = Cancels the remapping (performed at reset) of the page zero memory devices.

简单的脚本文件

脚本文件的简单示例如下：

```
;停止目标板
stop
;配置存储器
memwrite    0xffe00004 0x02002121
memwrite    0xffe00024 0x06
;下面是重映射
memwrite    0xffe00020 0x01
refresh
```

尝试访问RAM区


Address: 0x00000000
+0 +1 +2 +3 +4 +5 +6
00000000 55 55 55 55 55 55 55 UUUUUUU
00000007 55 55 55 55 55 55 55 UUUUUUU
0000000E 55 55 55 55 55 AA AA UUUUU..
00000015 AA AA AA AA AA AA AA ....
0000001C AA AA AA A5 48 37 07 ....H7.
00000023 02 E4 35 07 02 68 37 ..5..h7
0000002A 07 02 54 37 07 02 58 ..T7..X
00000031 37 07 02 5C 37 07 02 7..\7..

测试程序

- 编写硬件测试程序
- 确认程序执行正确
- 确认开发工具可以正确监控目标

程序代码

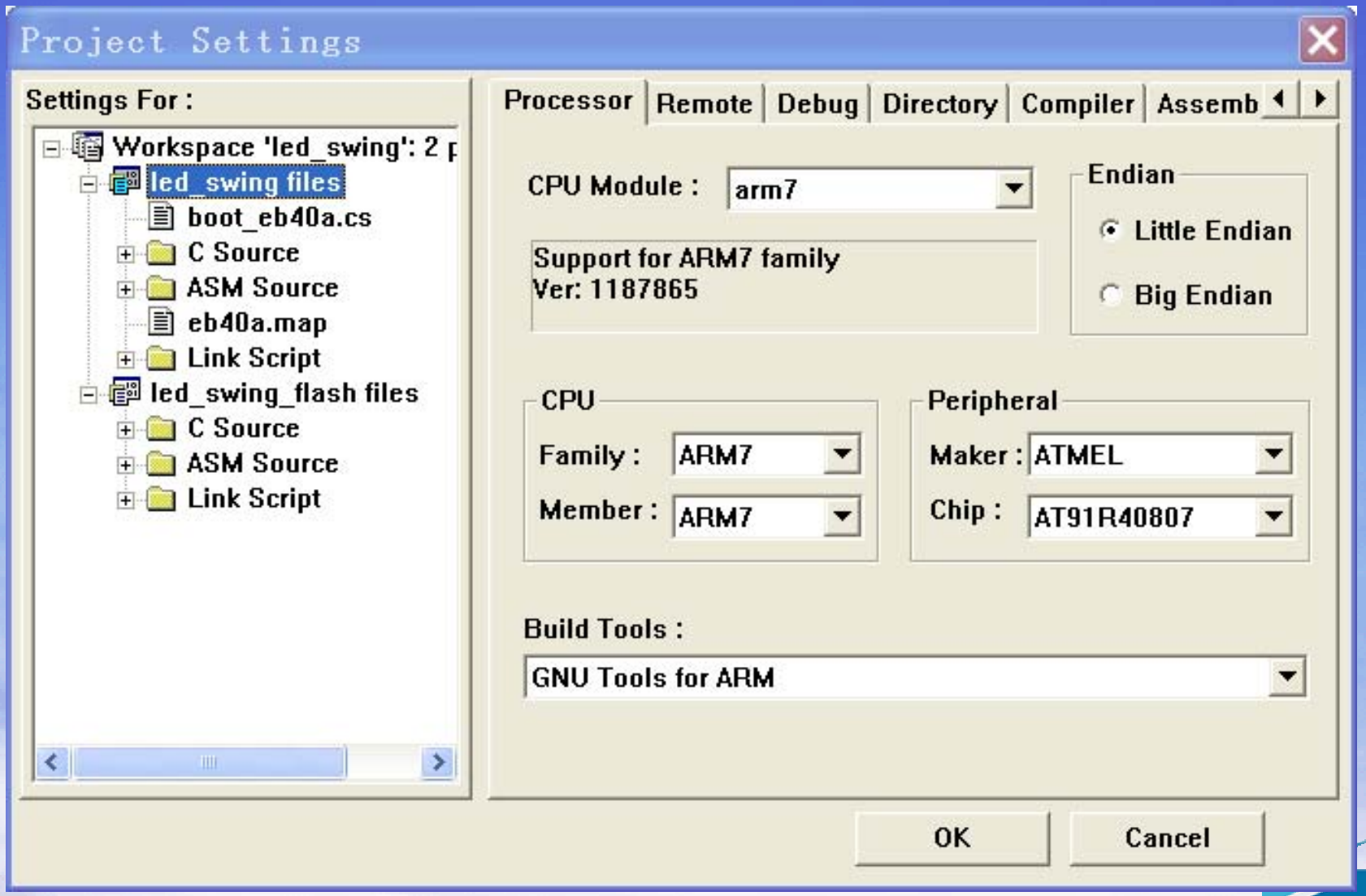
```
.text
        MOV     r0, #10             /* Set up parameters */
        MOV     r1, #3
        ADD     r0, r0, r1          /* r0 = r0 + r1 */

stop:
        MOV     r0, #0x18           /* angel_SWIreason_ReportException */
        LDR     r1, =0x20026        /* ADP_Stopped_ApplicationExit */
        SWI     0x123456            /* Angel semihosting ARM SWI */
.end
```

