
JMY980 核心板与 JMY901 读写板用户手册

(Revision 1.00)

北京金木雨电子有限公司

2012/6/4



在使用本产品前请仔细阅读本说明书，如果有任何疑问，请联系我们，我们会给您详尽的解答



目录

1 简介.....	2
2 接口与地址分配说明.....	2
2.1 管脚说明.....	2
2.2 地址空间分配和片选信号定义.....	4
3 程序烧写与系统下载.....	5
3.1 bootloader 烧写.....	5
3.1.1 烧写 Nor Flash 软件安装.....	6
3.1.2 ARM9 NOR Flash 烧写流程.....	8
3.2 下载操作系统.....	12
3.2.1 下载系统前准备工作.....	12
3.2.2 下载 Linux 系统.....	13
3.2.3 下载 WindowsCE 系统.....	19
4 WindowsCE 6.0 开发指南.....	21
4.1 建立 WindowsCE 6.0 开发环境.....	21
4.1.1 安装 Visual Studio 2005 及补丁.....	22
4.1.2 安装 Windows CE 6.0 及补丁.....	29
4.1.3 安装 BSP 及内核工程示例.....	38
4.1.4 各个驱动程序源代码的位置.....	40
4.2 配置和编译 Windows CE 6.0 内核及 Bootloader.....	41
4.2.1 编译缺省内核工程示例.....	41
4.2.2 编译和烧写 Bootloader 之 NBOOT.....	44
4.2.3 在 BSP 中修改 LCD 类型及串口输出功能.....	46
4.2.4 制作和修改 Windows CE 启动 Logo.....	47
4.2.5 创建 SDK.....	49
4.2.6 安装 SDK.....	51
4.3 与 PC 同步.....	54
4.3.1 安装同步驱动与软件.....	54
4.4 通过 VS2005 创建应用程序，并编译下载到开发板运行.....	60
4.4.1 创建项目.....	60



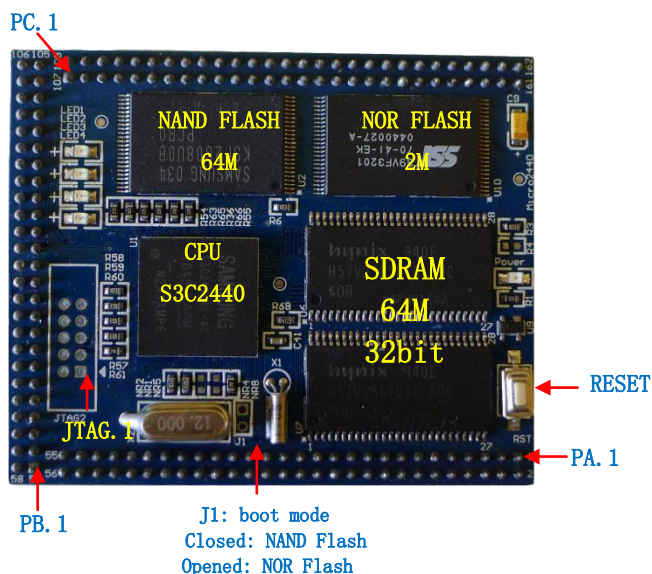
1 简介

JMY980 是一个最小系统板，具有最基本的系统配置：

- CPU: 三星 S3C2440, 主频 400MHz;
- NOR FLASH: 2MByte, 掉电非易失;
- NAND FLASH: 256MByte, 掉电非易失;
- SDRAM: 64MByte, 由 2 片 16bit 宽度的 32MByte SDRAM 组成, 时钟频率高达 100MHz;
- 系统时钟源: 12M 无源晶振;
- 实时时钟: 内部实时时钟 (需另接备份锂电池);
- 系统供电: +5V;
- 支持系统: Linux2.6.32/WindowsCE6.0
uCos2/2440test (裸机测试程序);

尺寸: 63×52mm;

- 1 个 56Pin 2.0mm 间距 GPIO 接口 PA;
- 1 个 50Pin 2.0mm 间距 LCD、CMOS CAMERA 接口 PB;
- 1 个 56Pin 2.0mm 间距系统总线接口 PC;
- 10Pin 2.0mm 间距 JTAG 接口;
- 1 个电源指示灯和 4 个用户指示灯;
- 在板 JTAG, 专业电压调节芯片, 用户只要接上 5V 电源即可做简单调试开发了。



2 接口与地址分配说明

2.1 管脚说明

端口 PA	网络名称	说明(有些端口可复用)	端口 PA	网络名称	说明(有些端口可复用)
PA1	VDD5V	5V 电源	PA2	GND	地
PA3	EINT19	EINT19/GPG11	PA4	EINT18	EINT18/GPG10/nCTS1
PA5	EINT17	EINT17/GPG9/nRST1	PA6	EINT16	EINT16/GPG8
PA7	EINT15	EINT15/GPG7/SPICLK1	PA8	EINT14	EINT14/GPG6/SPIMOSI1
PA9	EINT13	EINT13/GPG5/SPIMISO1	PA10	EINT11	EINT11/GPG3/nSS1
PA11	EINT8	EINT8/GPG0	PA12	EINT6	EINT6/GPF6
PA13	EINT5	EINT5/GPF5	PA14	EINT4	EINT4/GPF4
PA15	EINT3	EINT3/GPF3	PA16	EINT2	EINT2/GPF2



PA17	EINT1	EINT1/GPF1	PA18	EINT0	EINT0/GPF0
PA19	WP_SD	WP_SD/GPH8	PA20	SDCLK	SDCLK/GPE5
PA21	SDCMD	SDCMD/GPE6	PA22	SDDATA2	SDDATA2/GPE9
PA23	SDDATA3	SDDATA3/GPE10	PA24	SDDATA0	SDDATA0/GPE7
PA25	SDDATA1	SDDATA1/GPE8	PA26	LCDVF2	OM0 (NOR-NAND Select)
PA27	LCDVF0	LCDVF0/GPC5, Used for USB_EN	PA28	M_nRESET	手动复位信号 (低电平有效)
PA29	DN1	DN1/PDN0, USB Slave' s D-	PA30	DP1	DP1/PDP0, USB Slave' s D+
PA31	DNO	DNO, USB Host' s D-	PA32	DP0	DP0, USB Host' s D+
PA33	AIN2	AIN2	PA34	VDDRTC	RTC 电源输入 (1.8V)
PA35	AIN0	AIN0	PA36	AIN1	AIN1
PA37	L3MODE	L3MODE/TOUT2/GPB2	PA38	L3DATA	L3DATA/TOUT3/GPB3
PA39	L3CLOCK	L3LOCK/TCLK0/GPB4	PA40	I2SLRCK	I2SLRCK/GPE0
PA41	I2SSCLK	I2SSCLK/GPE1	PA42	CDCLK	CDCLK/GPE2
PA43	I2SSDI	I2SSDI/GPE3	PA44	I2SSDO	I2SSDO/GPE4
PA45	GPB0	TOUT0/GPB0	PA46	GPB1	TOUT1/GPB1
PA47	TXD2	TXD2/nRTS1/GPH6	PA48	RXD2	RXD2/nCTS1/GPH7
PA49	TXD1	TXD1/GPH4	PA50	RXD1	RXD1/GPH5
PA51	TXD0	TXD0/GPH2	PA52	RXD0	RXD0/GPH3
PA53	nCTS0	nCTS0/GPH0	PA54	nRTS0	nRTS0/GPH1
PA55	I2CSDA	I2CSDA/GPE15	PA56	I2CSCL	I2CSCL/GPE14

端口 PB	网络名称	说明 (有些端口可复用)	端口 PA	网络名称	说明 (有些端口可复用)
PB1	TSYM		PB2	TSYP	
PB3	TSXM		PB4	TSYM	
PB5	VD22	VD22/GPD14	PB6	VD23	VD23/GPD15
PB7	VD20	VD20/GPD12	PB8	VD21	VD21/GPD13
PB9	VD18	VD18/GPD10	PB10	VD19	VD19/GPD11
PB11	VD16	VD16/GPD8	PB12	VD17	VD17/GPD9
PB13	VD14	VD14/GPD6	PB14	VD15	VD15/GPD7
PB15	VD12	VD12/GPD4	PB16	VD13	VD13/GPD5
PB17	VD10	VD10/GPD2	PB18	VD11	VD11/GPD3
PB19	VD8	VD8/GPD0	PB20	VD9	VD9/GPD1
PB21	VD6	VD6/GPC14	PB22	VD7	VD7/GPC15
PB23	VD4	VD4/GPC12	PB24	VD5	VD5/GPC13
PB25	VD2	VD2/GPC10	PB26	VD3	VD3/GPC11
PB27	VDO	VDO/GPC8	PB28	VD1	VD1/GPC9
PB29	LCD_PWR	LCD_PWR/EINT12/GPG4	PB30	VM	VM/VDEN/GPC4
PB31	VFRAME	VFRAME/VSYNC/GPC3	PB32	VLINE	VLINE/HSYNC/GPC2
PB33	VCLK	VCLK/GPC1	PB34	LEND	LEND/GPC0
PB35	CAMDATA7	CAMDATA7/GPJ7	PB36	CAMDATA6	CAMDATA6/GPJ6
PB37	CAMDATA5	CAMDATA5/GPJ5	PB38	CAMDATA4	CAMDATA4/GPJ4
PB39	CAMDATA3	CAMDATA3/GPJ3	PB40	CAMDATA2	CAMDATA2/GPJ2



PB41	CAMDATA1	CAMDATA1/GPJ1	PB42	CAMDATA0	CAMDATA0/GPJ0
PB43	CAMCLK	CAMCLK/GPJ11	PB44	CAM_PCLK	CAM_PCLK/GPJ8
PB45	CAM_VSYNC	CAM_VSYNC/GPJ9	PB46	CAM_HREF	CAM_HREF/GPJ10
PB47	EINT20	EINT20/GPG12	PB48	CAMRST	CAMRESET/GPJ12
PB49	VDD5V	VDD5V	PB50	GND	GND

端口 PC	网络名称	说明(有些端口可复用)	端口 PA	网络名称	说明(有些端口可复用)
PC1	EINT7	EINT7/GPF7	PC2	EINT9	EINT9/GPG1
PC3	LnGCS1	片选 LnGCS1	PC4	LnGCS3	片选 LnGCS3
PC5	LnGCS2	片选 LnGCS2	PC6	LnWBE1	LnWBE1
PC7	LnGCS4	片选 LnGCS4	PC8	LnWE	LnWE
PC9	LnOE	LnOE	PC10	nRESET	nRESET
PC11	nWAIT	nWAIT	PC12	nXDACK0	nXDACK0
PC13	LADDR0	地址线 0	PC14	nXDREQ0	nXDREQ0
PC15	LADDR1	地址线 1	PC16	LADDR2	地址线 2
PC17	LADDR3	地址线 3	PC18	LADDR4	地址线 4
PC19	LADDR5	地址线 5	PC20	LADDR6	地址线 6
PC21	LADDR7	地址线 7	PC22	LADDR8	地址线 8
PC23	LADDR9	地址线 9	PC24	LADDR10	地址线 10
PC25	LADDR11	地址线 11	PC26	LADDR12	地址线 12
PC27	LADDR13	地址线 13	PC28	LADDR14	地址线 14
PC29	LADDR15	地址线 15	PC30	LADDR16	地址线 16
PC31	LADDR17	地址线 17	PC32	LADDR18	地址线 18
PC33	LADDR19	地址线 19	PC34	LADDR20	地址线 20
PC35	LADDR21	地址线 21	PC36	LADDR22	地址线 22
PC37	LADDR23	地址线 23	PC38	LADDR24	地址线 24
PC39	LDATA0	数据线 0	PC40	LDATA1	数据线 1
PC41	LDATA2	数据线 2	PC42	LDATA3	数据线 3
PC43	LDATA4	数据线 4	PC44	LDATA5	数据线 5
PC45	LDATA6	数据线 6	PC46	LDATA7	数据线 7
PC47	LDATA8	数据线 8	PC48	LDATA9	数据线 9
PC49	LDATA10	数据线 10	PC50	LDATA11	数据线 11
PC51	LDATA12	数据线 12	PC52	LDATA13	数据线 13
PC53	LDATA14	数据线 14	PC54	LDATA15	数据线 15
PC55	VDD5V	电源 5V	PC56	GND	地

2.2 地址空间分配和片选信号定义

S3C2440 支持两种启动模式：Nand Flash 和 Nor Flash 启动。

两种启动模式下，各个片选的存储空间分配是不同的，如下图：

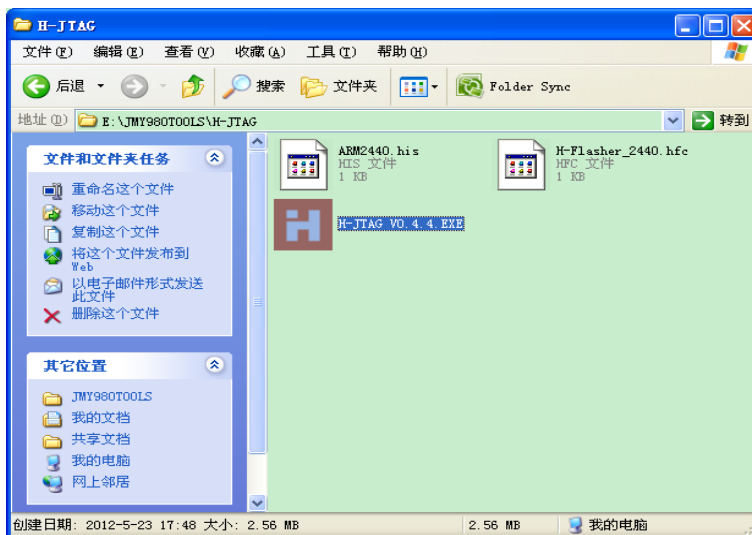


3.1.1 烧写 Nor Flash 软件安装

H-JTAG 软件安装要求：计算机必须有并口。(该软件安装仅仅在第一次使用的情况下，若已安装，该步骤省略)

1、安装 H-JTAG

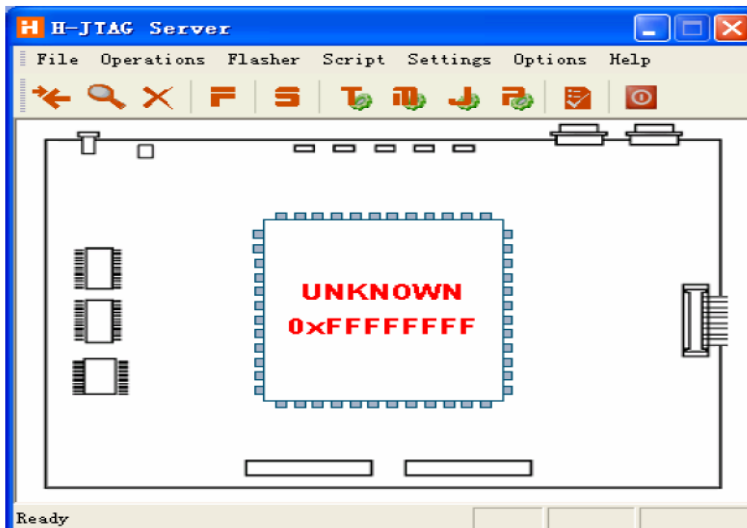
H-JTAG 安装文件位于光盘的“JMY980TOOLS\H-JTAG”目录，双击运行，按照其提示安装即可。



安装完会在桌面生成 H-JTAG 和 H-Flasher 快捷方式，双击运行 H-JTAG，程序将自动检测是否连接了 JTAG 设备，因为之前我们还没有做任何设置，所以会跳出一个提示窗口：



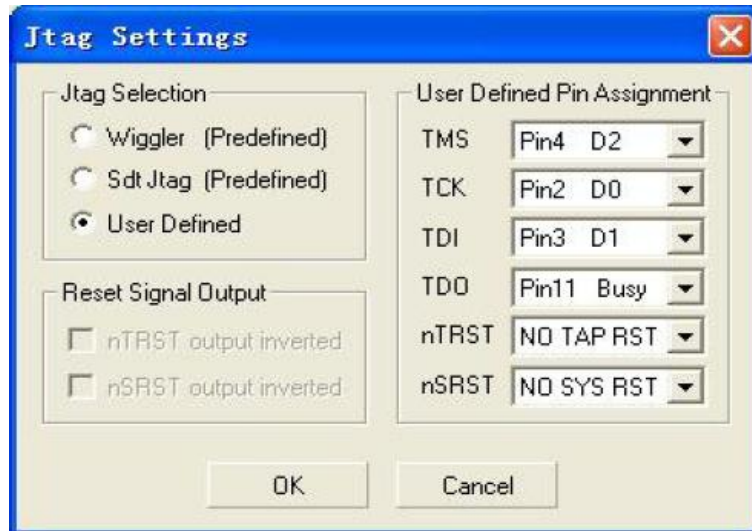
点击确定，进入程序主界面，因为没有连接任何目标器件，因此显示如图所示：





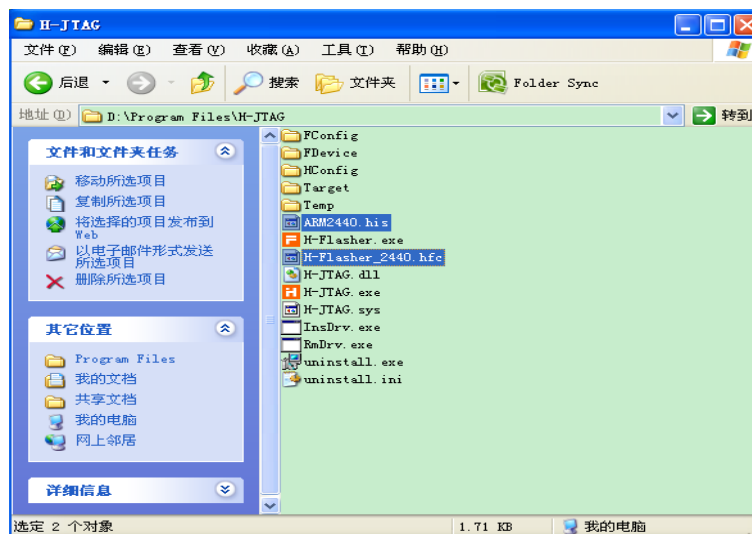
2、设置 JTAG 端口

在 H-JTAG 主界面的菜单里点 Setting->Jtag Setting, 做如下图所示设置, 点 OK 返回主界面。

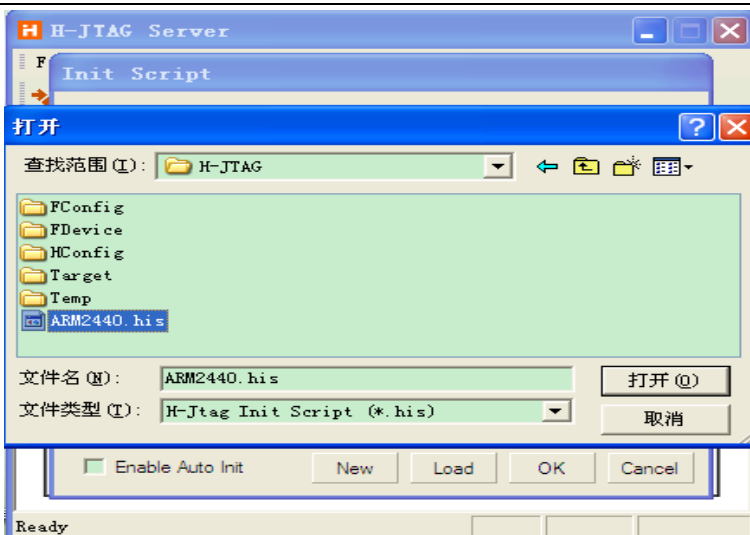


3、设置初始化脚本

把光盘“JMY980TOOLS\H-JTAG”目录中的 ARM2440.his 和 H-Flasher_2440.hfc 文件复制到 H-JTAG 的安装目录, 如图:



在 H-JTAG 的主界面, 点 Script->Init Script, 弹出 Init Script 窗口, 点该窗口下面的 Load 按钮, 找到并选择打开刚刚复制的 ARM2440.his 文件, 如图:



这时，Init Script 窗口会被载入的脚本填充，如图，注意不要点选“Enable Auto Init”，点 OK 退回 H-JTAG 主界面。

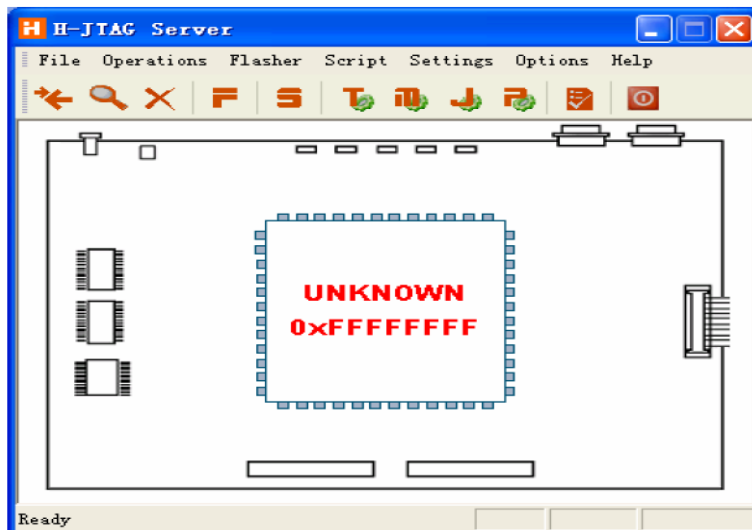
3.1.2 ARM9 NOR Flash 烧写流程

1、检查烧写工具

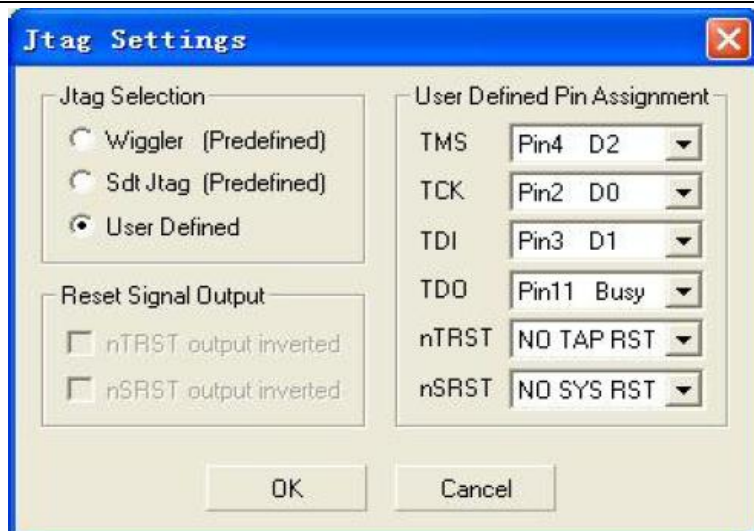
- (1) 带串口，并且安装了 H-JTAG 软件电脑一台。
- (2) NOR Flash 串口线一条。
- (3) JMY901 开发板或自带开发板一个。

2、配置 H-JTAG 软件

打开软件如图：



在 H-JTAG 主界面的菜单里点 Setting->Jtag Settings，做如下图配置：

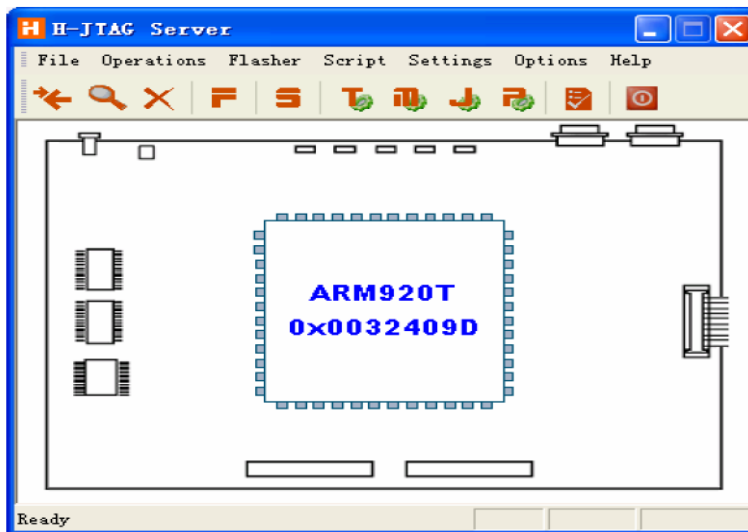


3、连接设备

- (1) 给核心板连接好+5V 电源线，不供电。
- (2) 并口线连接核心板和电脑。
- (3) 确认 JMY901 拨动开关 S2 位于 NOR 端。
- (4) 打开电源。

4、检查设备连接是否正常

点 Operations->Detect Target，出现如下界面，说明已经连接上：

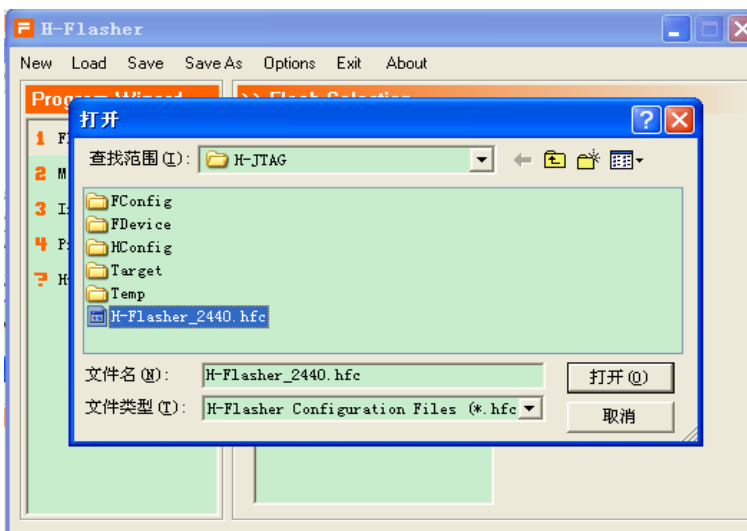


5、装载 H-Flasher_2440.hfc

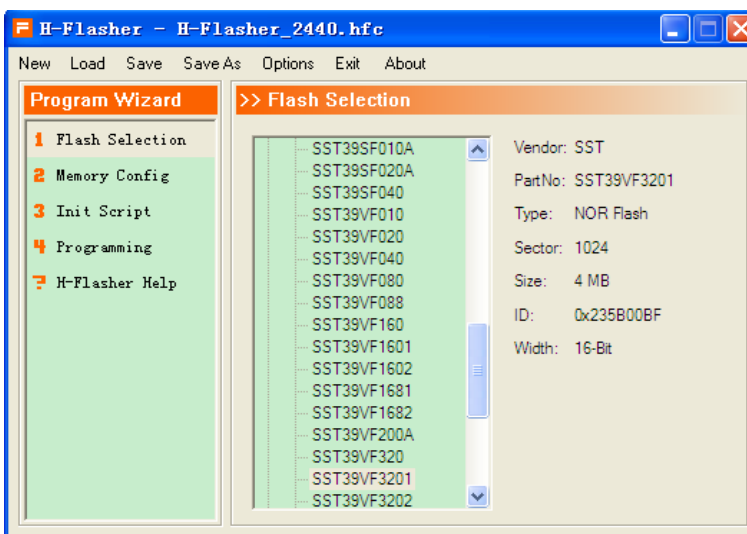
点 Flasher->Start H-Flasher，出现如下 H-Flash 界面：



点 H-Flash 界面中 Load，装入 H-Flasher_2440.hfc:



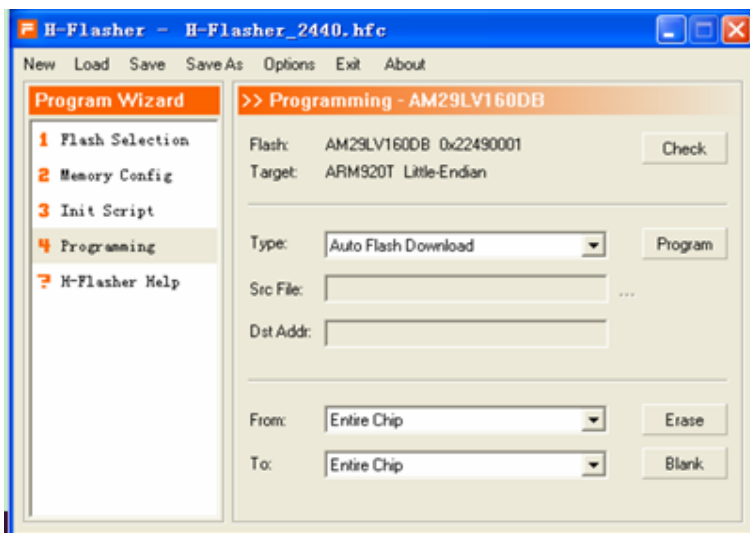
载入后，会出现如下界面，选择 SST39VF3201:



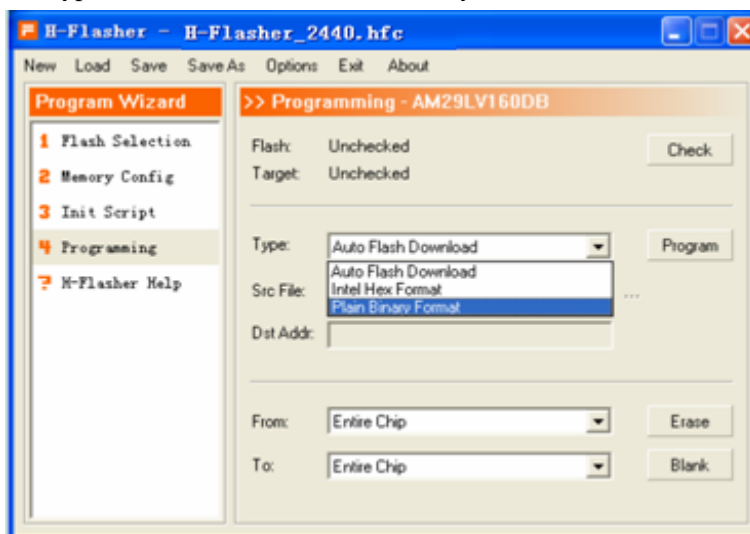
- 6、设置烧写参数
 - (1) 点击 4Programming



(2) 点击 Check 按钮，假如核心板子没有问题，显示如下界面：



(3) 点 Type 下拉列表，选择“Plain Binary Format”：



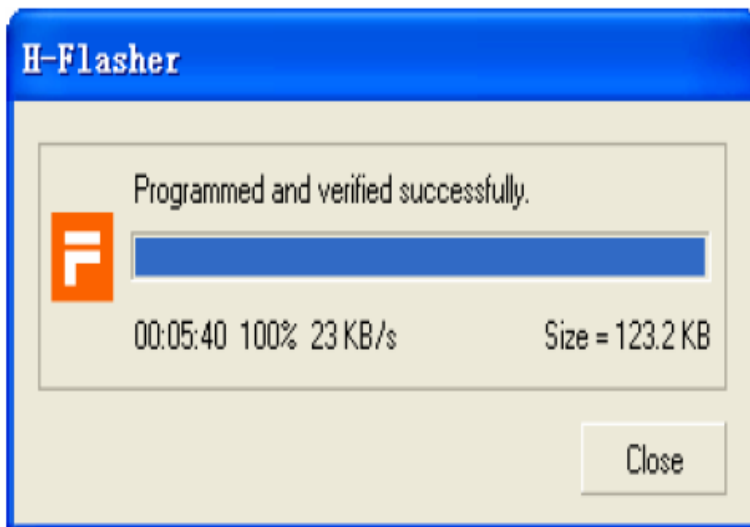
(4) 再点 Src File 右侧的浏览按钮...，选择所要烧写的文件 supervivi (JMY980TOOLS\images\supervivi-128M)。

(5) 并在 Dst Addr 一栏中输入 0。



7、烧写

点 Program 烧写，烧写成功画面如下图：



3.2 下载操作系统

3.2.1 下载系统前准备工作

1、检查下载工具

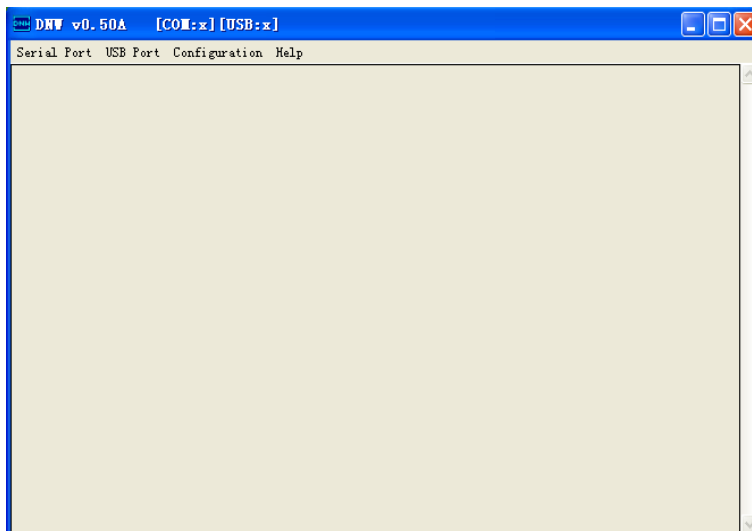
- (1) 带 USB 及串口电脑一台。
- (2) USB、串口电缆各一条。
- (3) JMY980 核心板一个。
- (4) 开发底板一个（自备或使用 JMY901）。
- (5) DNW 软件和 115200.ht 超级终端软件（这 2 个软件无需安装，直接复制到硬盘即可运行）。
- (6) 安装 USB 下载驱动（JMY980TOOLS\usb 下载驱动）。
- (7) JMY901 拨动开关 S2 处于 NOR 端（NOR Flash 启动模式）。

2、打开软件

- (1) 打开 115200.ht 超级终端软件，如下图：

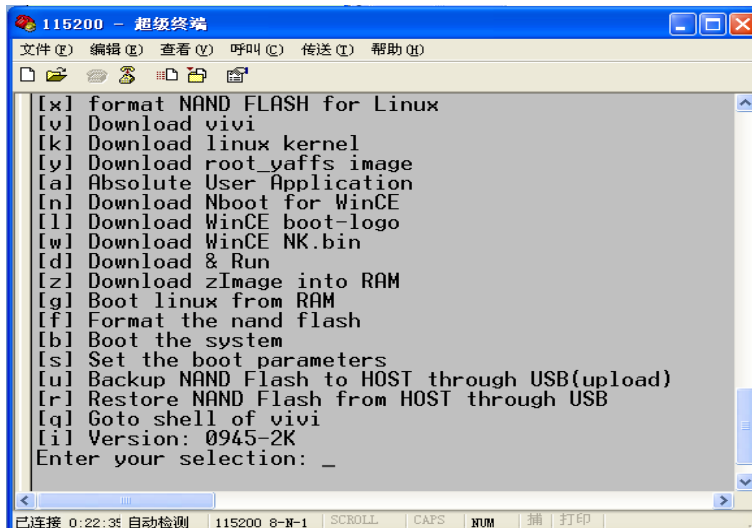


(2) 打开 DNW 软件，如下图：



3.2.2 下载 Linux 系统

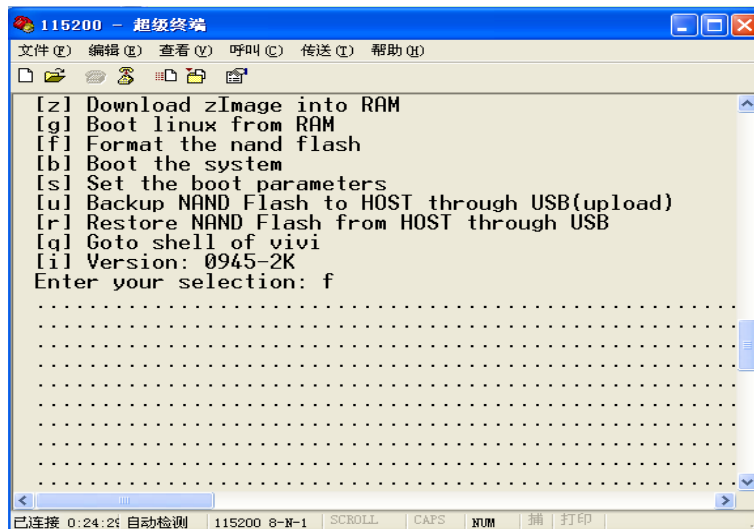
1、确保串口和 USB Slave 口已连接好，上电后 115200.sh 软件会出现如下图所示信息：



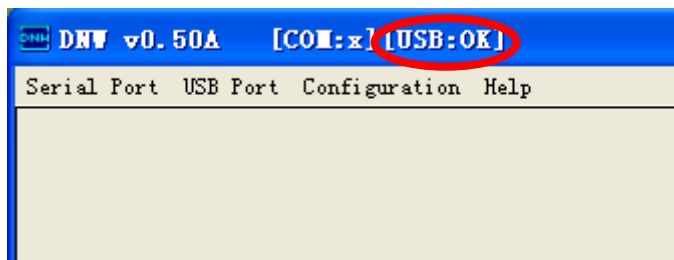


2、格式化 Nand Flash

选择功能键[f], 开始对 Nand Flash 进行分区, 如下图:

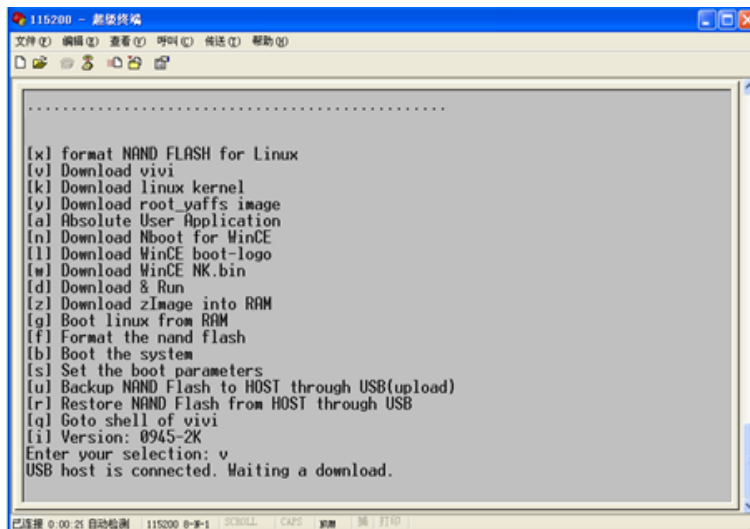


3、查看 DNW 软件中 “USB: OK”, 如下图:

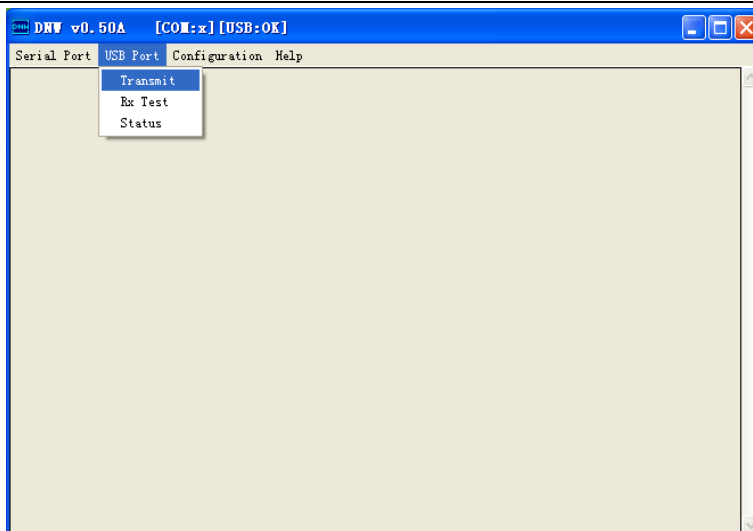


4、安装 bootloader

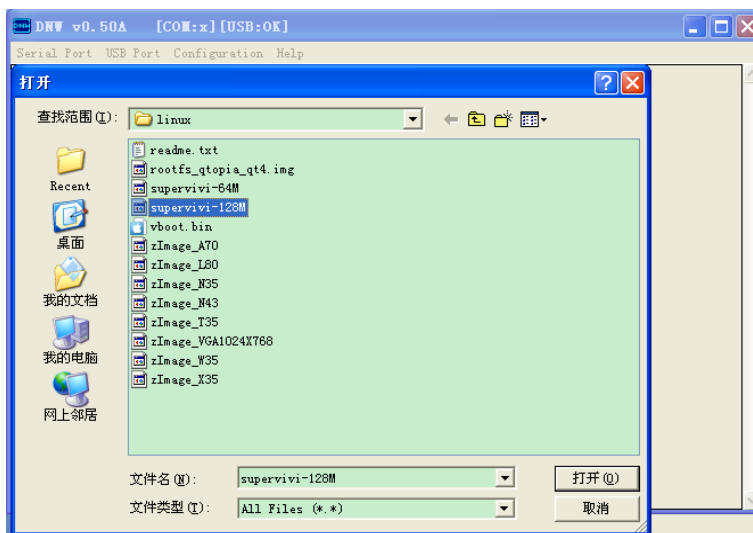
选择功能键[v], 如下图:



点击 DNW 软件的 “USB Port->Transmit->Transmit”, 如下图:

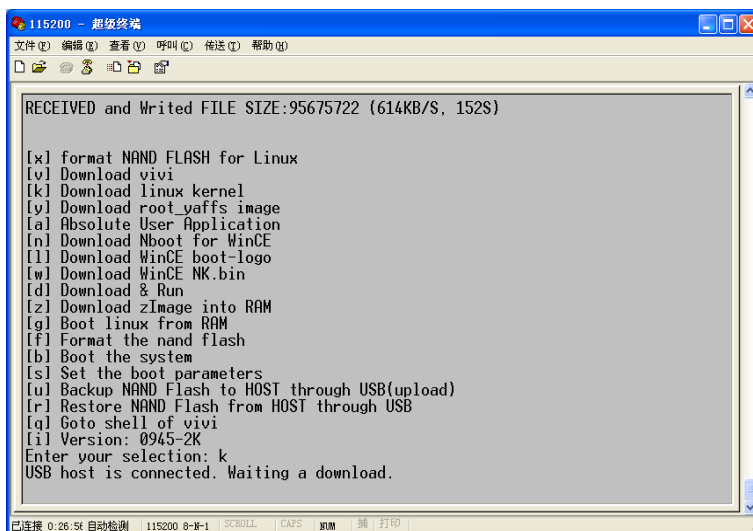


选择 supervivi-128M，点打开，如下图：

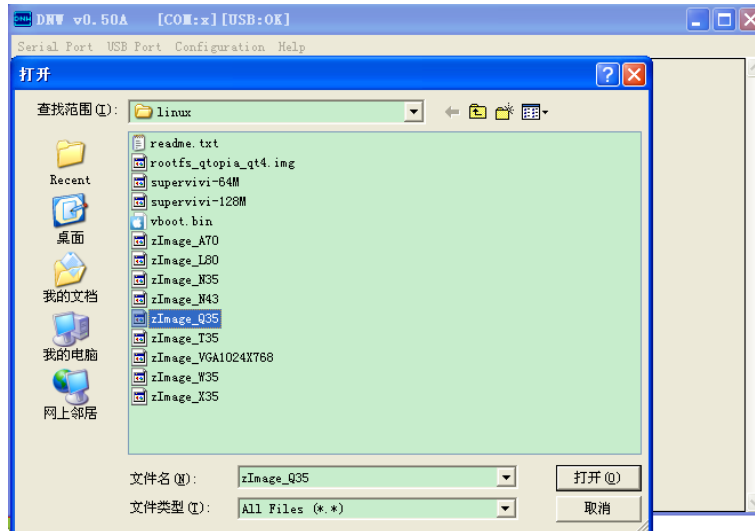


5、安装 Linux 内核

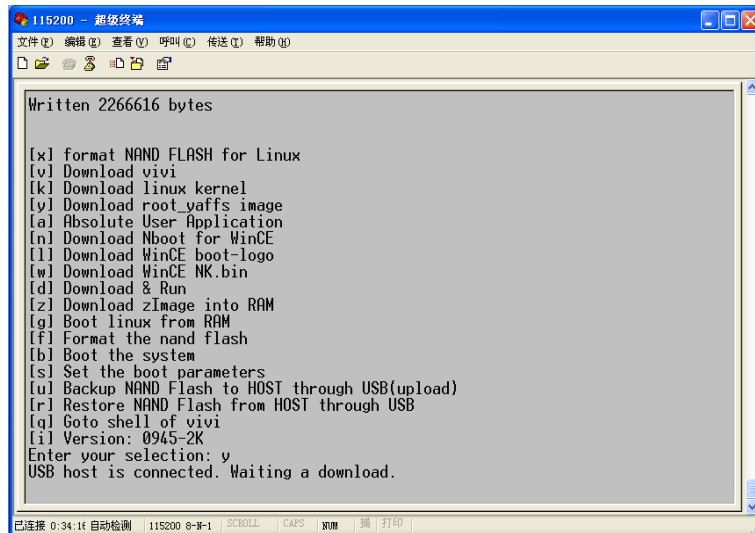
选择功能键[k]，如下图：



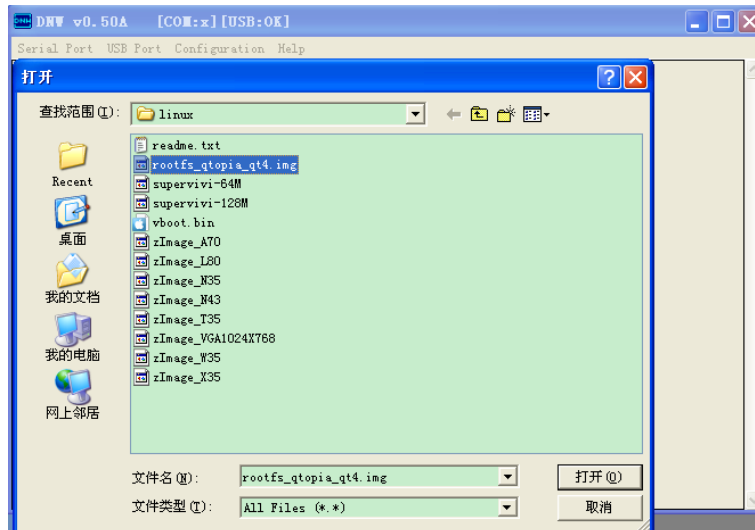
点击 DNW 软件的“USB Port->Transmit->Transmit”，选择 zImage_Q35，如下图：



6、安装根文件系统
选择功能键[y], 如下图:



点击 DNW 软件的“USB Port->Transmit->Transmit”，选择 rootfs_qtopia_qt4.img，如下图:



开始传送文件系统，时间稍长，请等候，传送过程如下图:



```
115200 - 超级终端
文件(F) 编辑(E) 查看(V) 呼叫(C) 传送(T) 帮助(H)
[?] [?] [?] [?] [?] [?]
[a] Absolute User Application
[n] Download Nboot for WinCE
[l] Download WinCE boot-logo
[w] Download WinCE NK.bin
[d] Download & Run
[z] Download zImage into RAM
[g] Boot linux from RAM
[f] Format the nand flash
[b] Boot the system
[s] Set the boot parameters
[u] Backup NAND Flash to HOST through USB(upload)
[r] Restore NAND Flash from HOST through USB
[q] Goto shell of vivi
[i] Version: 0945-2K
Enter your selection: v
USB host is connected. Waiting a download.

Now, Downloading [ADDRESS:30000000h,TOTAL:956757221
Downloaded file at 0x30000000, size = 95675712 bytes
Flash params: oobsize = 64, oobblock = 2048, erasesize = 131072, partition size =
262668288
Erasing and programming NAND with yaffs image
Block erasing(addr/count) --- Block bad(addr/count) --- Block processed/All(%)

0x0ffc0000/01995      0x0ebe0000/00009      02004/02004=100%
已连接 0:53:15 自动检测 115200 8-N-1 SCROLL CAPS NUM 插 打印
```

传送完毕显示 Load ysffs OK。

7、进入 Linux 系统

断电后将 JMY901 拨动开关 S2 拨到 NAND 端，然后重新上电，便从 NAND Flash 启动系统了，如下图：

```
115200 - 超级终端
文件(F) 编辑(E) 查看(V) 呼叫(C) 传送(T) 帮助(H)
[?] [?] [?] [?] [?] [?]
yaffs: dev is 32505859 name is "mtdblock3"
yaffs: passed flags ...
yaffs: Attempting MTD mount on 31.3, "mtdblock3"
yaffs: auto selecting yaffs2
block 783 is bad
block 943 is bad
block 1090 is bad
block 1110 is bad
block 1111 is bad
block 1136 is bad
block 1171 is bad
block 1189 is bad
block 1845 is bad
yaffs_read_super: isCheckpointed 0
VFS: Mounted root (yaffs filesystem) on device 31:3.
Freeing init memory: 156K
[29/Nov/1999:16:00:01 +0000] boa: server version Boa/0.94.13
[29/Nov/1999:16:00:01 +0000] boa: server built Jul 26 2010 at 15:58:29.
[29/Nov/1999:16:00:01 +0000] boa: starting server pid=679, port 80

Try to bring eth0 interface up.....eth0: link down
Done
Please press Enter to activate this console. _
已连接 0:57:05 自动检测 115200 8-N-1 SCROLL CAPS NUM 插 打印
```

按回车便可进入 Linux 文件系统进行操作了，如下图：

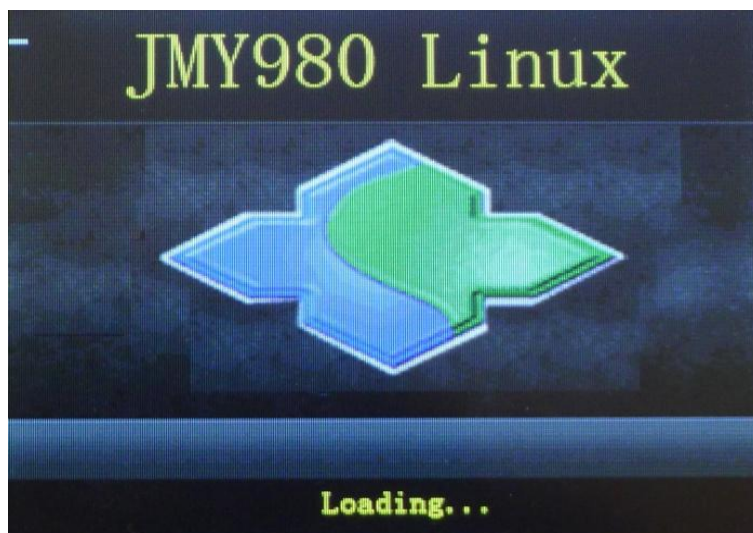
```
115200 - 超级终端
文件(F) 编辑(E) 查看(V) 呼叫(C) 传送(T) 帮助(H)
[?] [?] [?] [?] [?] [?]
block 943 is bad
block 1090 is bad
block 1110 is bad
block 1111 is bad
block 1136 is bad
block 1171 is bad
block 1189 is bad
block 1845 is bad
yaffs_read_super: isCheckpointed 0
VFS: Mounted root (yaffs filesystem) on device 31:3.
Freeing init memory: 156K
[29/Nov/1999:16:00:01 +0000] boa: server version Boa/0.94.13
[29/Nov/1999:16:00:01 +0000] boa: server built Jul 26 2010 at 15:58:29.
[29/Nov/1999:16:00:01 +0000] boa: starting server pid=679, port 80

Try to bring eth0 interface up.....eth0: link down
Done
Please press Enter to activate this console.
[root@FriendlyARM /]# ls
bin          home        lost+found  proc        sys         var
dev          lib         mnt        root        tmp         www
etc          linuxrc    opt        sbin       usr
[root@FriendlyARM /]#
已连接 1:07:28 AMSTW 115200 8-N-1 SCROLL CAPS NUM 插 打印
```

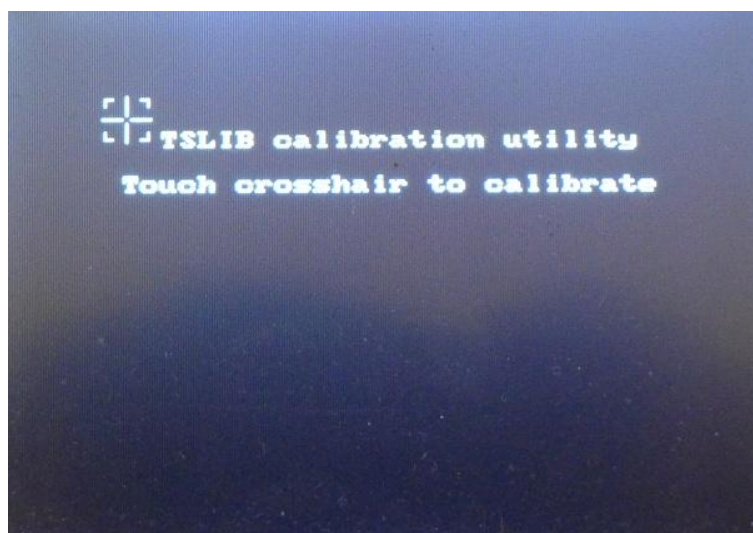


8、带触摸显示屏的操作

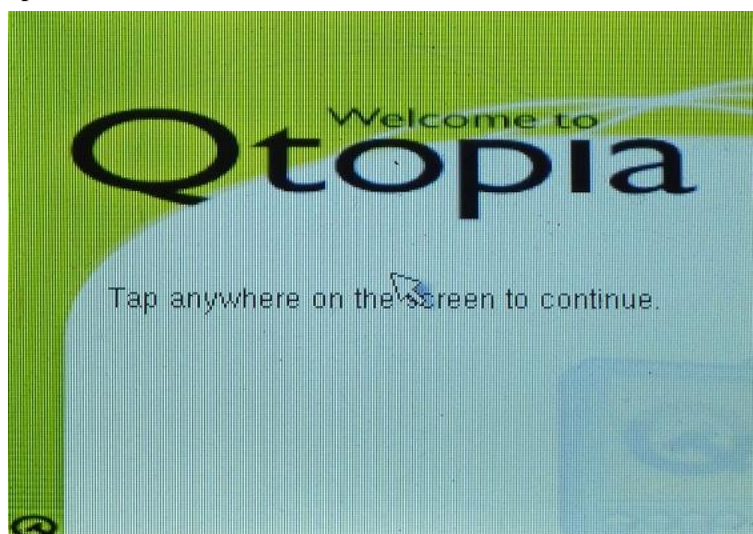
如果连接了触摸屏，开机后便可看见如下 Linux 启动画面：



触控校准，点击十字光标校准屏幕，如下图：



校准后进入 Qtopia 界面，如下图：



点击屏幕进入系统界面，如下图：



到此 Linux 系统安装完毕！

3.2.3 下载 WindowsCE 系统

下载 WinCE 系统的方法与 Linux 系统一样，这里就不一一演示了，只是功能选择和烧写的文件不同，烧写文件存放在“JMY980TOOLS\images\wince6.0”目录下。

烧写步骤：

- 1、选择功能键[n]，烧写文件 nboot_Q35.bin；
- 2、选择功能键[l]，烧写文件 bootlogo.bmp；
- 3、选择功能键[w]，烧写文件 NK_Q35.bin；
- 4、安装 WinCE 与 WindowsXP 的同步软件 ActiveSync，存放目录“JMY980TOOLS\windows 平台工具\ActiveSync”；

烧写完系统后切换到 NAND Flash 启动，触摸屏会出现如下界面：



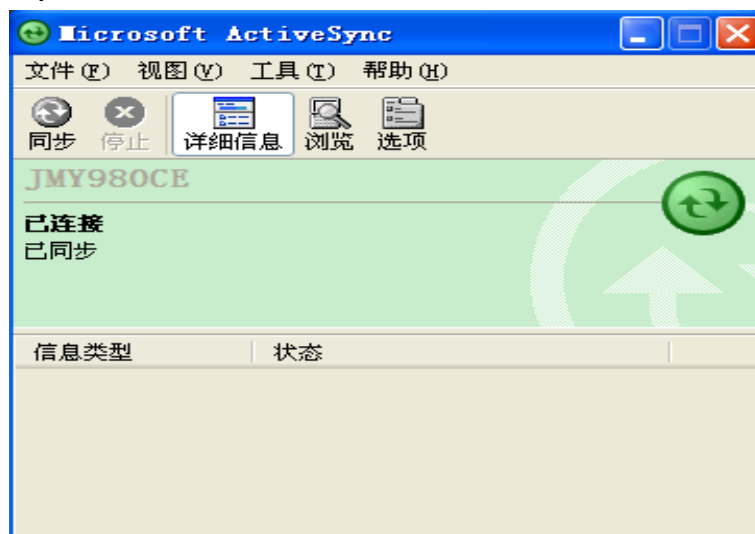
启动 WinCE 界面，如下图：



进入 WinCE 操作系统，如下图：



同步软件 ActiveSync 会弹出，如下图：



到此 WinCE 系统安装完毕！



4 WindowsCE 6.0 开发指南

4.1 建立 WindowsCE 6.0 开发环境

注：以下软件和步骤均基于 Microsoft Windows XP SP3 系统，其它 Windows 系统未经测试。

Windows CE 6.0 的安装过程十分繁琐，并且对开发主机的要求比较高（否则会很慢），我们建议用户特别是初学者务必按照我们介绍的步骤安装开发环境。

这里是我们采用的开发主机的配置，仅供参考：

CPU：Pentium(R) Dual-Core E6700 @3.20GHZ

内存：DDR2 4GB

硬盘空间：500GB

安装所需的软件列表如下（部分提供）：

Visual Studio 2005(不提供)

下载地址：

http://download.microsoft.com/download/e/1/4/e1405d9e-47e3-404c-8b09-489437b27fb0/En_vs_2005_Pro_90_Trial.img

Visual Studio 2005 Service Pack 1(文件名：VS80sp1-KB926601-X86-ENU.exe)

下载地址：

<http://www.microsoft.com/en-us/download/details.aspx?id=5553>

Visual Studio 2005 Service Pack 1 Update for Windows Visat

(文件名：VS80sp1-KB932232-X86-ENU.exe)

下载地址：

<http://www.microsoft.com/en-us/download/details.aspx?id=7524>

Visual Studio 2005 Service Pack 1 ATL Security Update

(文件名：VS80sp1-KB971090-X86-INTL.exe)

下载地址：

<http://www.microsoft.com/en-us/download/details.aspx?id=25287>

Windows Embedded CE 6.0

下载地址：

<http://www.microsoft.com/en-us/download/details.aspx?id=20083>

Windows Embedded CE 6.0 Platform Builder Service Pack 1

下载地址：

<http://www.microsoft.com/en-us/download/details.aspx?id=4097>

Windows Embedded CE 6.0 R2

下载地址：

<http://www.microsoft.com/en-us/download/details.aspx?id=18111>

Windows Embedded CE 6.0 R3

下载地址：

<http://www.microsoft.com/downloads/details.aspx?familyid=bc247d88-ddb6-4d4a-a595-8eee3556fe46&displaylang=ja&displaylang=en>

以上列表顺序也说明了这些软件的安装顺序：先安装 Visual Studio 2005 及补丁，再安装

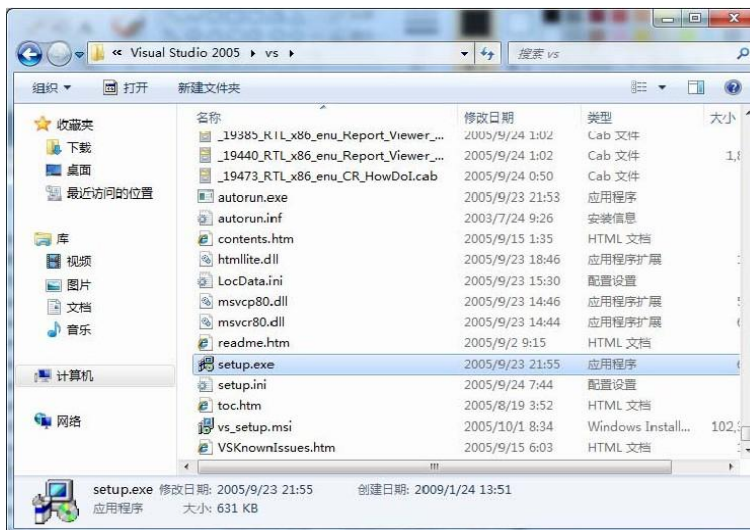


Windows CE 6.0 及补丁。

注：Windows CE 6.0 所使用的 Platform Builder 和以往的 Windows CE 5.0/4.2 等均不同，它并不是独立的开发平台软件，而是作为 VS2005 的一个插件来安装使用的，因此必须先安装 VS2005，以后所有的内核配置编译等开发都基于 VS2005 进行。

4.1.1 安装 Visual Studio 2005 及补丁

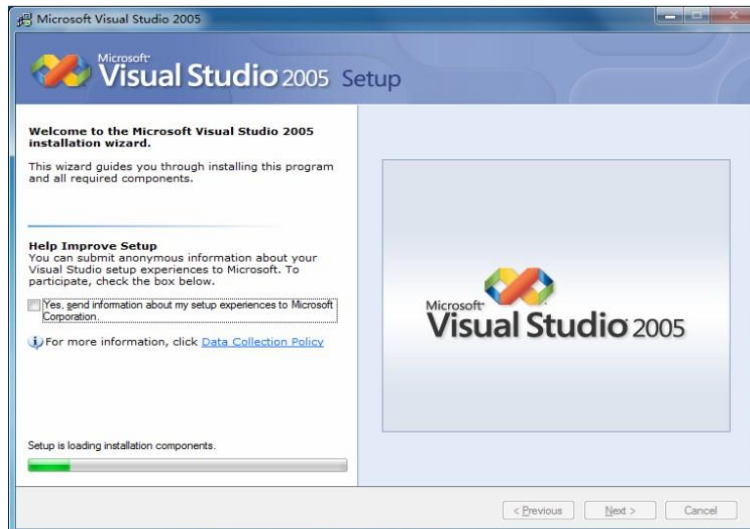
Step1: 打开 Visual Studio 2005 文件夹，找到 setup.exe，双击运行开始安装。