新建工程

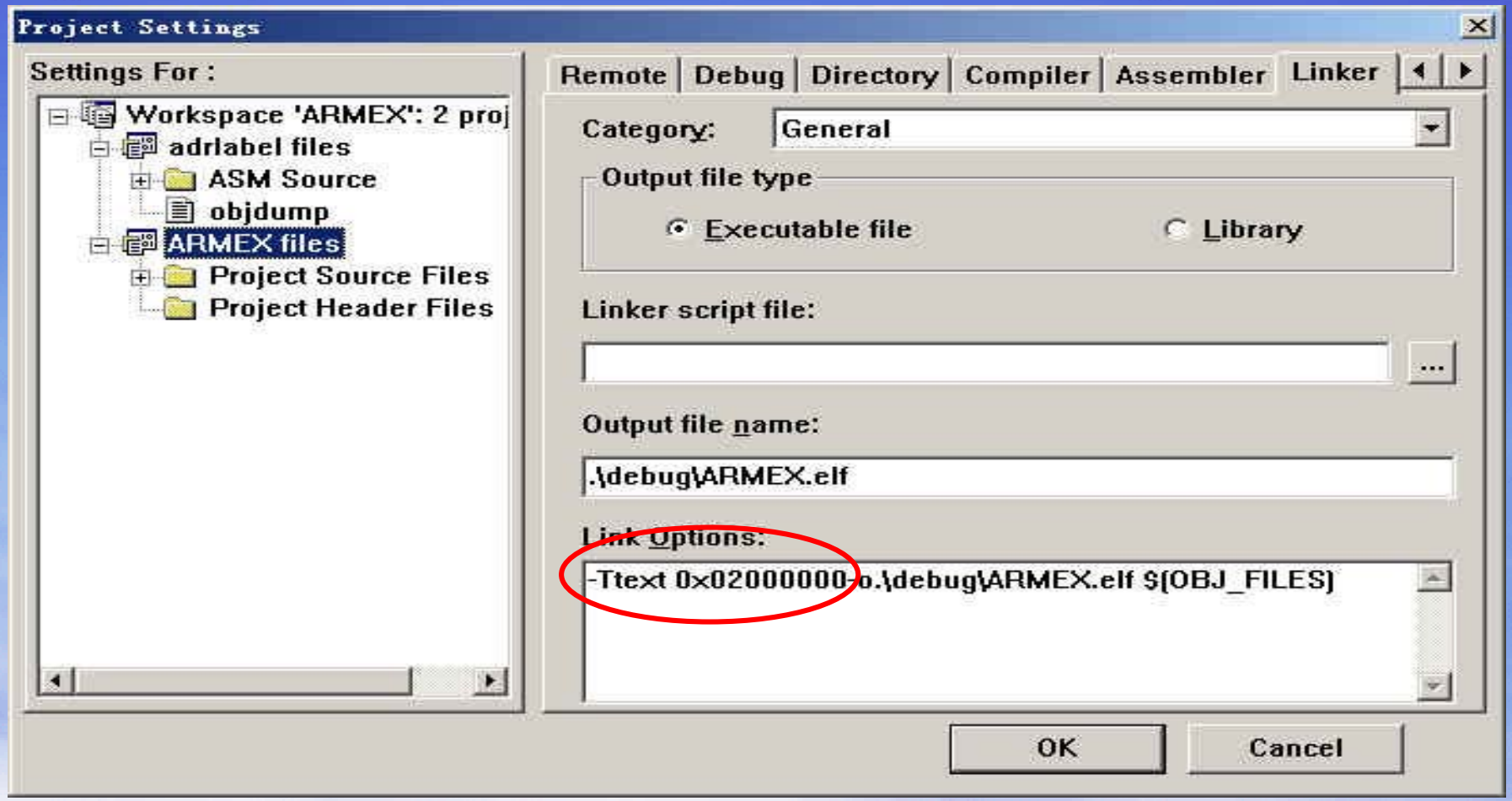

Create a New Project
Project name:
ARMEX
Location:
E:\EmbestIDE\Examples\arm\EXAMPI
...
OK
Cancel

添加文件

工程配置

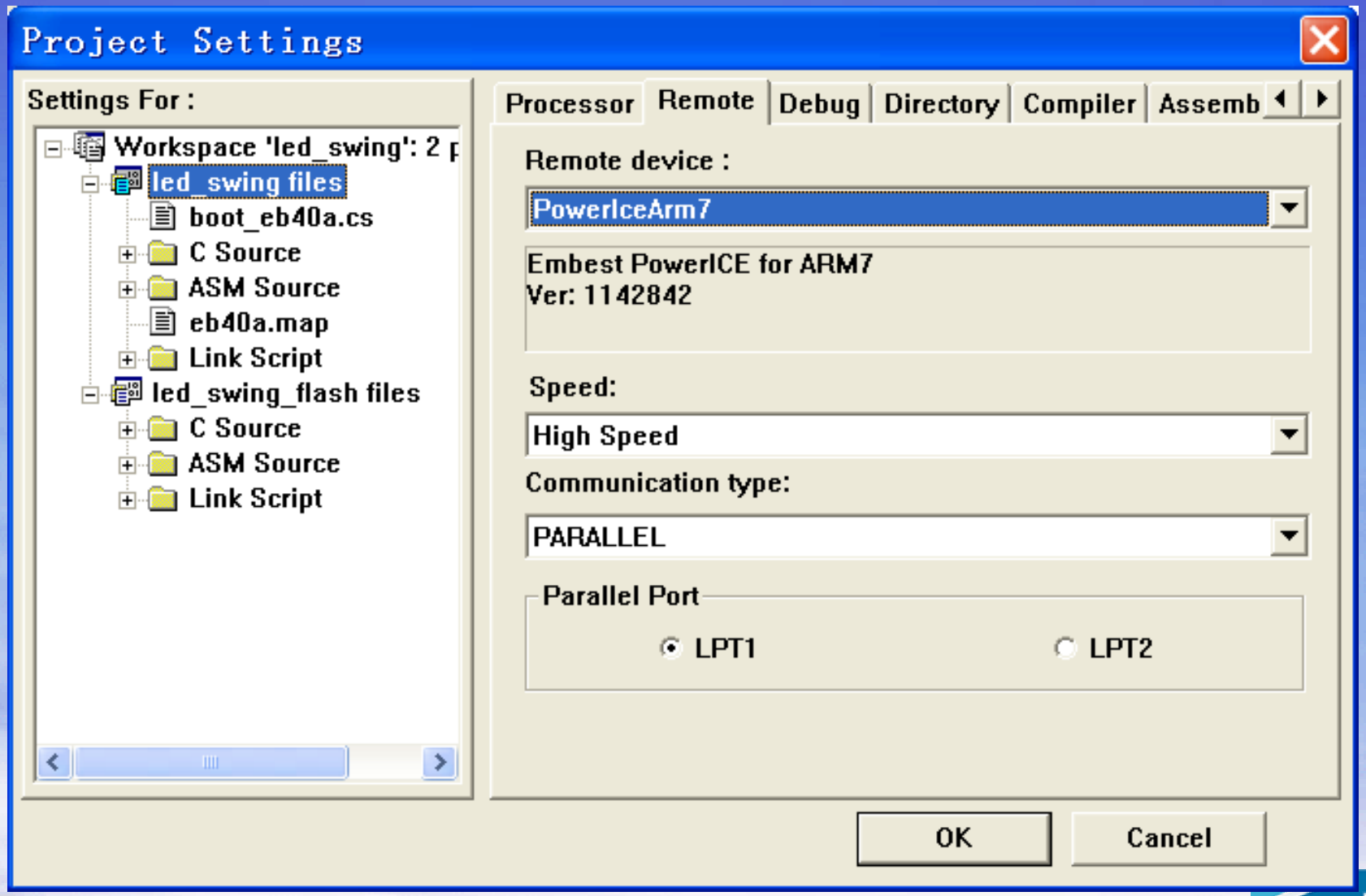

Project Settings
Settings For :
Workspace 'led_swing': 2 p
led_swing files
boot_eb40a.cs
C Source
ASM Source
eb40a.map
Link Script
led_swing_flash files
C Source
ASM Source
Link Script
Processor
Remote
Debug
Directory
Compiler
Assemb
CPU Module :
arm7
Support for ARM7 family
Ver: 1187865
Endian
Little Endian
Big Endian
CPU
Family :
ARM7
Member :
ARM7
Peripheral
Maker :
ATMEL
Chip :
AT91R40807
Build Tools :
GNU Tools for ARM
OK
Cancel