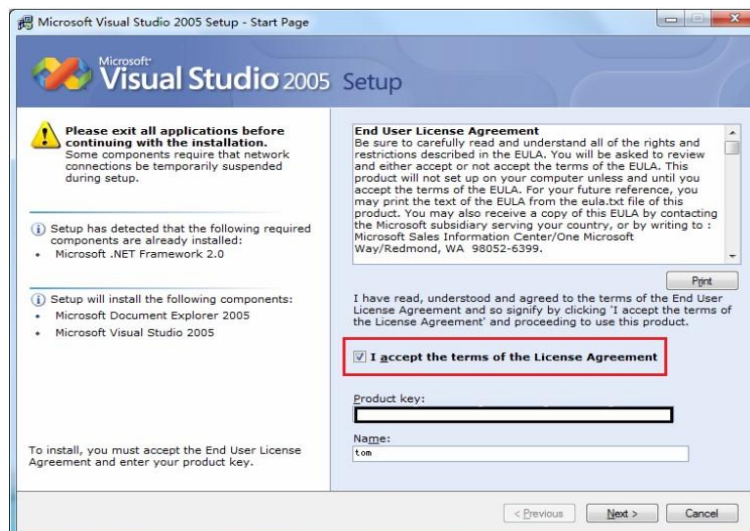
Step2: 出现如图界面，点“Install Visual Studio 2005”，继续



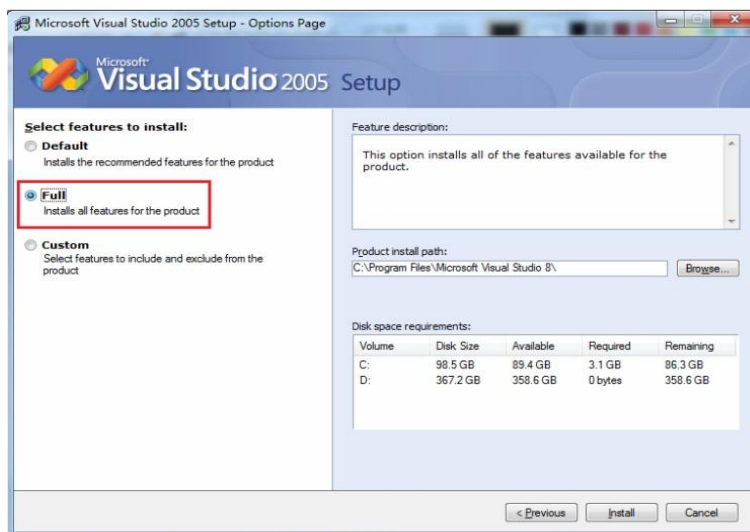
Step3: 出现如图界面，稍等片刻，点“Next”继续



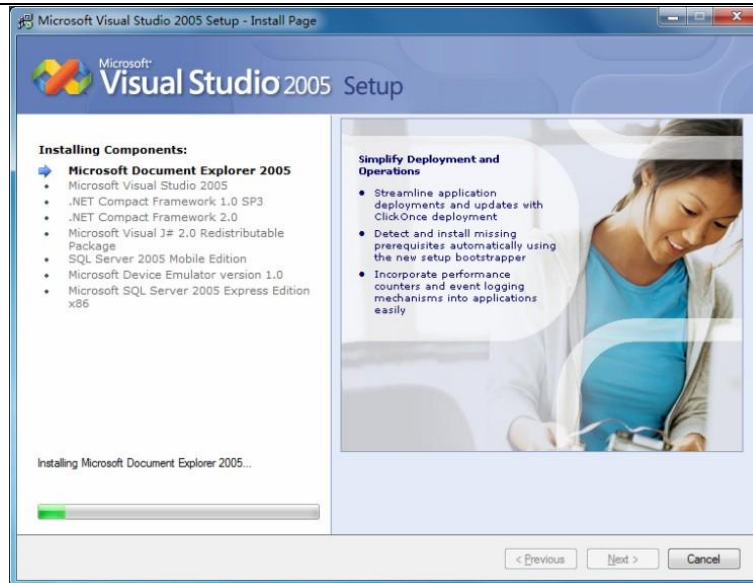
Step4: 出现如图界面，注意点选红色框的，并输入序列号，点“Next”继续



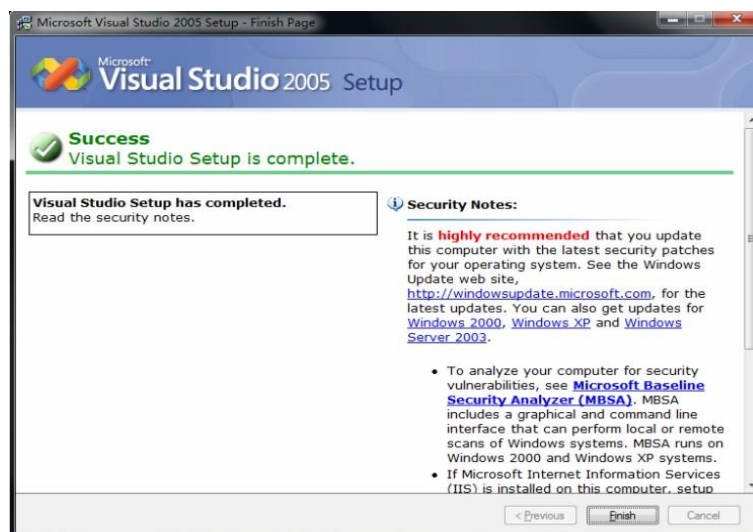
Step5: 出现如图界面，选择安装类型，在此选择完全安装，即“Full”，点“Next”继续



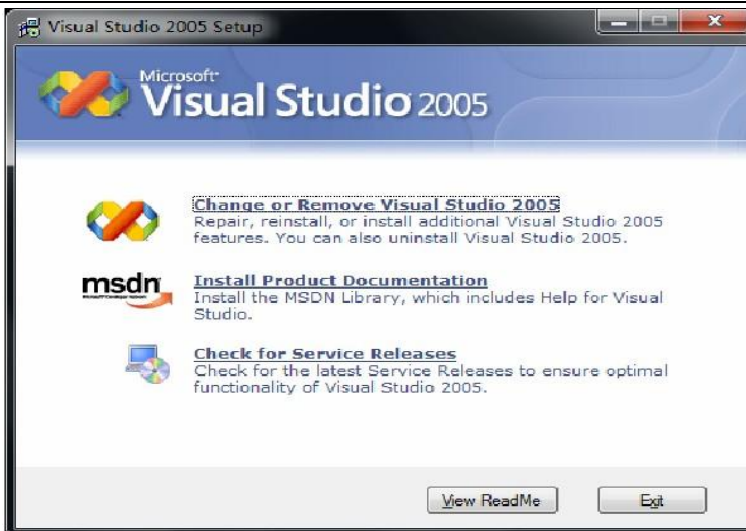
Step6: 出现如图界面，开始正式安装 Visual Studio 2005，此过程较长，请耐心等待。



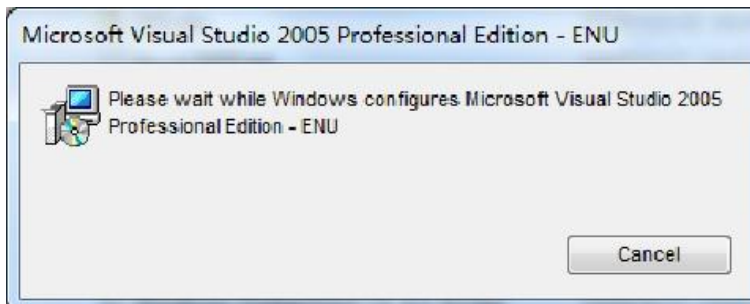
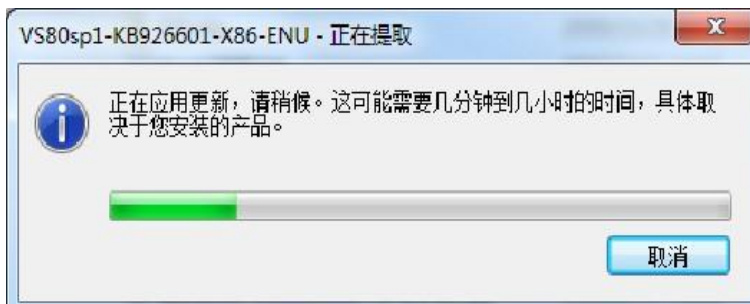
Step7: Visual Studio 2005 安装完毕，出现如下画面，点“Finish”结束安装。



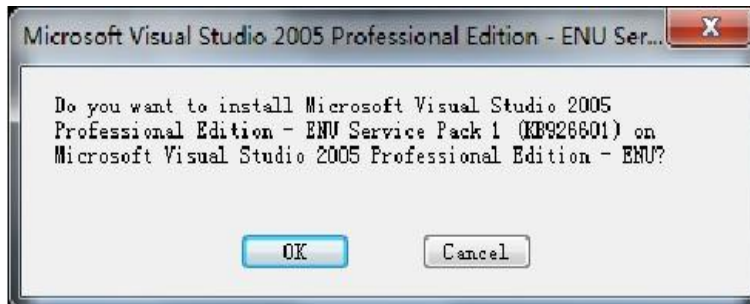
接着会出现如图界面，点“Exit”退出即可。



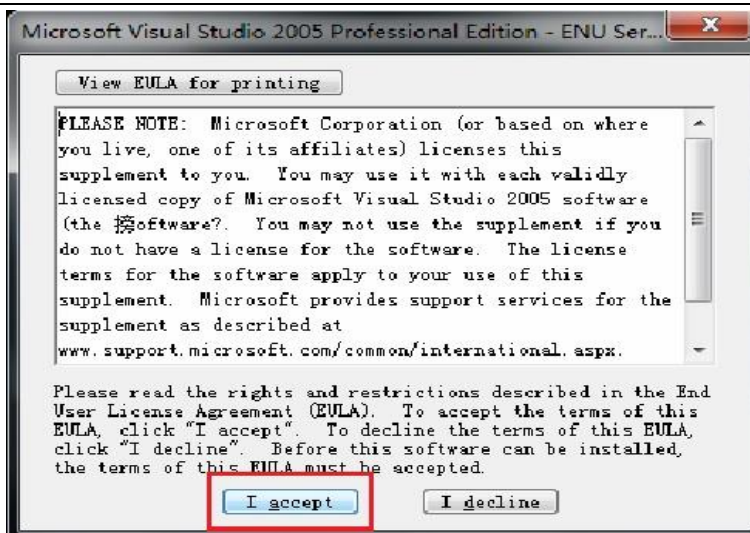
Step8: 现在开始安装第一个补丁文件 Visual Studio 2005 Service Pack 1，双击运行 VS80sp1-KB926601-X86-ENU.exe 开始安装，出现如图界面



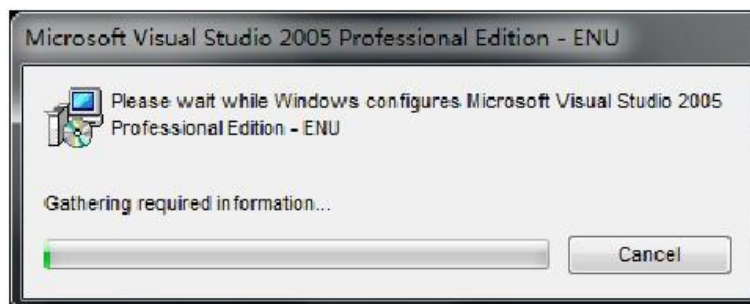
Step9: 须稍等片刻，出现如图画面，点“OK”开始正式安装



Step10: 接受安装许可协议，点“I accept”继续



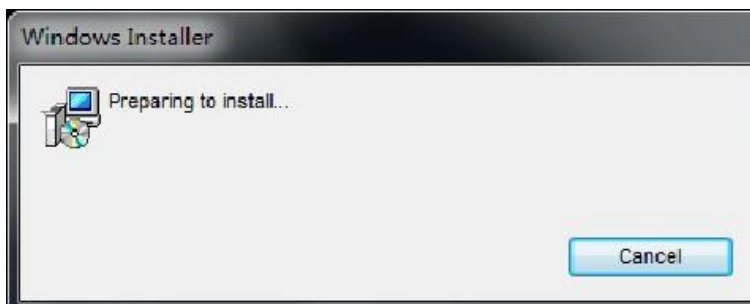
Step11: 出现安装过程界面，此过程较长，请耐心等待

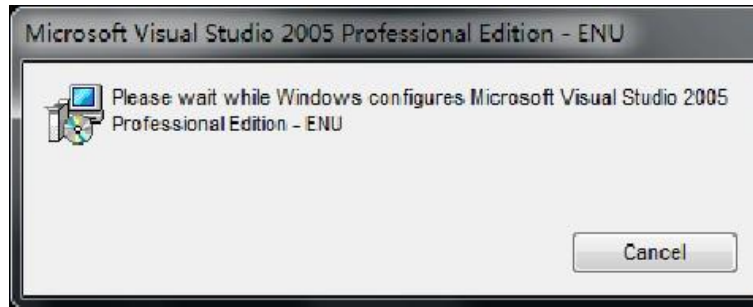


Step12: 安装完毕，出现如下界面，点“OK”结束本补丁的安装



Step13: 接下来安装第二个补丁 Visual Studio 2005 Service Pack 1 Update for Windows Vista，双击运行 VS80sp1-Kb932232-X86-ENU.exe，依次出现如图界面

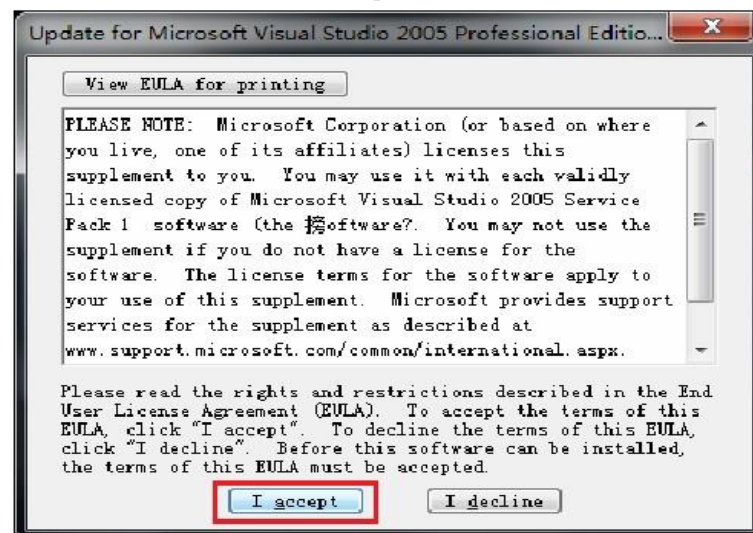




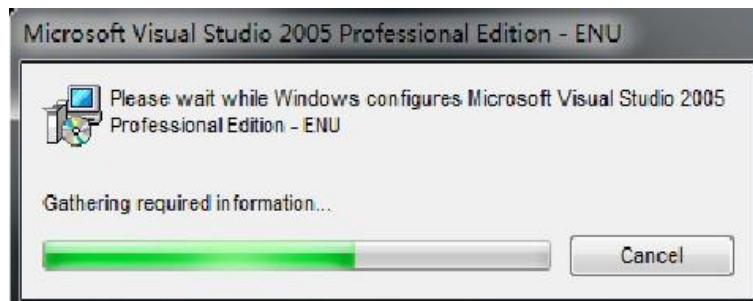
Step14: 稍等片刻，出现如图界面，点“OK”继续



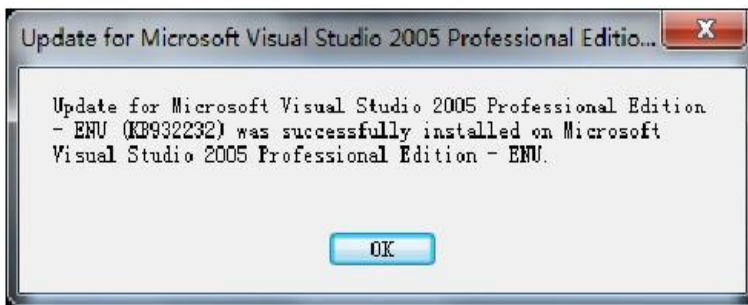
Step15: 出现安装许可协议界面，点“I accept”继续



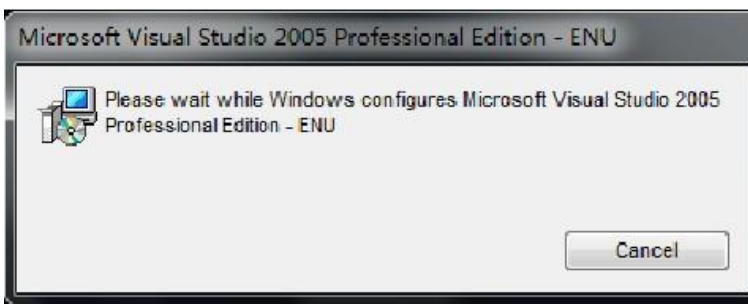
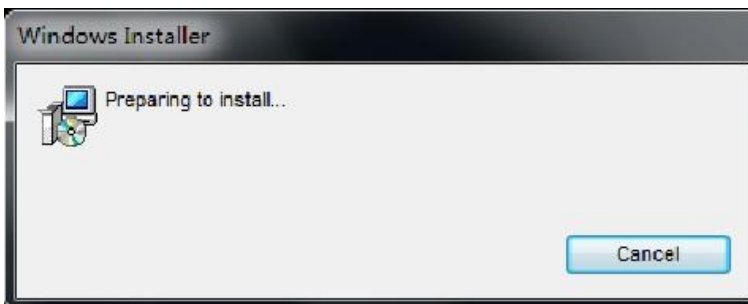
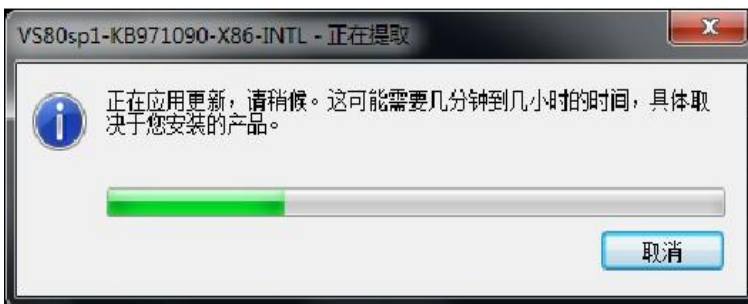
Step16: 出现安装过程界面，此过程较长，请耐心等待



Step17: 安装完毕，出现如下界面，点“OK”结束本补丁的安装



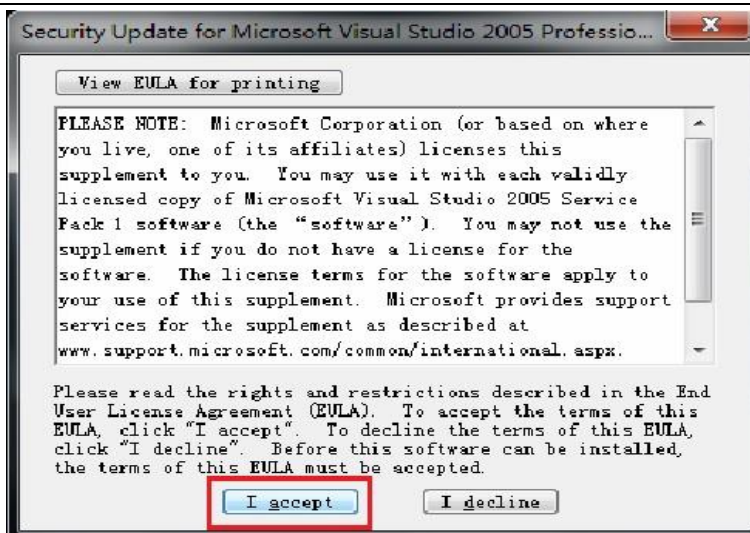
Step18: 接下来安装第三个补丁 Visual Studio 2005 Service Pack 1 ATL Security Update, 双击运行 VS80sp1-KB971090-X86-INTL.exe, 依次出现如图界面