连接配置

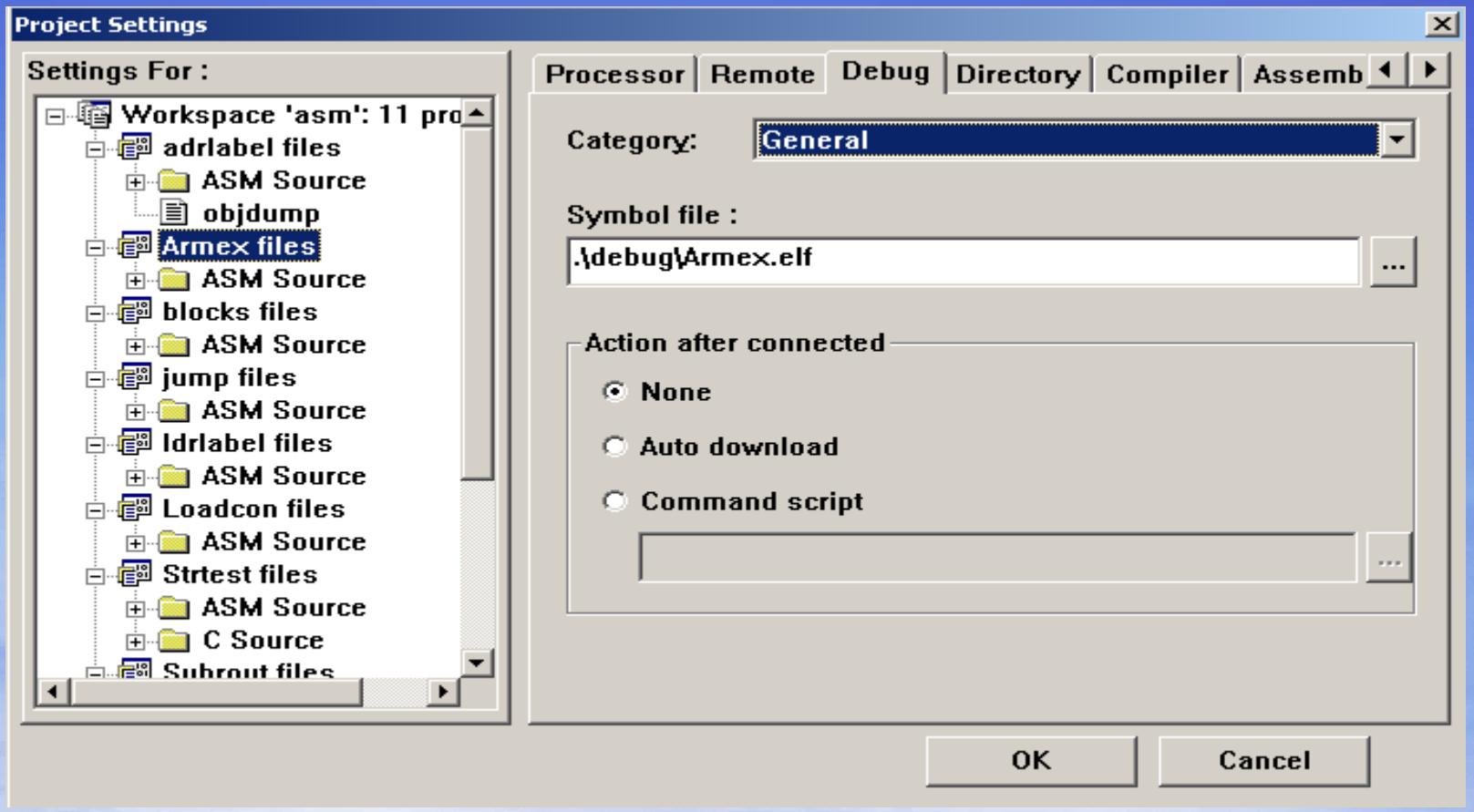

Project Settings
Settings For :
Workspace 'ARMEX': 2 proj
adrlabel files
ASM Source
objdump
ARMEX files
Project Source Files
Project Header Files
Remote
Debug
Directory
Compiler
Assembler
Linker
Category:
General
Output file type
Executable file
Library
Linker script file:
...
Output file name:
.\debug\ARMEX.elf
Link Options:
-Ttext 0x02000000 -o.\debug\ARMEX.elf $(OBJ_FILES)
OK
Cancel

代码编译、链接

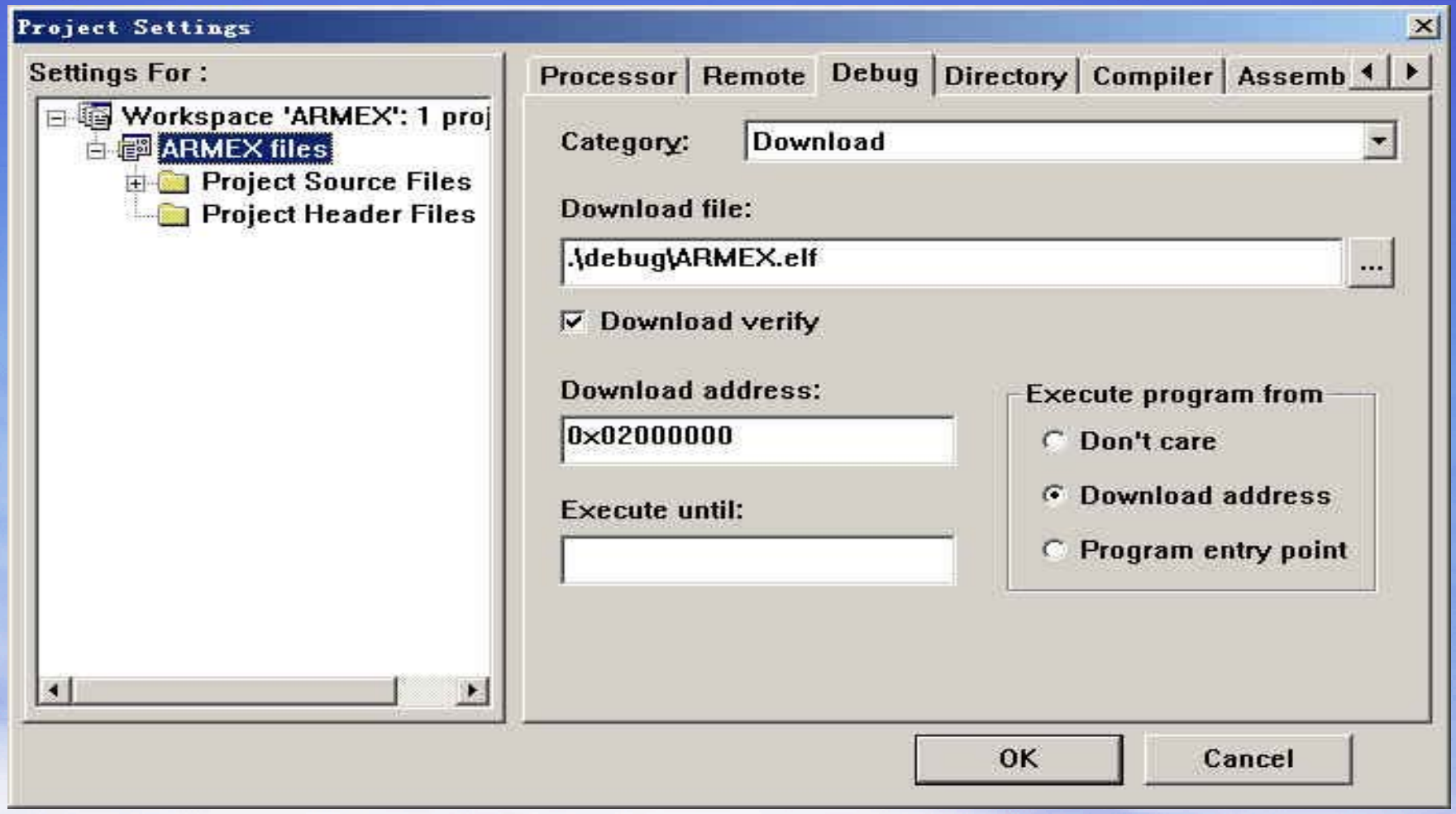
调试模块配置


Project Settings
Settings For :
Workspace 'led_swing': 2 p
led_swing files
boot_eb40a.cs
C Source
ASM Source
eb40a.map
Link Script
led_swing_flash files
C Source
ASM Source
Link Script
Processor
Remote
Debug
Directory
Compiler
Assemb
Remote device :
PowerIceArm7
Embest PowerICE for ARM7
Ver: 1142842
Speed:
High Speed
Communication type:
PARALLEL
Parallel Port
LPT1
LPT2
OK
Cancel