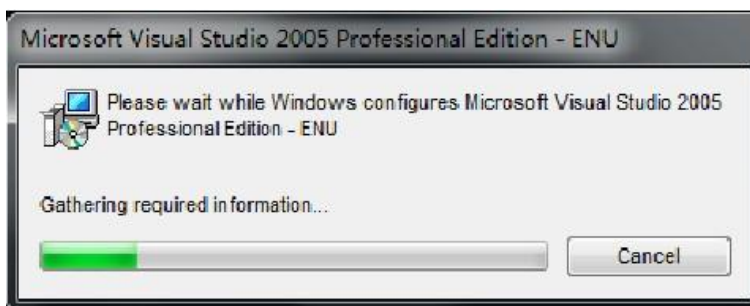
Step19: 稍等片刻, 出现如图界面, 点“OK”继续



Step20: 出现安装许可协议界面, 点“I accept”继续



Step21: 出现安装过程界面，此过程较长，请耐心等待



Step22: 安装完毕，出现如下界面，掉“OK”结束本补丁的安装



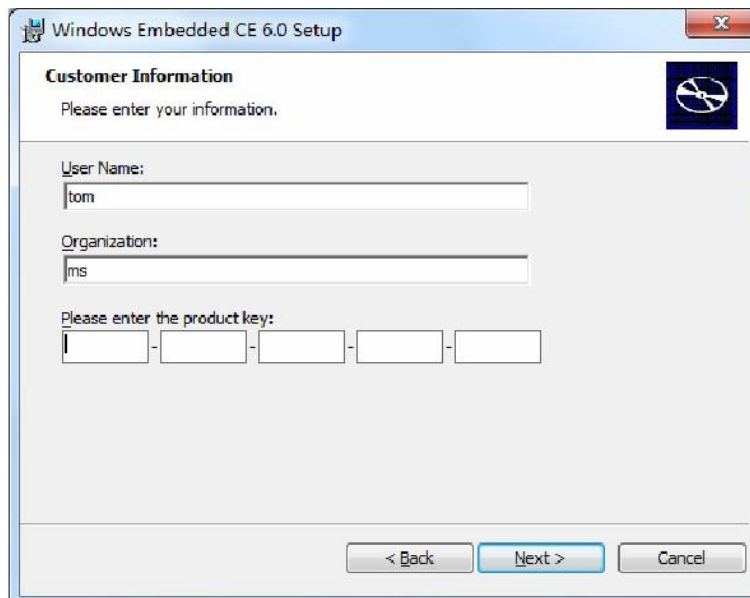
至此，基于 Windows XP 平台的 Visual Studio 2005 及其补丁已经安装完毕。

4.1.2 安装 Windows CE 6.0 及补丁

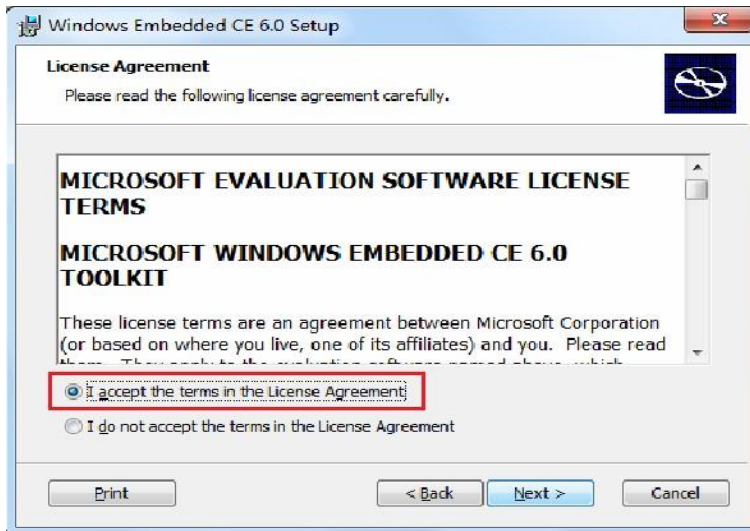
Step1: 点击“Windows Embedded CE 6.0.msi”开始安装，如图，点“Next”继续



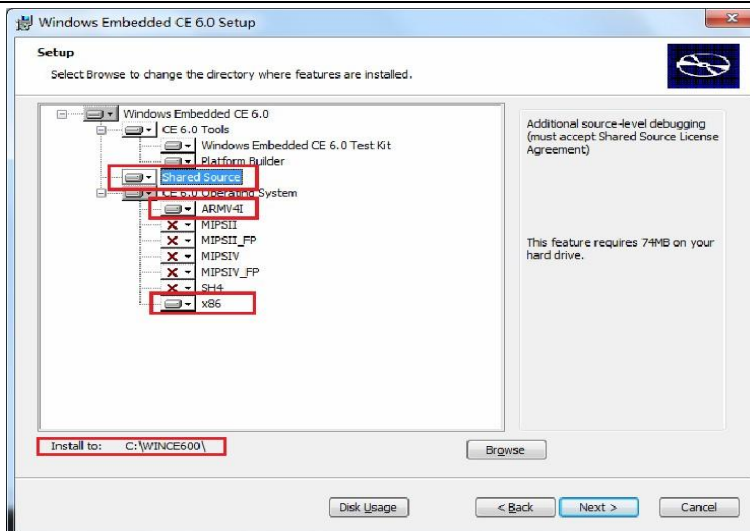
Step2: 输入产品密钥，点“Next”继续



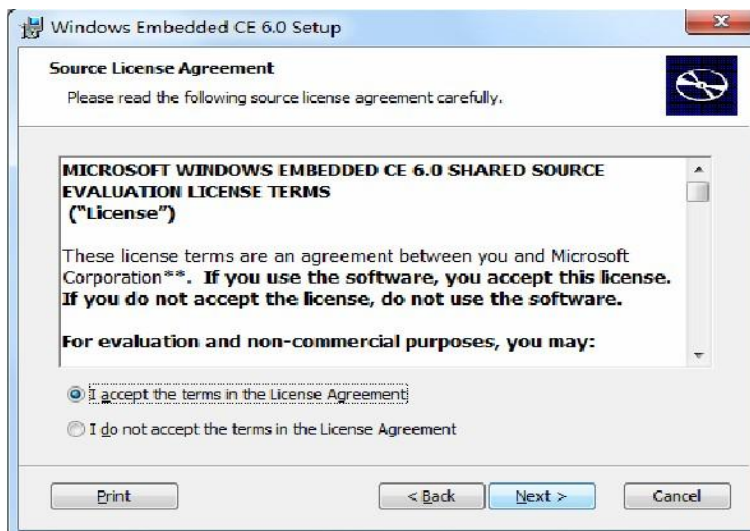
Step3: 出现安装许可协议界面，选择“I accept”，点“Next”继续



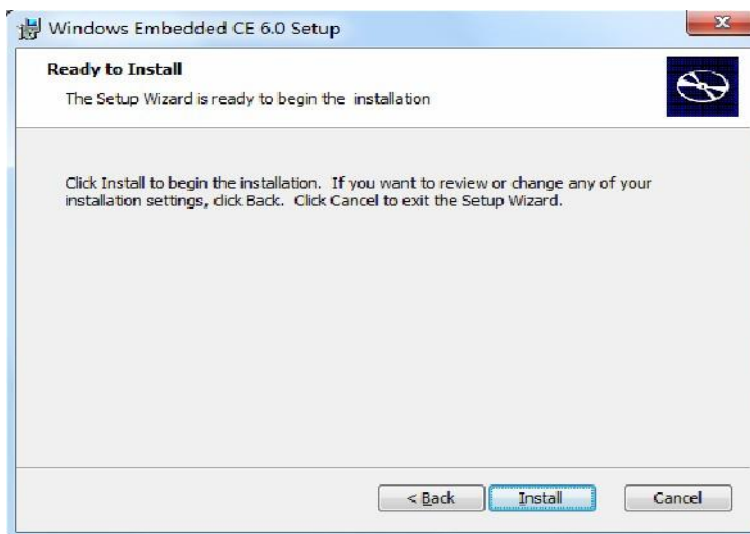
Step4: 选择及设置如图，点“Next”继续



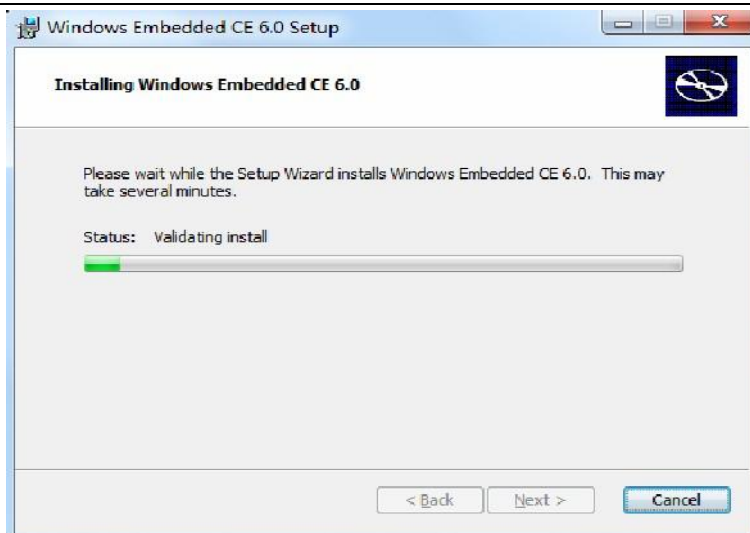
Step5: 出现如图界面，选择如图，点“Next”继续



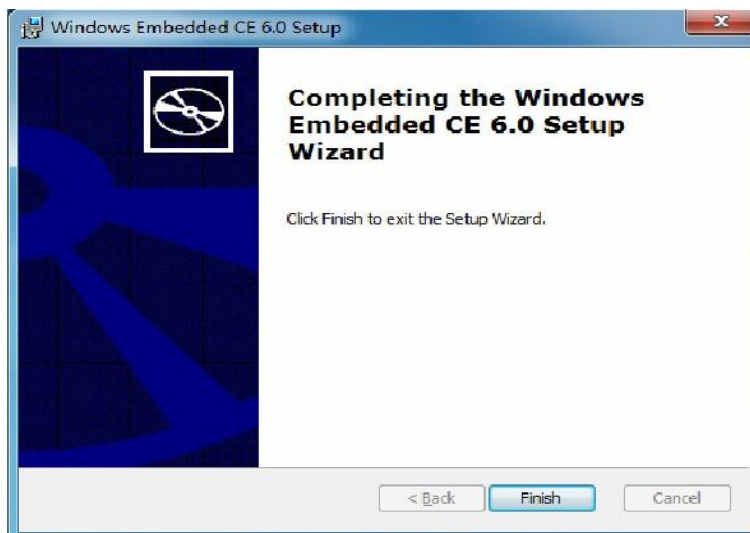
Step6: 出现如图界面，点“Install”继续



Step7: 开始正式安装，如图，此过程时间较长，请耐心等待



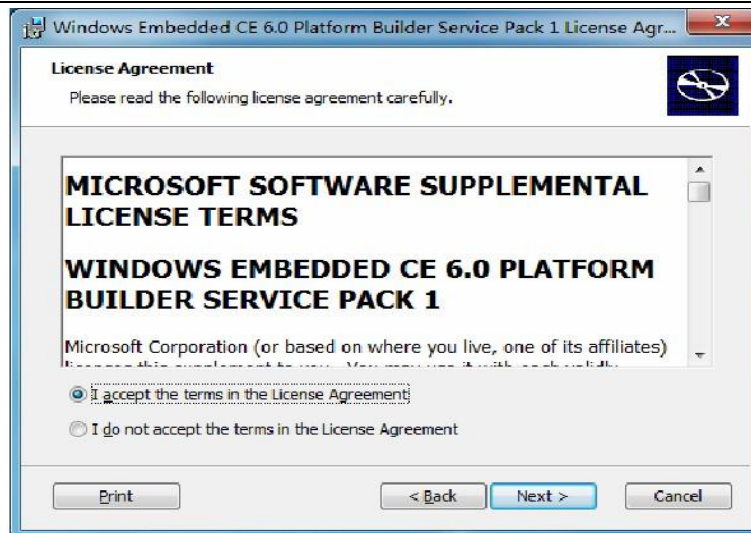
Step8: 安装结束，出现如图界面，点“Finish”结束安装。



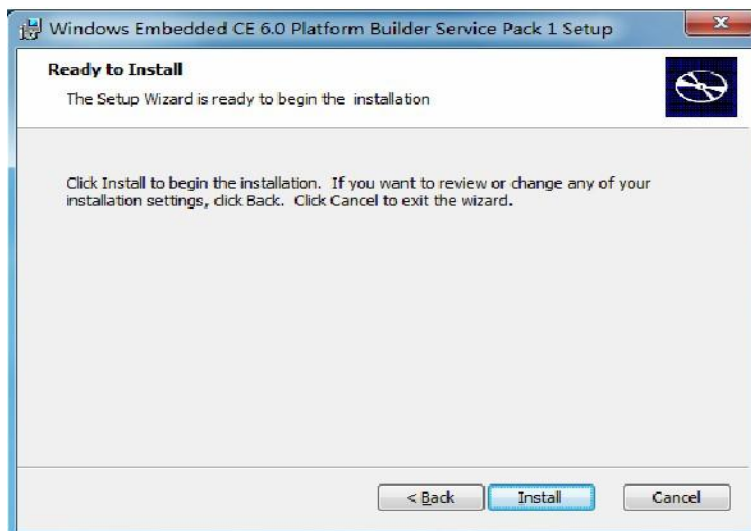
Step9: 接下来安装 Windows CE 6.0 的第一个补丁“Windows Embedded CE 6.0 Platform Builder Service Pack 1.msi”，点击安装文件，出现如图界面，点“Next”继续



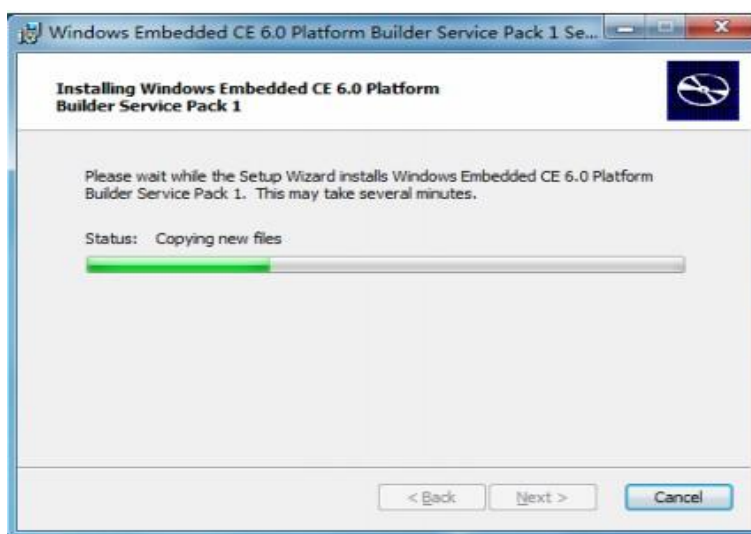
Step10: 出现如图界面，选“I accept”，并点“Next”继续



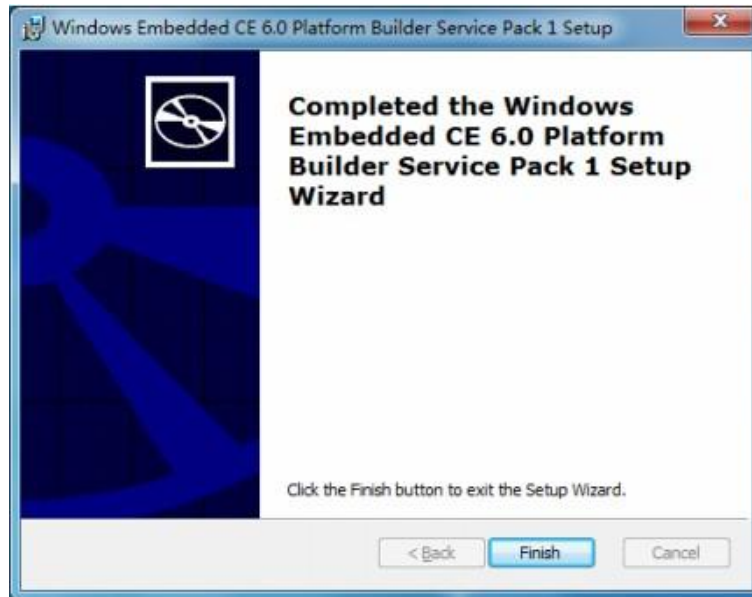
Step11: 出现如图界面，点“Next”继续



Step12: 开始正式安装，如图，此过程时间较长，请耐心等待



Step13: 安装结束，出现如图界面，点“Finish”结束安装。



Step14: 接下来安装 Windows CE 6.0 的第二个补丁“Windows Embedded CE 6.0 R2.msi”，如图，点“Next”继续

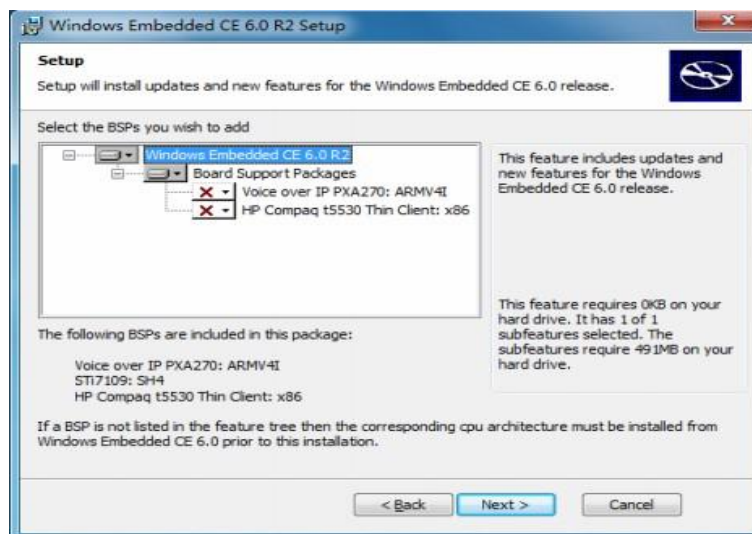


Step15: 出现如图界面，选“I accept”，点“Next”继续

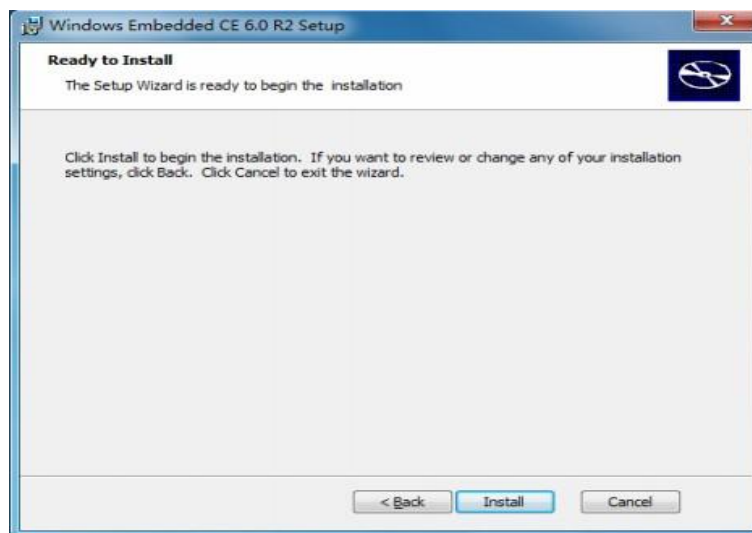




Step16: 出现如图界面，不用做任何改动，点“Next”继续



Step17: 出现如图界面，点“Next”继续



Step18: 开始正式安装，此过程时间较长，请耐心等待



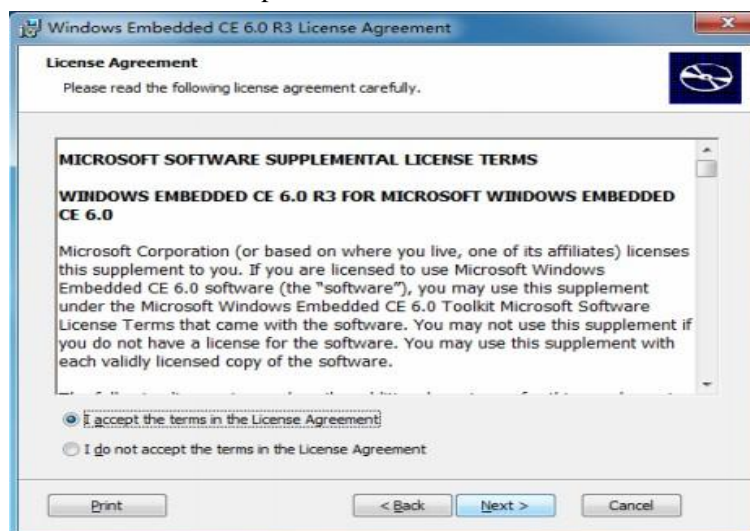
Step19: 安装结束，出现如图界面，点“Finish”结束安装



Step20: 现在开始安装 Windows CE 6.0 的第三个补丁 R3, 开始安装“Windows Embedded CE 6.0 R3.msi”, 如图