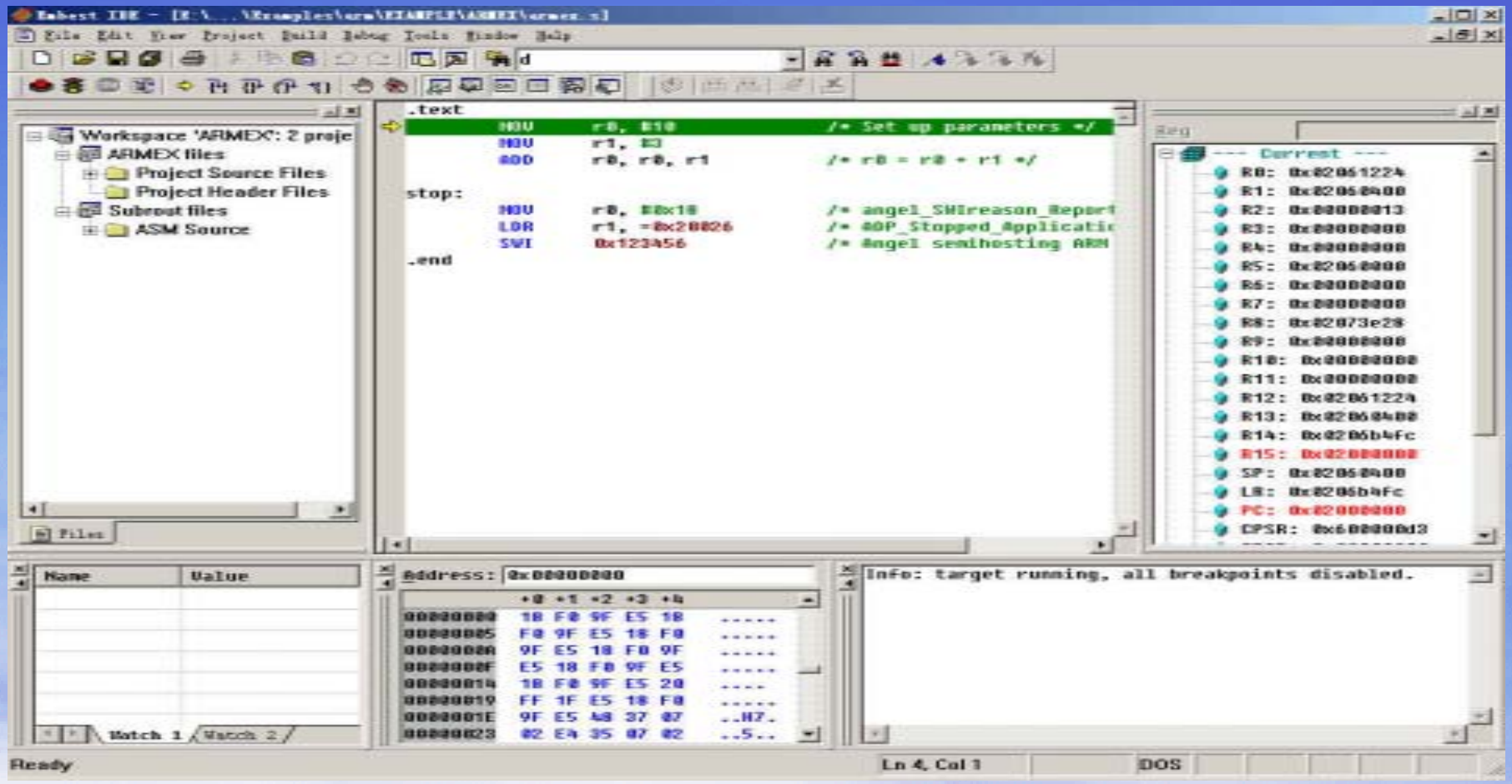
调试配置


Project Settings
Settings For :
Workspace 'asm': 11 pro
adrlabel files
ASM Source
objdump
Armex files
ASM Source
blocks files
ASM Source
jump files
ASM Source
ldrlabel files
ASM Source
Loadcon files
ASM Source
Strtest files
ASM Source
C Source
Subrout files
Processor
Remote
Debug
Directory
Compiler
Assemb
Category:
General
Symbol file :
.\debug\Armex.elf
...
Action after connected
None
Auto download
Command script
...
OK
Cancel

下载配置


Project Settings
Settings For :
Workspace 'ARMEX': 1 proj
ARMEX files
Project Source Files
Project Header Files
Processor
Remote
Debug
Directory
Compiler
Assemb
Category:
Download
Download file:
.\debug\ARMEX.elf
...
Download verify
Download address:
0x02000000
Execute until:
Execute program from
Don't care
Download address
Program entry point
OK
Cancel

调试界面


Workspace 'ARMEX': 2 proje
ARMEX files
Project Source Files
Project Header Files
Subrout files
ASM Source
.text
MOV r0, #10 /* Set up parameters */
MOV r1, #3
ADD r0, r0, r1 /* r0 = r0 + r1 */
stop:
MOV r0, #0x18 /* angel_SWIreason_Report
LDR r1, =0x20026 /* ADP_Stopped_Applicatio
SWI 0x123456 /* Angel semihosting ARM
.end
Current
R0: 0x82061224
R1: 0x82060400
R2: 0x00000013
R3: 0x00000000
R4: 0x00000000
R5: 0x82060000
R6: 0x00000000
R7: 0x00000000
R8: 0x82073e28
R9: 0x00000000
R10: 0x00000000
R11: 0x00000000
R12: 0x82061224
R13: 0x82060400
R14: 0x8206b4fc
R15: 0x82000000
SP: 0x82060400
LR: 0x8206b4fc
PC: 0x82000000
CPSR: 0x600000d3
Name
Value
Watch 1
Watch 2
Address: 0x00000000
+0 +1 +2 +3 +4
00000000 18 F0 9F E5 18 .....
00000005 F0 9F E5 18 F0 .....
0000000A 9F E5 18 F0 9F .....
0000000F E5 18 F0 9F E5 .....
00000014 18 F0 9F E5 20 ....
00000019 FF 1F E5 18 F0 .....
0000001E 9F E5 48 37 07 ..H7.
00000023 02 E4 35 07 02 ..5..
Info: target running, all breakpoints disabled.
Ready
Ln 4, Col 1
DOS

简单链接定位文件

```
SECTIONS
{
    .= 0x10000
    .text:{*(.text)}
    .= 0x8000000
    .data:{*(.data)}
    .bss:{*(.bss)}
}
```