Step21: 出现如图界面, 选 “I accept”, 并点 “Next” 继续



Step22: 出现如图界面, 点 “Next” 继续



Step23: 开始正式安装，此过程时间较长，请耐心等待



Step24: 安装结束，出现如图界面，点“Finish”结束安装



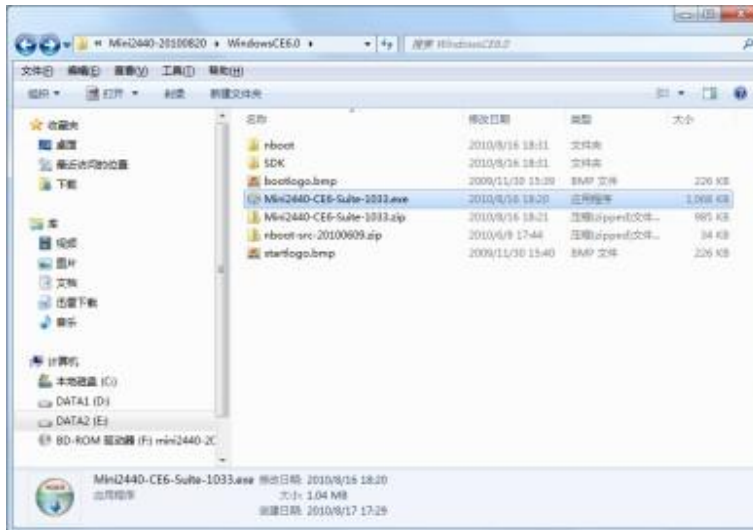


4.1.3 安装 BSP 及内核工程示例

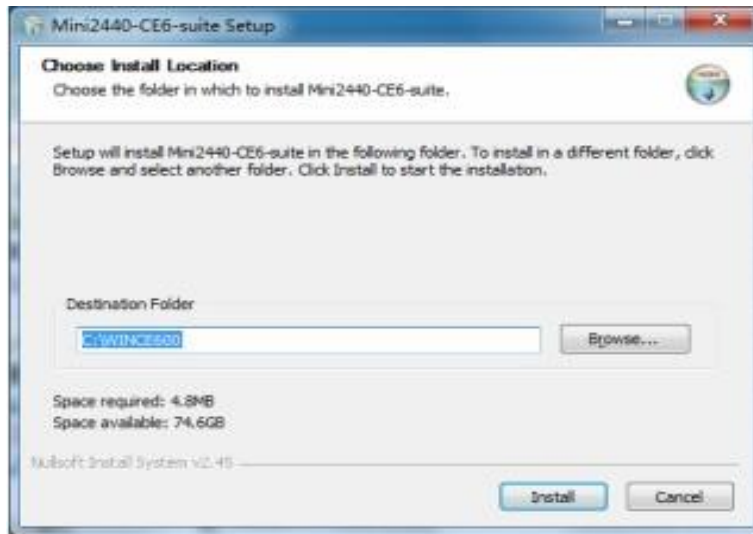
JMY980 (mini2440) 的 BSP 和 示例 工程 等 文件 只 有 一 个 安 装 文 件 mini2440-ce6-suite-1033.exe，其中包含所有的 BSP 源代码及两个内核工程示例。

注：请以以下步骤安装 BSP，建议不要改变安装路径，否则有可能无法编译通过。

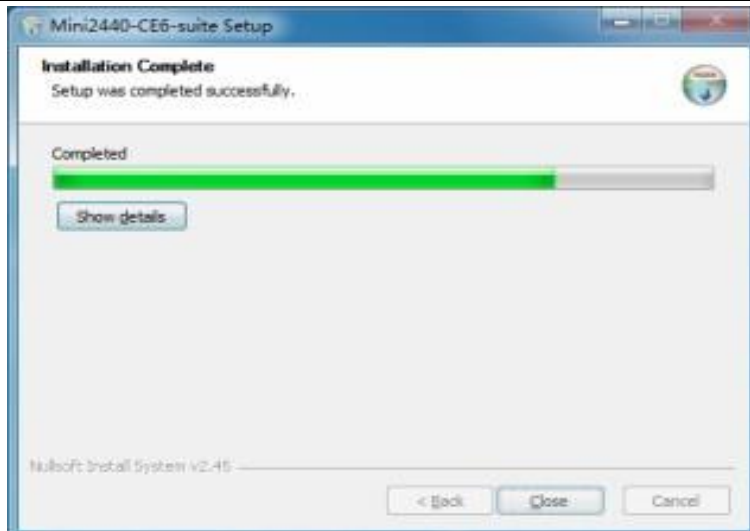
Step1: 找到 mini2440-ce6-suite-1033.exe 可执行安装文件，并双击运行



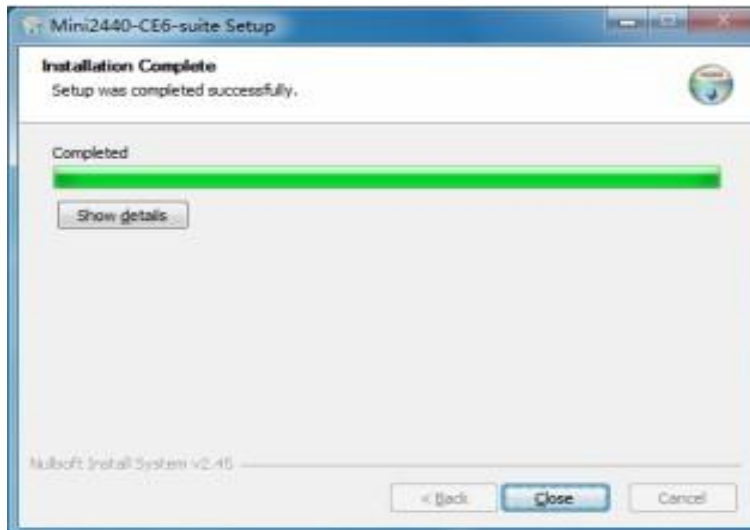
Step2: 保持各项设置不变，点“Install”继续



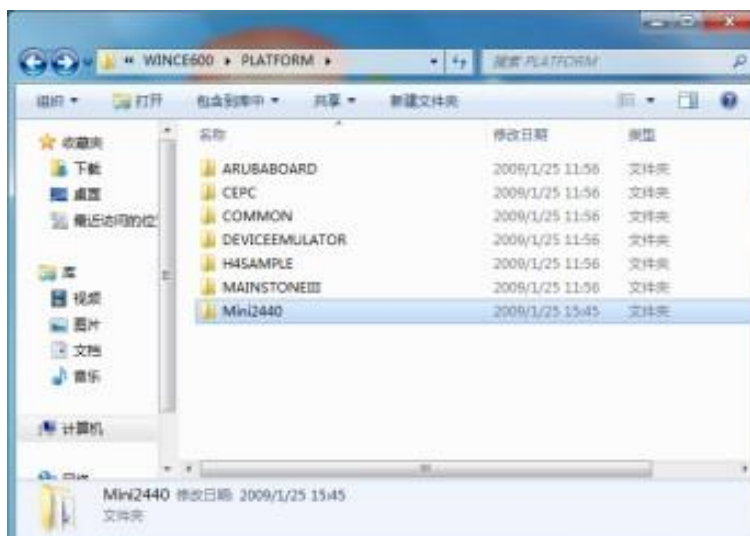
Step3: 出现安装过程界面，因为安装的文件很小，安装会很快结束



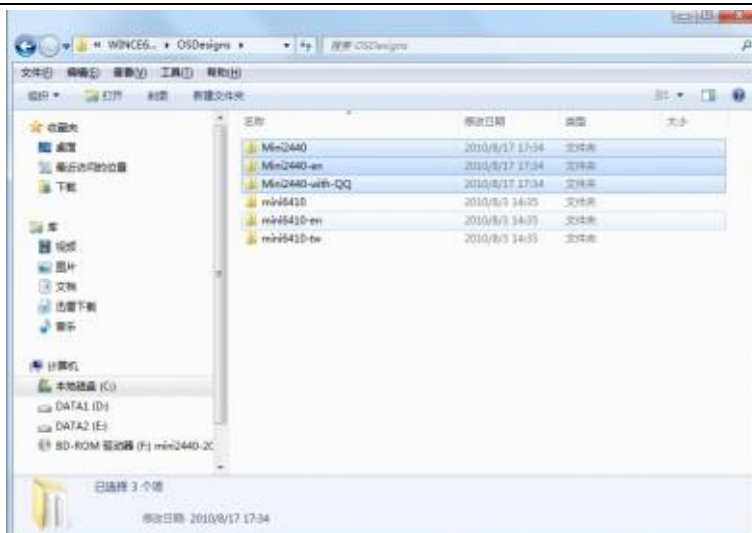
Step4: 安装结束，出现如图界面，点“Close”结束安装



安装完毕，会在 WinCE600\PLATFORM 目录下创建 mini2440 BSP 目录，如图



并在 WinCE600\OSDesigns 目录下分别创建三个内核示例工程文件目录，如图



其中:

Mini2440 目录中包含的工程文件, 可以用来编译生成光盘中对应的中文版 WinCE 内核映像
Mini440-with-QQ 目录中的工程文件, 可以用来编译生成包含腾讯 QQ 的 WinCE 内核映像
Mini440-en 目录中包含的工程文件, 可以用来生成英文版的 WinCE 内核映像
至此, Windows CE 6.0 的开发环境就已经完全创建了。

4.1.4 各个驱动程序源代码的位置

Mini440 目前拥有最齐全的 BSP, 也就是驱动程序, 并且每个驱动基本都有相应的图形界面测试程序, 各个驱动程序的源代码位置说明如下:

- (1) LED 驱动
 \Mini2440\SRC\DRIVERS\LEDdriver
- (2) 按键驱动
 \Mini2440\SRC\DRIVERS\Userkey
- (3) PWM 控制蜂鸣器驱动
 \Mini2440\SRC\DRIVERS\PWM
- (4) ADC 转换驱动
 \Mini2440\SRC\DRIVERS\Touch
 说明: ADC 驱动实际和触摸屏驱动在同一个文件中实现
- (5) I2C 驱动
 \Mini2440\SRC\DRIVERS\IIC
- (6) RTC 驱动
 \Mini2440\SRC\DRIVERS\Rtc
- (7) 串口驱动
 \Mini2440\SRC\DRIVERS\Serial
- (8) 触摸屏驱动
 \Mini2440\SRC\DRIVERS\Touch
- (9) USB 驱动
 \Mini2440\SRC\DRIVERS\Usb
- (10) SD 卡驱动
 \Mini2440\SRC\DRIVERS\SDHC



说明：支持高速大容量 SD 卡，最高可达 32GB

- (11) DM9000 网卡驱动
 \Mini2440\SRC\DRIVERS\dm9000
- (12) 音频驱动
 \Mini2440\SRC\DRIVERS\Wavedev
- (13) LCD 驱动
 \Mini2440\SRC\DRIVERS\Display
- (14) 背光驱动
 \Mini2440\SRC\DRIVERS\Backlight
- (15) CMOS 摄像头驱动
 \Mini2440\SRC\DRIVERS\Camera

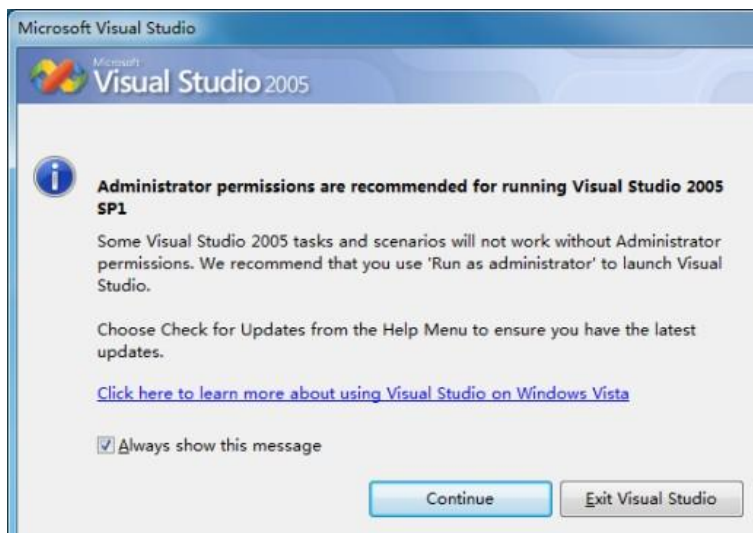
4.2 配置和编译 Windows CE 6.0 内核及 Bootloader

因为 Windows CE6 的内核配置比较复杂，很容易因配置不对而导致无法编译通过，众所周知 Windows CE 平台的编译是十分耗时的，因此用户按照下面的步骤直接打开编译就可以了，光盘中 images\wince6.0 目录中有相应的编译好的内核映像文件。

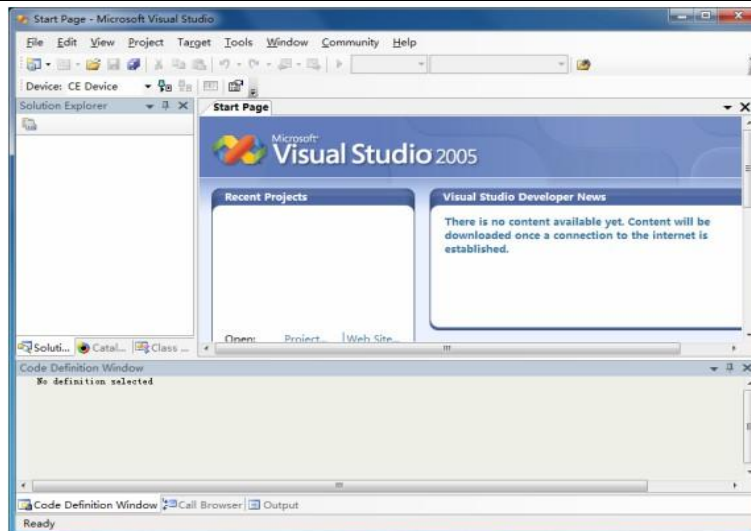
4.2.1 编译缺省内核工程示例

现在，我们启动 VS2005 来编译刚刚安装的 mini2440 BSP，第一次启动 VS2005 时有些事项要注意一下，如下步骤：

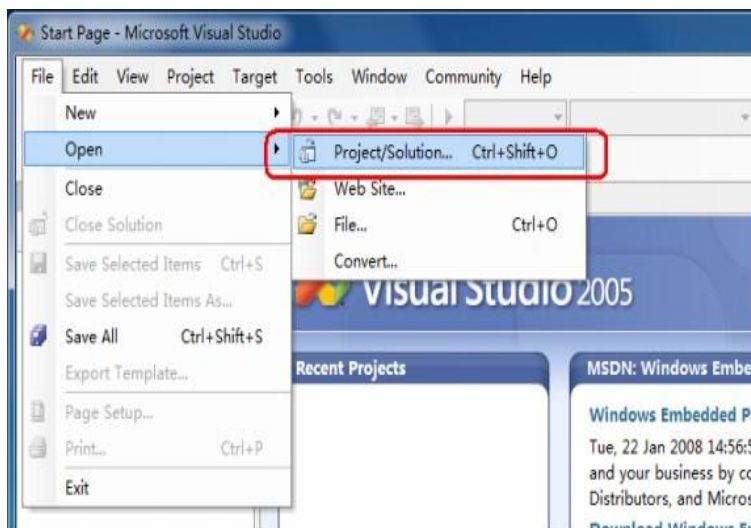
Step1: 点“开始”->“程序”->“Microsoft Visual Studio 2005”->“Microsoft Visual Studio 2005”，出现如图界面，点“Continue”继续



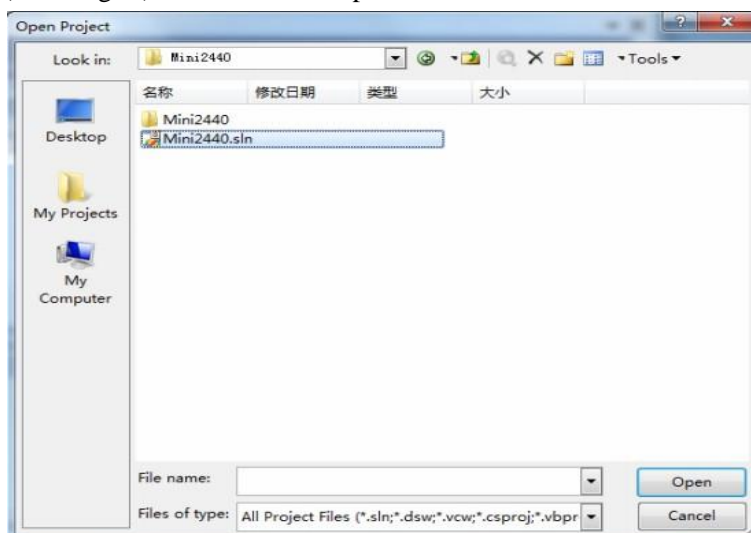
Step2: 出现如图界面，这是 VS2005 的工作界面，在此就不再对该界面赘述了，请用户参考常用的 VS2005 资料即可



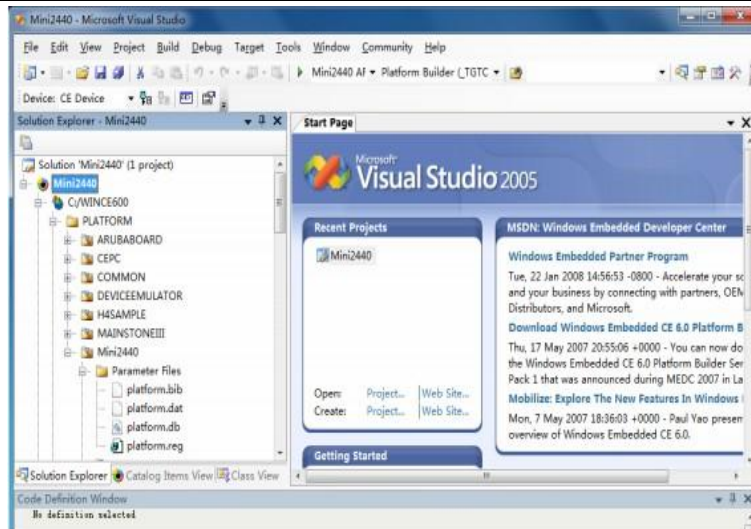
Step3: 点 File->Open->Project/Solution..., 如图



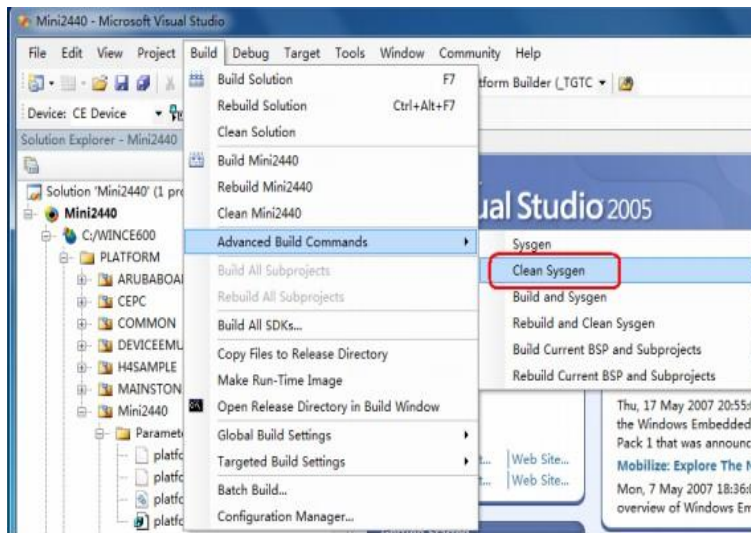
Step4: 出现文件选择窗口, 找到 mini2440 的缺省内核项目文件 (路径为: C:\WINCE600\OSDesigns\Mini2440), 点 Open 打开它, 如图



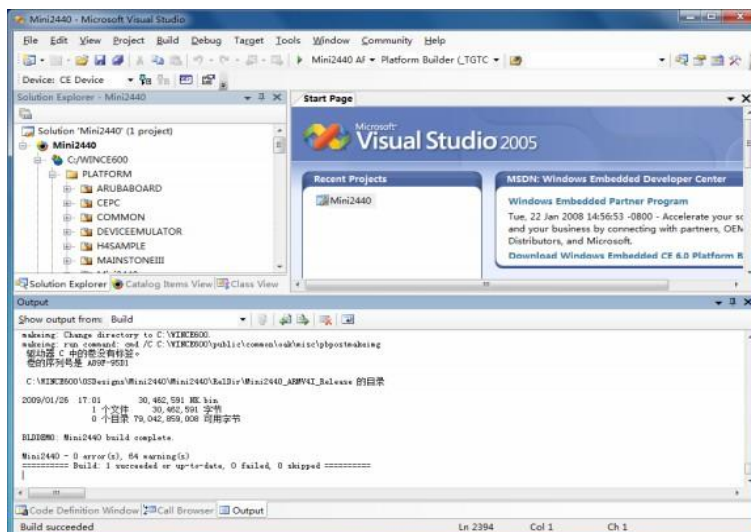
Step5: 稍等片刻, mini2440 的缺省内核项目被载入工作区, 出现如图界面

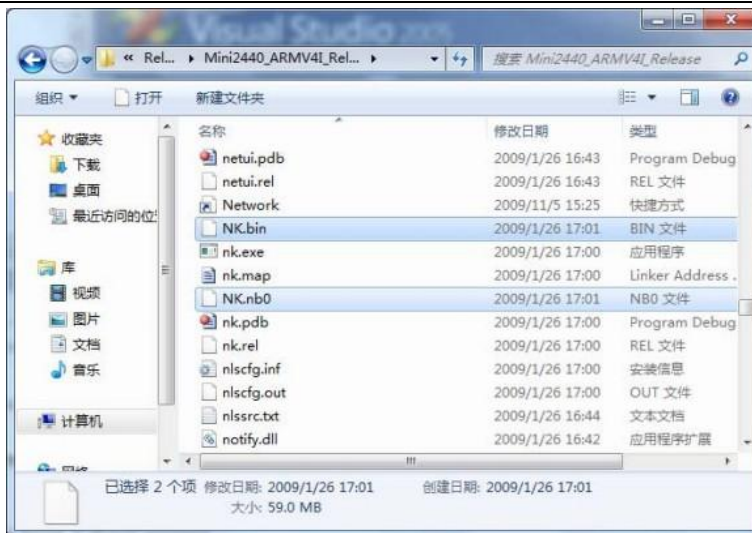


Step6: 点“Build->Advanced Build Commands->Clean Sysgen”开始编译内核，如图，此过程较长，请耐心等待



Step7: 编译完毕，结果如图所示，此时会生成内核映像文件 NK.bin 和 NK.nb0，路径如下：
C:\WINCE600\OSDesigns\Mini2440\Mini2440\RelDir\Mini2440_ARMV4I_Release





4.2.2 编译和烧写 Bootloader 之 NBOOT

说明：编译 Nboot 需要使用 ADS 集成开发环境，详见第五章。

Nboot 是一个十分简单的 bootloader，其大小不到 4K，一般烧写到 Nand Flash 的 Block 0 位置用来启动 WinCE 内核，Nboot 原由三星提供，我们对此做了很多改进，目前有如下特色功能：

- 自适应支持 64M/128M/256M/1G JMY980
- 支持开机画面快速显示
- 支持加载 WinCE 内核的动态进度条
- 启动 WinCE 仅需 5-10 秒，视内核大小而定

需要注意的是，Nboot 并不具备烧写功能，它只能读取已经烧写处理好的文件：开机画面（BootLogo）和 WinCE 内核。

Nboot 具有很方便的定制性，你可以通过头文件定义修改开机画面的显示位置、背景，以及进度条的颜色、位置、长宽等，这些定义位于 option.h 文件中，如下：

```
//通过更改定义，选择相应的 LCD 型号，此处默认选择 Q35,表示奇美横屏 LCD
```

```
//#define LCD_N35  
//#define LCD_L80  
//#define LCD_Q35  
//#define LCD_X35  
//#define LCD_W35  
//#define LCD_A70  
//#define LCD_VGA1024768
```

```
//设置背景色
```

```
#define BACKGROUND_R 0x00  
#define BACKGROUND_G 0x00  
#define BACKGROUND_B 0x00
```

```
//设置进度条的颜色
```

```
#define PROGRESS_BAR_R 0xFF  
#define PROGRESS_BAR_G 0xFF
```



```
#define PROGRESS_BAR_B 0x00
```

```
//设置开机图片的位置
```

```
#define LOGO_POS_TOP 0
```

```
#define LOGO_POS_LEFT 0
```

```
//设置启动条的位置和长宽
```

```
#define PROGRESS_BAR_TOP 260
```

```
#define PROGRESS_BAR_LEFT 20
```

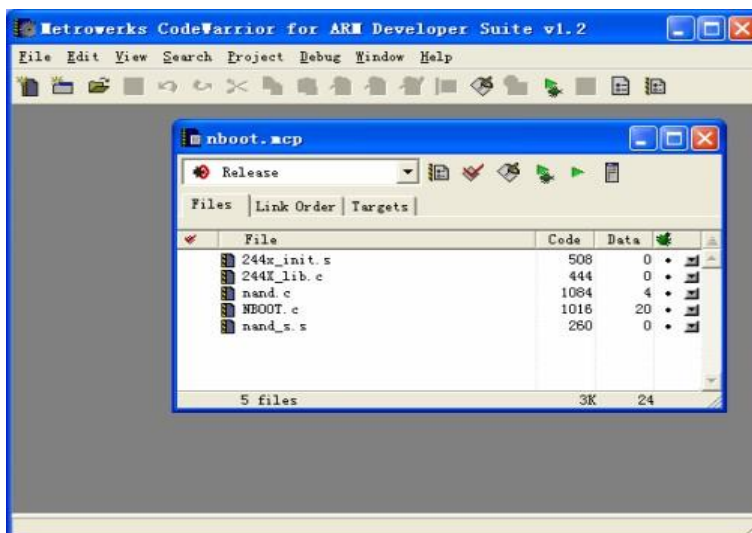
```
#define PROGRESS_BAR_WIDTH 200
```

```
#define PROGRESS_BAR_HEIGHT 12
```

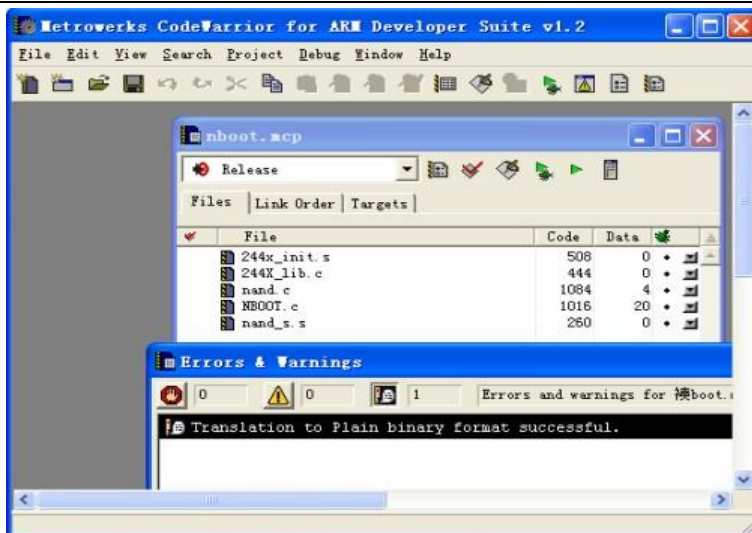
下面介绍 Nboot 的编译方法和步骤:

编译 Nboot

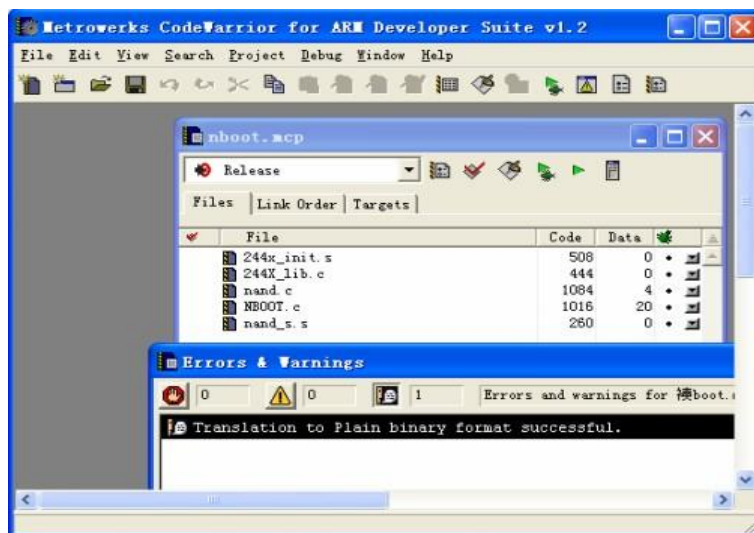
把光盘中“WindowsCE6.0”目录中的文件夹“NBOOT”文件夹复制到硬盘的某一个目录（在此为D:\work），去掉其只读属性，运行 ADS1.2 集成开发环境，点 File->Open... 打开 nboot.mcp 文件，如图。



这时点菜单 Project->Make 或者直接按 F7 键，开始编译 nboot 项目，编译完毕如图:



在 D:\work\NBOOT\nboot_Data\DebugRel 目录下会生成 nboot.bin 可执行文件，如图。



把 NBOOT 烧写到开发板的 Nand Flash。

4.2.3 在 BSP 中修改 LCD 类型及串口输出功能

说明：我们提供的 BSP 目前支持以下型号的液晶屏：

- 奇美 3.5 寸 LCD 带触摸
- NEC3.5 寸屏带触摸
- 统宝 3.5 寸 LCD 带触摸
- Sharp 8 寸 LCD 带触摸
- 7 寸屏带触摸

通过修改 mini2440\Src\Inc\options.h 头文件中 LCD_TYPE 的定义，可以选择相应的 LCD 类型：

```
//#define LCD_Q35 适用于奇美 3.5 寸 LCD  
//#define LCD_L80 适用于 Sharp 8 寸 LCD  
//#define LCD_T35 适用于统宝 3.5 寸 LCD  
//#define LCD_X35 适用于 Sony3.5 寸 LCD  
//#define LCD_A70 适用于群创 7 寸 LCD
```

提示：光盘中缺省 LCD 型号是 LCD_Q35。



在 options.h 文件中, 用户也可以修改串口的输出功能: 作为普通串口功能或者调试输出 (仅限于串口 1 和 2), 如下定义:

```
#define KITL_NONE
#define KITL_SERIAL_UART0
#define KITL_SERIAL_UART1
#define KITL_USBSERIAL
#define KITL_ETHERNET
```

这里缺省的定义是作为普通串口功能, 如果要把串口 1 作为调试信息输出使用, 则应该定义为:

```
//#define KITL_NONE
//#define KITL_SERIAL_UART0
//#define KITL_SERIAL_UART1
//#define KITL_USBSERIAL
//#define KITL_ETHERNET
```

4.2.4 制作和修改 Windows CE 启动 Logo

在前面的章节, 我们提到过:

Windows CE 系统的启动过程有两种 Logo: BootLogo 和 StarLogo。其中 BootLogo 是有 Nboot 加载显示的, 用户可以通过修改 Nboot 的源代码调整 BootLogo 的显示位置和背景色; StartLogo 则属于 BSP 的一部分, 它是一个数组文件 (StartLogo.c), 位于“mini2440\Src\Kernel\Oal”目录, 由该目录下的 init.c 文件实现加载显示, StartLogo.c 文件可以通过本光盘中的 StartLogoMaker.exe 工具制作生成。

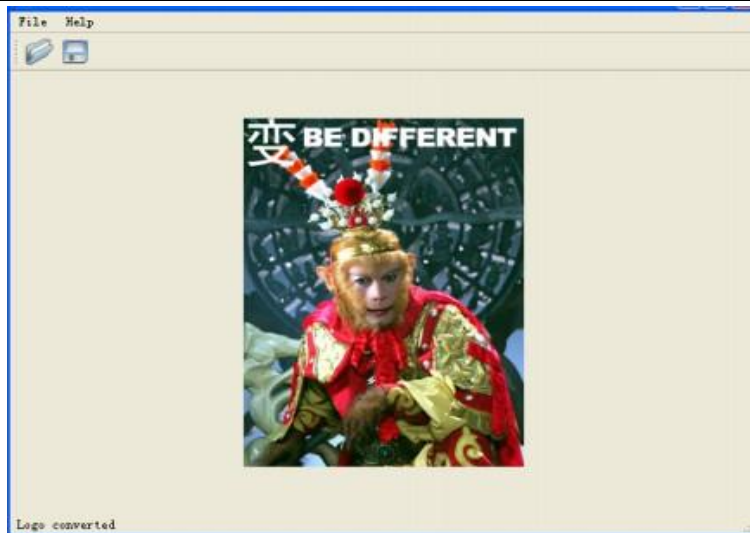
StartLogoMaker 由 Linux Logo 制作工具 LogoMaker 移植而来, 是一个“绿色软件”, 它不需要安装, 直接复制到 WindowsXP/Vista 平台即可运行, 使用它可以把 bmp, jpg, png 等格式的图片转换为 mini2440 BSP 所需要的数组文件 StartLogo.c, 使用新生成的文件替换 BSP 中的同名文件, 即可更换 WindowsCE 的启动画面, StartLogo.c 数组的头部内容如下:


```
//Automatic generated by StartLogo.exe
```

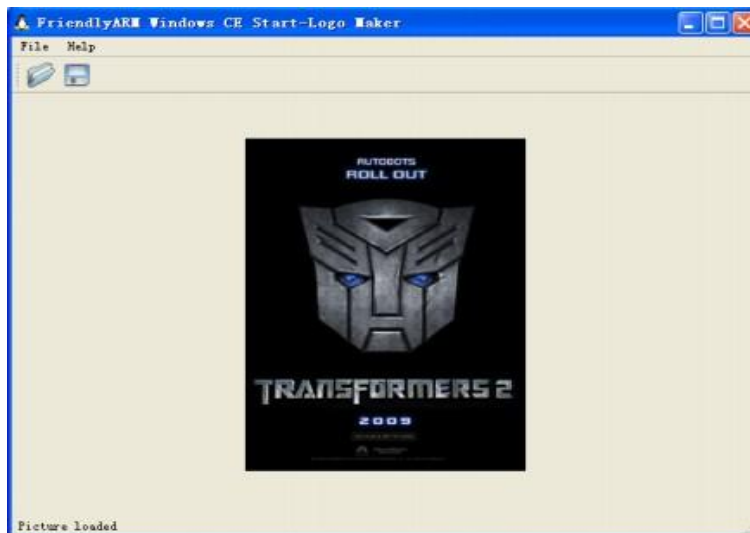
```
Static const unsigned short StartLogoData[] = {
240,320
0x965, 0x945, 0x164, 0x9C4, 0x1246, 0x22CA, 0x22A8, 0x2AA7,
```


下面是使用 StartLogoMaker.exe 制作 StartLogo.c 的步骤:

Step1: 双击运行“windows 平台工具\StartLogoMaker”中的 StartLogoMaker.exe 程序, 打开如图界面:



Step2: 点 File->Open 打开一个图片文件, 也可以在工具栏点  图标打开文件选择窗口:



Step3: 点 File->Convert, 或者点工具栏的图标  打开文件输出选择窗口:



点“确定”在相应的目录会生成 StartLogo.c 文件：



Step5: 把生成的文件替换 BSP 中的同名文件（位于 mini2440-BSP\Src\Kernel\Oal 目录中），重新编译内核，并烧写到板子中运行，即可看到自己制作的 WinCE 启动画面了：



4.2.5 创建 SDK

SDK 适用于：当开发主机只安装了 VS2005，但没有安装 Windows CE 6.0 的 Platform Builder 插件时，这时开发人员想通过 VS2005 开发 mini2440 的应用程序，就需要一个 SDK，它类似于 Embedded Visual C++ 所需的 SDK。

当你编译完缺省内核，此时可以通过 VS2005 平台创建相应的 SDK，注意：这里的 SDK 仅

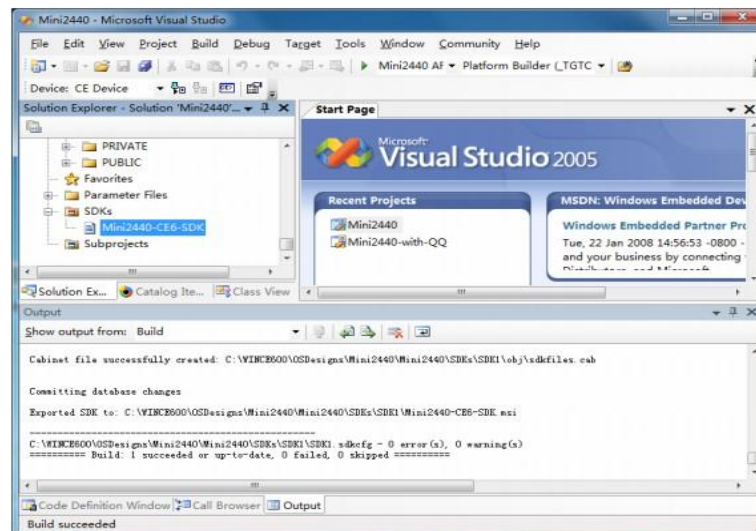


适用于 VS2005 开发环境，它不能安装到 EVC，也不能安装到 VS2008，下面是创建 SDK 的详细步骤。

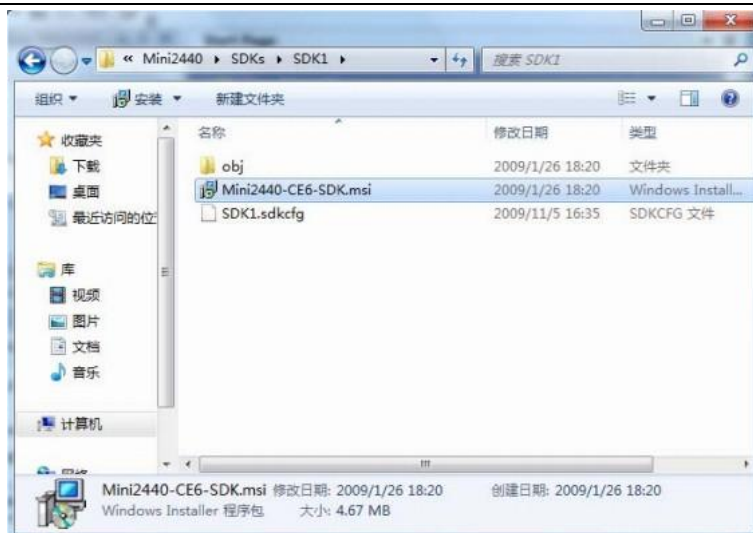
Step1: 运行 VS2005 并打开已经编译过的缺省内核示例工程 mini2440，找到如图位置，并右键点击“Mini2440-CE6-SDK”出现菜单，点 Build 开始创建 SDK



Step2: 稍等片刻，SDK 创建完毕，如图所示



Step3: 在 C:\WINCE600\OSDesigns\Mini2440\Mini2440\SDKs\SDK1 目录下，可以看到已经生成 Mini2440-CE-SDK.msi 安装文件



4.2.6 安装 SDK

要通过 VS2005 为 mini2440 开发应用程序，需要先安装刚才生产的 SDK，步骤如下：

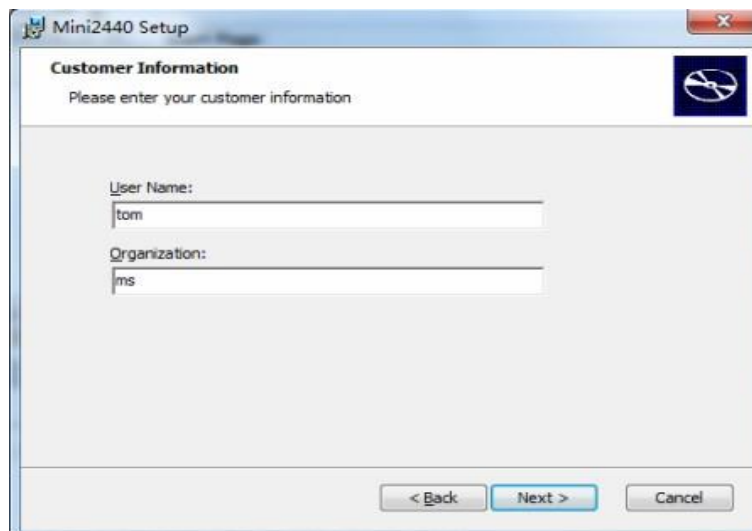
Step1: 双击运行 Mini2440-CE6-SDK.msi，出现如下界面，点“Next”继续