典型链接定位文件

```
SECTIONS
{
        . = 0x02000000;
        .text : { *(.text) }
        Image_RO_Limit = .;
        Image_RW_Base = .;
        .data : { *(.data) }
        .rodata : { *(.rodata) }
        .bss : { *(.bss) }
        PROVIDE (__stack = .);
        end = .;
        _end = .;
        .debug_info     0 : { *(.debug_info)  }
        .debug_line     0 : { *(.debug_line)  }
        .debug_abbrev   0 : { *(.debug_abbrev)}
        .debug_frame    0 : { *(.debug_frame) }
}
```

简单的脚本文件

脚本文件的简单示例如下：

```
;停止目标板
stop
;配置存储器
memwrite    0xffe00000 0x01002535
memwrite    0xffe00004 0x02002122
memwrite    0xffe00024 0x06
;下面是重映射
memwrite    0xffe00020 0x01
refresh
download -v D:\Demo\armdemo\debug\led.bin 0x2000000
```

.MAP文件

.MAP文件如下：

#Name	Start	Size	Attribute
ONCHIPRAM	0	40000	RW
FLASH	1000000	400000	R
SRAM	2000000	80000	RW
PERIREG	FFC00000	400000	RW

AT91的中断处理机制

地址	寄存器
	……
0xFFFFF100	AIC_IVR
0xFFFFF0FC	AIC_SVR31
	……
0xFFFFF080	AIC_SVR0
0xFFFFF07C	AIC_SMR31
	……
0xFFFFF000	AIC_SMR0
	……

中断向量寄存器

当前中断处理函数地址

中断源向量寄存器

中断源模式寄存器

启动代码之flash中断向量表

```
            B       InitReset       /* reset */
undefvec:
            B       undefvec        /* Undefined Instruction */
swivec:
            B       swivec          /* Software Interrupt */
pabtvec:
            B       pabtvec         /* Prefetch Abort */
dabtvec:
            B       dabtvec         /* Data Abort */
rsvdvec:
            B       rsvdvec         /* reserved */
irqvec:
            B       irqvec          /* reserved */
fiqvec:
            B       fiqvec          /* reserved */
```

启动代码之RAM中断向量表

```
VectorTable:
        ldr     pc, [pc, #+0x18]        /* SoftReset                    */
        ldr     pc, [pc, #+0x18]        /* UndefHandler                 */
        ldr     pc, [pc, #+0x18]        /* SWIHandler                   */
        ldr     pc, [pc, #+0x18]        /* PrefetchAbortHandler */
        ldr     pc, [pc, #+0x18]        /* DataAbortHandler     */
        nop                             /* Reserved                     */
        ldr     pc, [pc,#-0xF20]        /* IRQ : read the AIC   */
        ldr     pc, [pc,#-0xF20]        /* FIQ : read the AIC   */

#- There are only 5 offsets as the vectoring is used.
        .long   SoftReset
        .long   UndefHandler
        .long   SWIHandler
        .long   PrefetchAbortHandler
        .long   DataAbortHandler
#- Vectoring Execution function run at absolut addresss
SoftReset:
        b       SoftReset
UndefHandler:
        b       UndefHandler
SWIHandler:
        b       SWIHandler
PrefetchAbortHandler:
        b       PrefetchAbortHandler
DataAbortHandler:
```

启动代码大致流程

- 中断向量表
- 初始化中断向量控制器
- 复制中断向量表
- REMAP
- 初始化各模式栈指针
- 初始化C程序变量
- 转到C入口地址

RAM区程序调试

- 程序各段均在RAM区
- 由ICE直接下载程序
- 下载速度快，断点设置方便
- 可以采用直接修改mem等手段避免一些程序下载
- 执行速度比在 Flash 中快
- 需要较大的RAM区

分块调试技术

- 将程序分成几个逻辑块
- 调试好的逻辑块写入 Flash 中
- 既具有RAM调试的优越性，又减少了程序的下载量，节约RAM，提高调试速度
- 减少了RAM程序一次性转到Flash时可能出现的问题