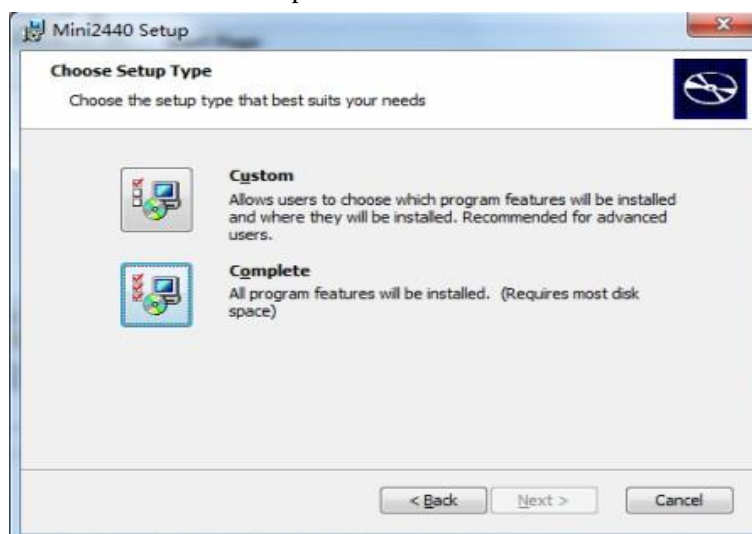
Step2: 如图选择“I accept”，点“Next”继续



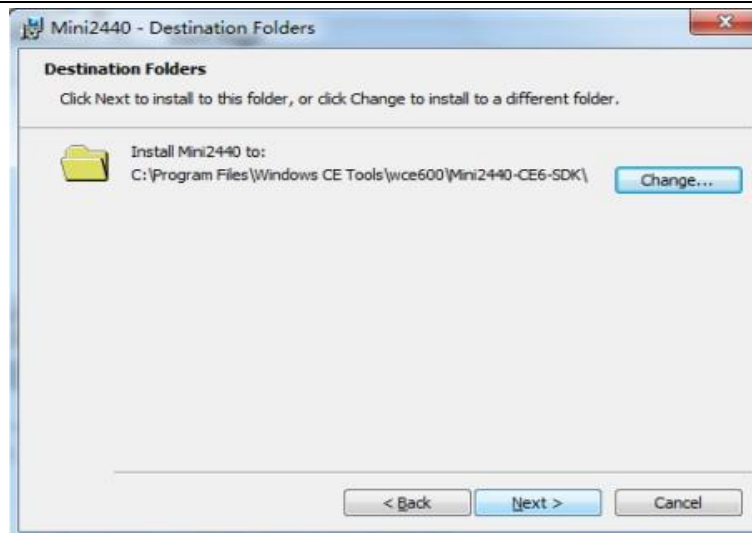
Step3: 出现如图界面，输入用户名和公司名，点“Next”继续



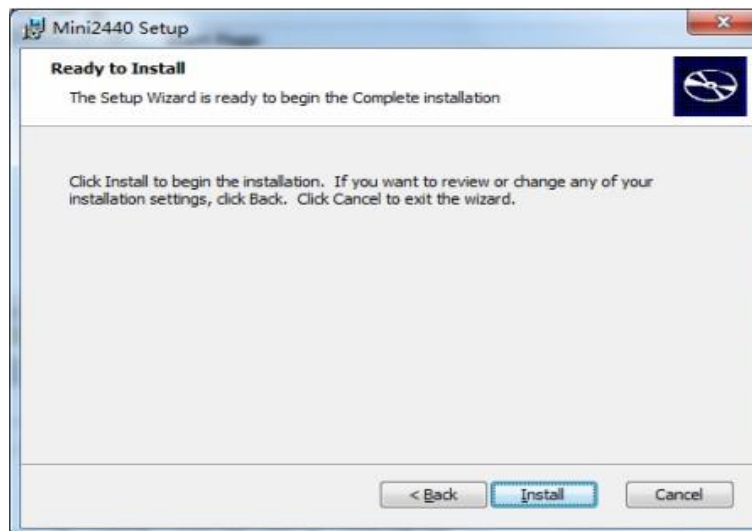
Step4: 出现如图界面，点“Complete”继续



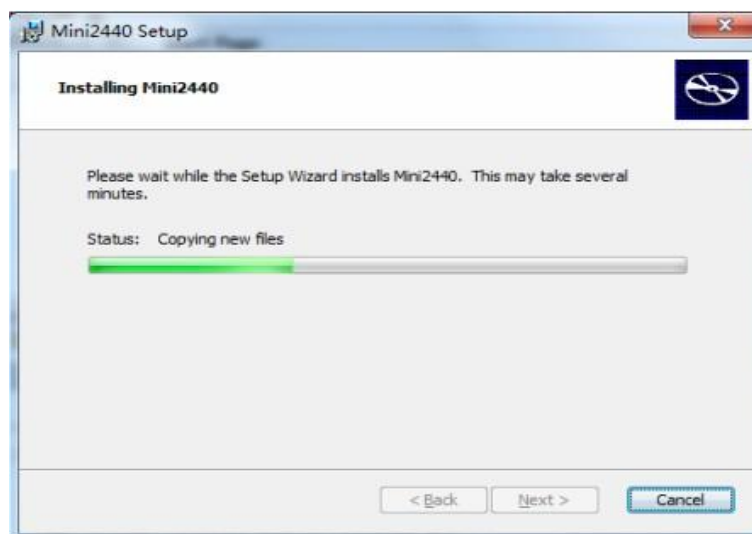
Step5: 出现如图界面，点“Next”继续



Step6: 出现如图界面，点“Install”继续



Step7: 出现如图安装进度界面，稍等片刻



Step8: 出现安装结束界面，点“Finish”结束



4.3 与 PC 同步

4.3.1 安装同步驱动与软件

Step1: 用 USB 线把 JMY901 与 PC 连接, 并打开 JMY901 电源, PC 上会弹出如下界面, 点“下一步”, 继续



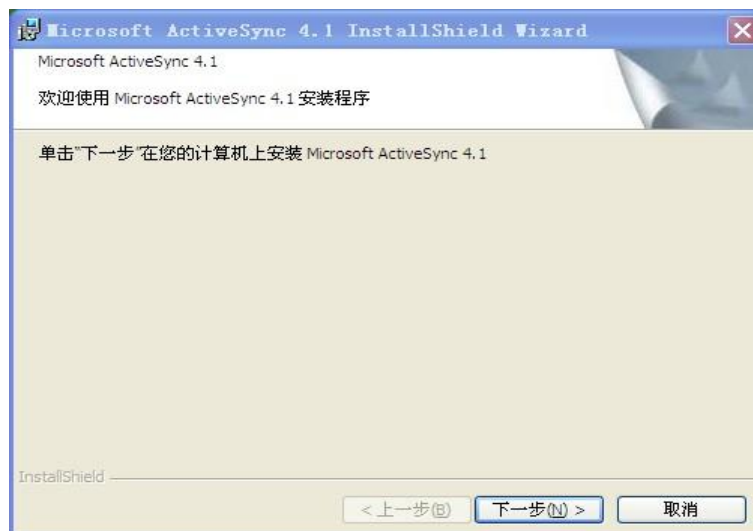
Step2: 点击“浏览”, 选择“CE 用同步 USB 驱动”, 点击“确定”, 点击“下一步”继续



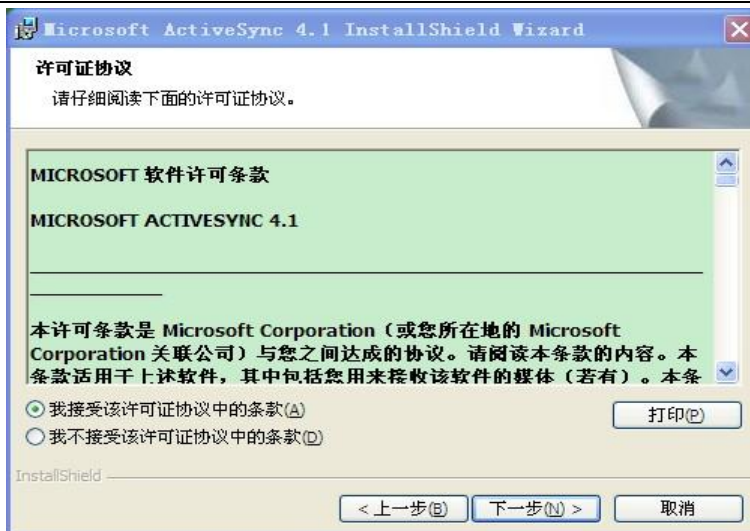
Step3: 出现如图界面，点击“完成”



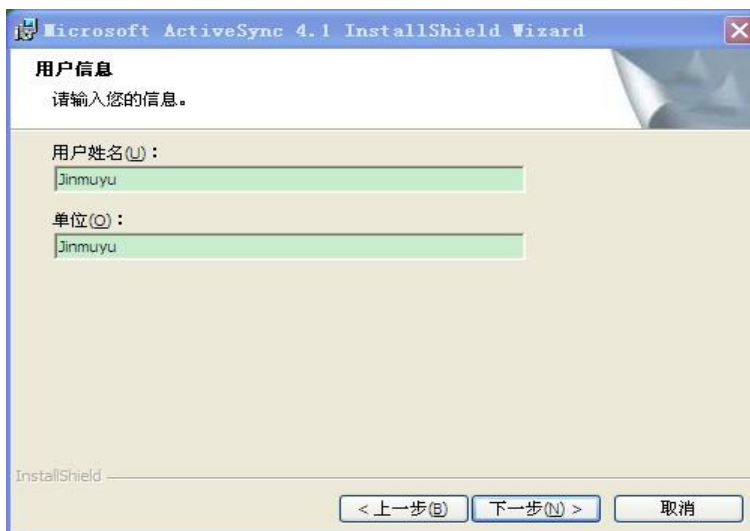
Step4: 安装 ActiveSync 同步软件，出现如下界面，点击“下一步”，继续



Step5: 出现如下界面，选择“我接受许可证协议中的条款”，点击“下一步”，继续



Step6: 出现如下界面, 输入用户姓名与单位, 点击“下一步”, 继续



Step7: 出现如下界面, 选择安装路径, 默认即可, 点击“下一步”, 继续



Step8: 出现如下界面, 点击“安装”, 开始安装软件



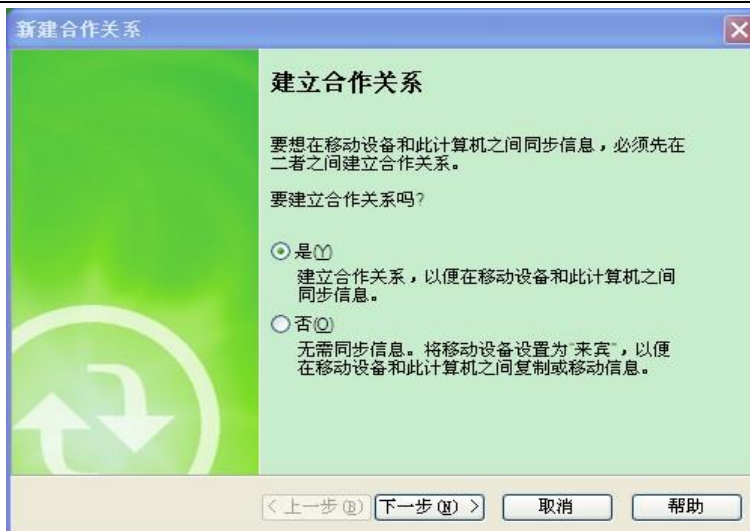
Step9: 出现如下界面，请等待



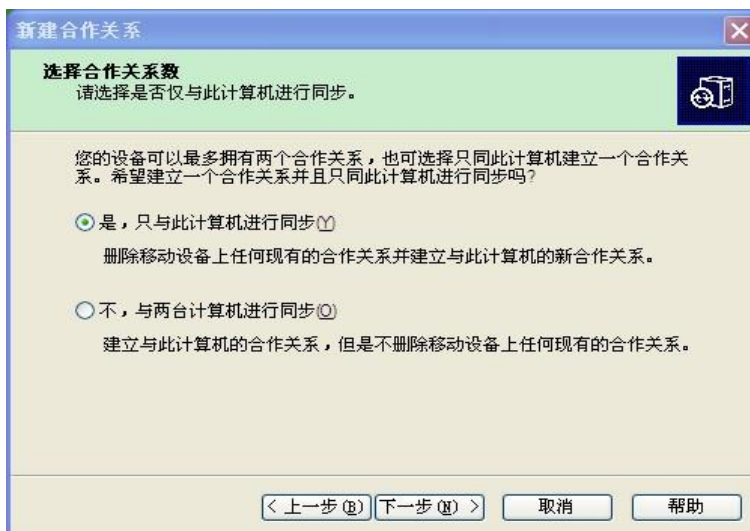
Step10: 出现如下界面，点击“完成”，结束安装



Step11: 安装完 ActiveSync 软件后，PC 上会弹出如下界面，选择“是”，点击“下一步”，继续



Step12: 出现如下界面，选择“是，...”点击“下一步”，继续



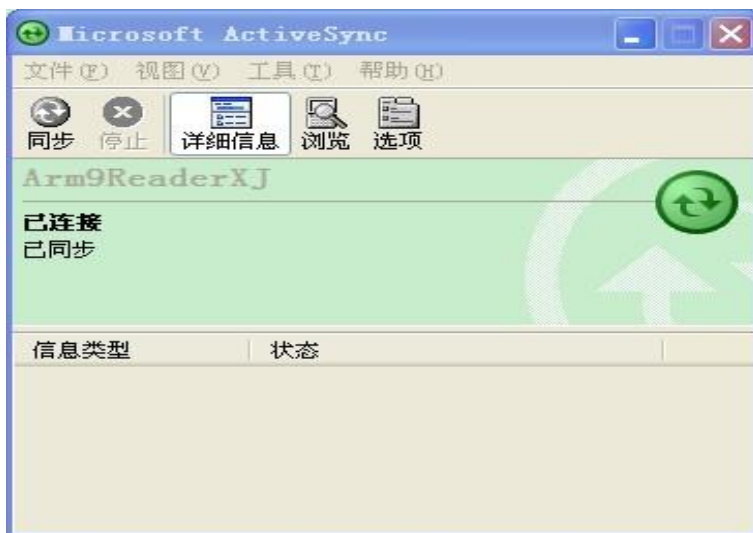
Step13: 出现如下界面，不做任何更改，点击“下一步”，继续



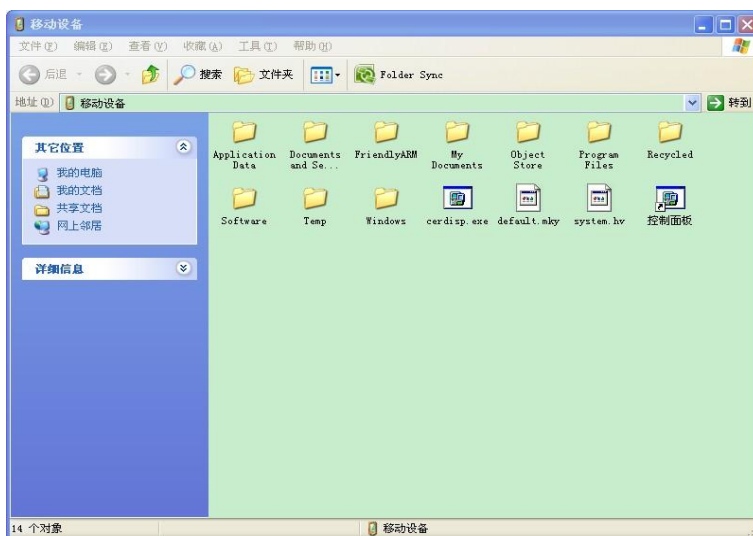
Step14: 出现如下界面，点击“完成”，结束设置



Step15: PC 自动弹出如下界面，显示“连接”，点击“浏览”



Step16: 弹出如下界面，为 WinCE6.0 的文件夹，此时 PC 与 JMY901 同步完成



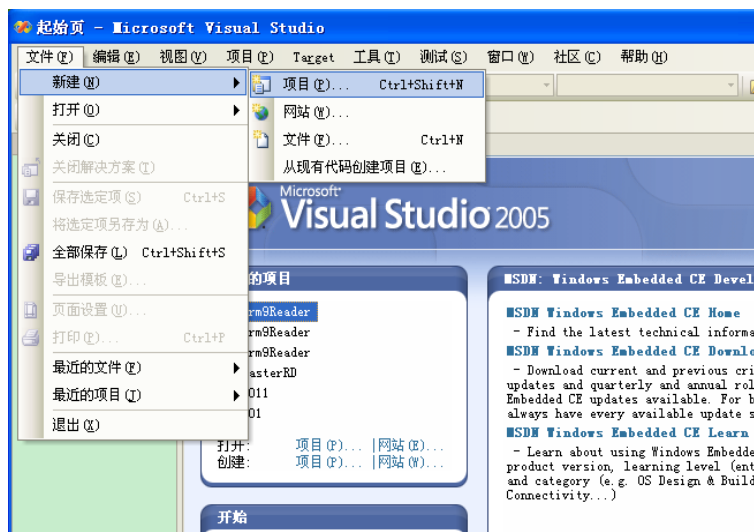


4.4 通过 VS2005 创建应用程序，并编译下载到开发板运行

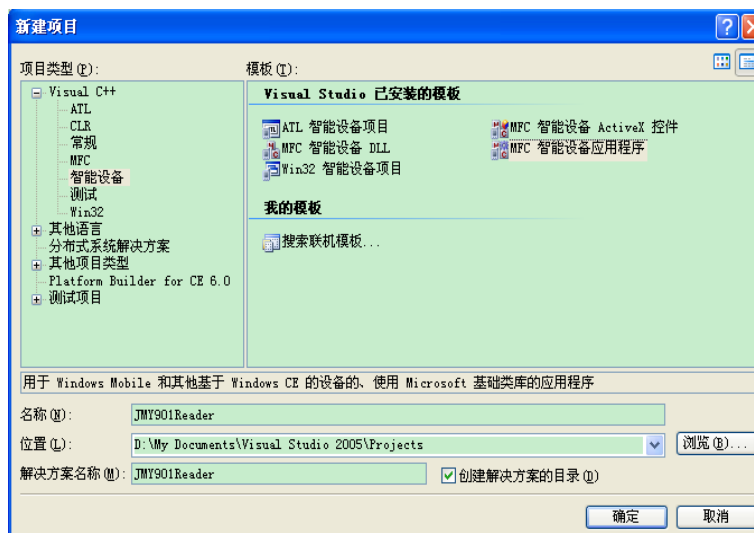
下面是使用 VS2005 的基本开发步骤：

4.4.1 创建项目

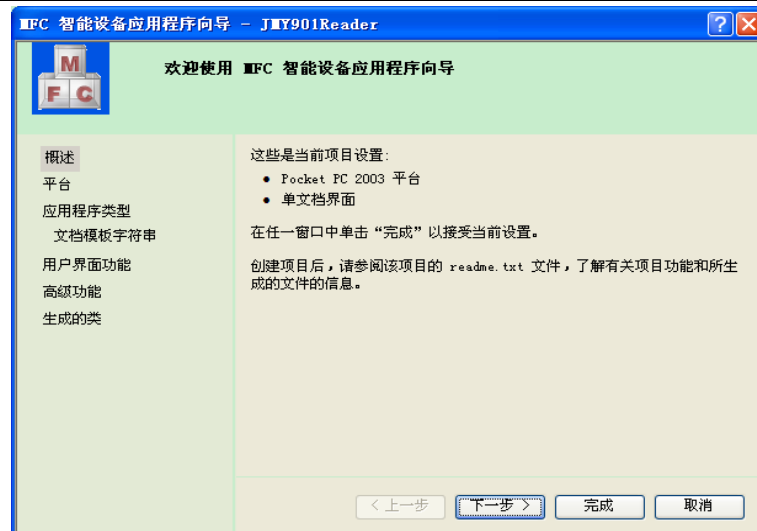
Step1: 打开运行 VS2005，点菜单文件->新建->项目，如图：



Step2: 出现“新建项目”，选择“Visual C++”展开，选择“智能设备”，在“Visual Studio 已安装的模板”里选择“MFC 智能设备应用程序”，输入名称：“JMY901Reader”，点击“确定”



Step3: 出现如下界面，点击“下一步”



Step4: 出现如下界面，点击 ，将“Pocket PC 2003”取消



Step5: 选中左边的“Mini2440-CE-SDK”，点击 ，点击“下一步”





Step6: 出现如图界面，选中“基于对话框”，在资源语言下选中“英语（美国）”，点击“下一步”



Step7: 出现如下界面，点击“下一步”



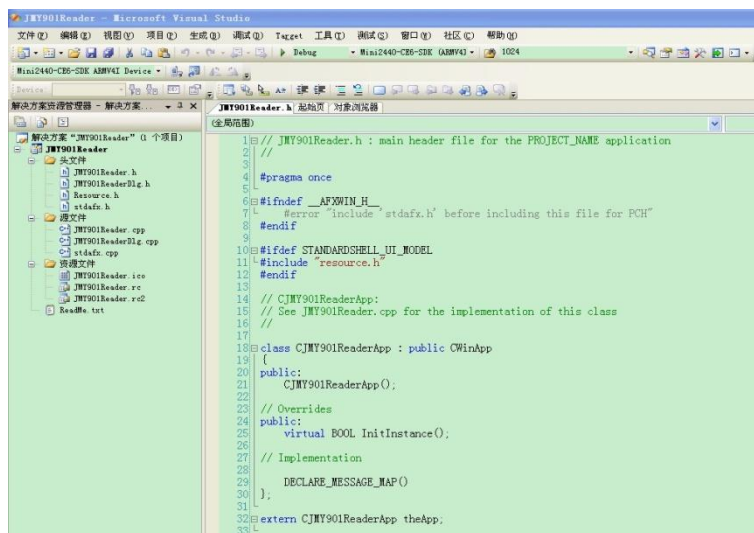
Step8: 出现如下界面，点击“下一步”



Step9: 出现如下界面，点击“完成”



Step10: 出现如下界面，WinCE 程序开发设置完成，下面就可以开始编写程序了！



到此 JMY901 开发过程介绍完毕！

此说明书在不断更新中，有问题可拨打技术支持电话：

+86 10-69559637，奚先生