常见的程序执行方式

- 程序代码在Flash中执行，仅将堆栈、数据段分配到RAM区
- 程序将自己完全拷贝到RAM区
- 程序将部分代码段拷贝到RAM区（较为少用）

常用的 Flash 编程方法

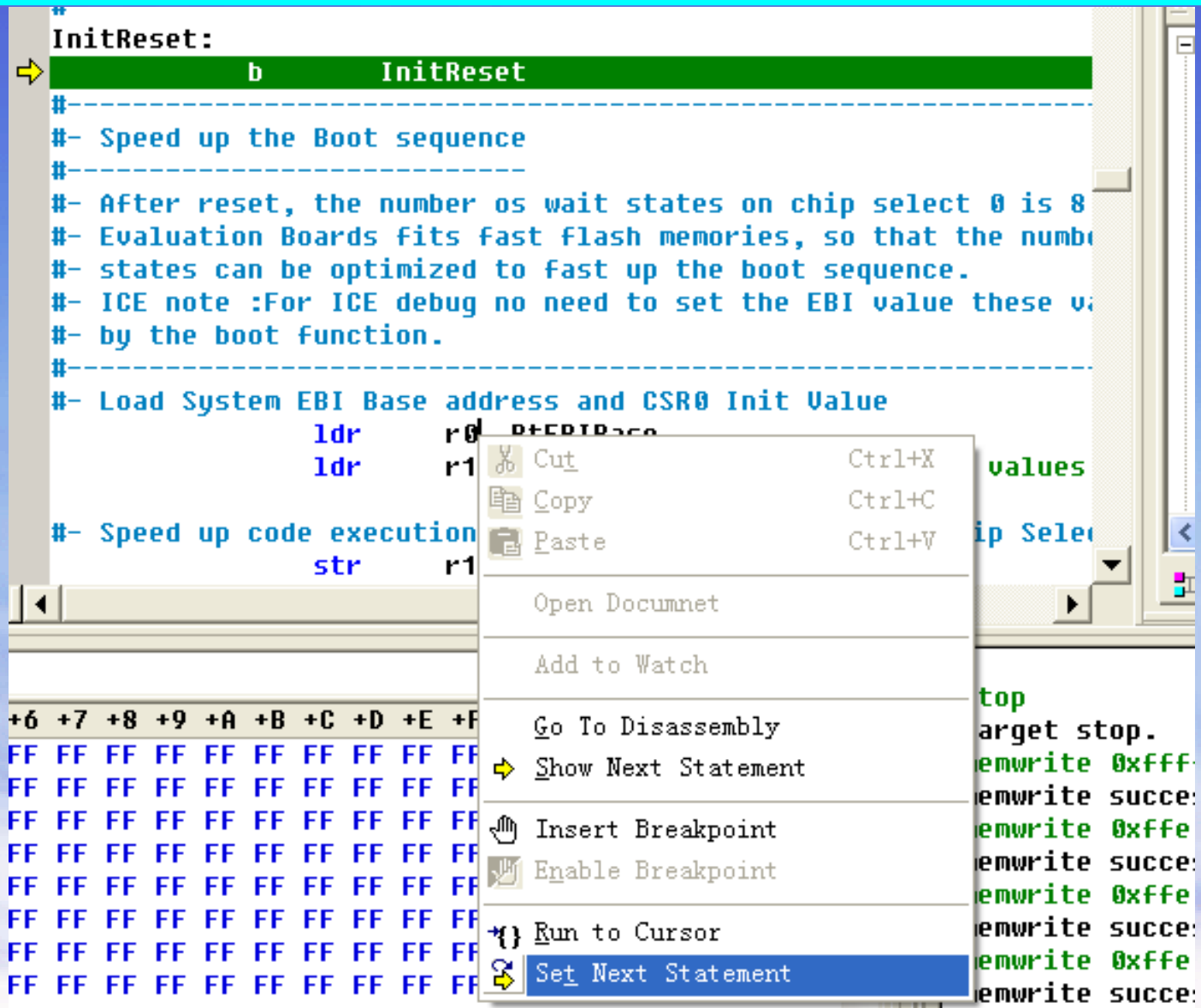
- 专用或通用的编程器
- 操作系统所带 Flash 编程工具
- 监控程序所支持的 Flash 编程工具
- 自编写 RAM 区运行的 Flash 编程工具
- 调试工具/JTAG仿真器支持的编程工具

FLASH程序调试

```
#-------------------------------
#- From here, the code is executed
#- in 0x100 0000.
#-------------------------------
InitReset:

#-------------------------------
#- Speed up the Boot sequence
#---------------------------
```

```
#-------------------------------
#- From here, the code is executed
#- in 0x100 0000.
#-------------------------------
InitReset:
            b       InitReset
#-------------------------------
#- Speed up the Boot sequence
#---------------------------
```

FLASH程序调试


InitReset:
b InitReset
#- Speed up the Boot sequence
#- After reset, the number os wait states on chip select 0 is 8
#- Evaluation Boards fits fast flash memories, so that the numb
#- states can be optimized to fast up the boot sequence.
#- ICE note :For ICE debug no need to set the EBI value these v
#- by the boot function.
#- Load System EBI Base address and CSR0 Init Value
ldr r1
values
#- Speed up code execution
str r1
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Open Documnet
Add to Watch
Go To Disassembly
Show Next Statement
Insert Breakpoint
Enable Breakpoint
Run to Cursor
Set Next Statement
+6 +7 +8 +9 +A +B +C +D +E
FF FF FF FF FF FF FF FF FF FF
top
arget stop.
emwrite 0xfff
emwrite succe
emwrite 0xffe
emwrite succe
emwrite 0xffe
emwrite succe
emwrite 0xffe
emwrite succe