修改记录

更新日期	更新类型	更新人	更新内容
2016-8-12	A	Echo	新建文档

注:

M-->修改

A -->添加

作者 Echo <echo.xjtu@gmail.com>保留本文档最终解释权

保留文档更新但不在第一时间通知用户的权利

请使用 PDF 书签阅读本文档，快速定位所需内容！

更多信息请关注

作者博客：http://blog.sina.com.cn/xjtuecho

作者微博：http://weibo.com/eth0

作者淘宝：http://shop114445313.taobao.com/

作者 github 主页：https://github.com/xjtuecho/

最新文档和设备固件请访问 github 项目主页：https://github.com/xjtuecho/UARTCAN/

ECHO Studio

1 特性介绍

XBOOT 是一款 TI C2000 平台的 bootloader 软件，配合 USBTTL，USBCAN 等硬件，可以实现 C2000 系列 DSP 固件 IAP 功能。

XBOOT 分为基础版本和定制版本，基础版本免费，用于评估，定制版本可用于商业场合。

1.1 基础版本特性

只支持 TMS320F28335 一款芯片。
只支持 30M 外部晶振。
支持一个 CAN 接口，3 个 TTL 串口，用于通讯的 GPIO 固定。
CAN 接口波特率固定 500kbps。
串口波特率固定 115200bps。
无固件加密功能。
只支持 1 个用户程序，入口地址固定。
支持用户固件更新。
支持 XBOOT 固件更新。

1.2 定制版本特性

支持 C2000 系列所有芯片（极少数 FLASH 过小的芯片除外）。
外部晶振频率可选。
支持一个 CAN 接口，3 个 TTL 串口或者 485 接口，用于通讯的 GPIO 可选。
CAN 接口波特率可设置，默认 500kbps。
串口波特率固定 115200bps。
可自带固件加密，完善的保护功能，确保用户固件安全。
可读取 FLASH 和 SRAM 内存地址。
支持最多 6 个用户程序。可以设置上电后自动运行的用户程序。
支持用户固件更新。
支持 XBOOT 固件更新。
可访问内置模拟 EEPROM。
可设置器件全球唯一 ID，用于产品序列号，固件加密等场合。

1.3 硬件资源占用

FLASH：FLASHA 和 FLASHB，地址 0x330000-0x33FF80
定时器：CpuTimer0
通信接口：SCIA（GPIO28、GPIO29），SCIB（GPIO22、GPIO23），SCIC（GPIO62、GPIO63），ECANA（GPIO30，GPIO31）
由于跳转用户程序以后 XBOOT 不再运行，因此 SRAM 与用户程序无冲突。

EPO Studio

2 工作原理

2.1 C2000 上电引导过程

C2000 系列 DSP 上电以后从内部 BOOTROM 引导，上电复位以后 DSP 跳转到 0x3FFFC0 执行内部 BOOTROM 中的代码，根据特定 GPIO 状态，判断引导模式，一般均使用 FLASH 引导模式，即 BOOTROM 中代码执行完毕之后将控制权交给 FLASH 中的代码。

整个过程如图 1 所示。

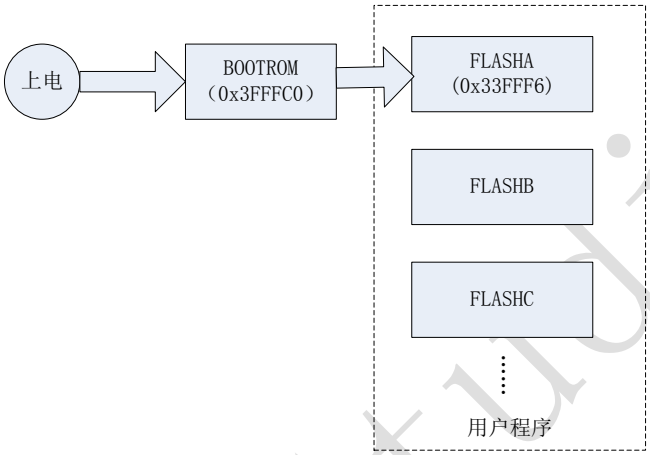

图 1 28335 上电引导过程

2.2 XBOOT 工作原理

C2000 内置 BOOTROM 通过 GPIO 来判断引导模式，单板设计时需要设计引导跳线，每次更新程序要插拔跳线，使用不便。同时内部 BOOTROM 中的代码只能执行简单的代码更新功能。

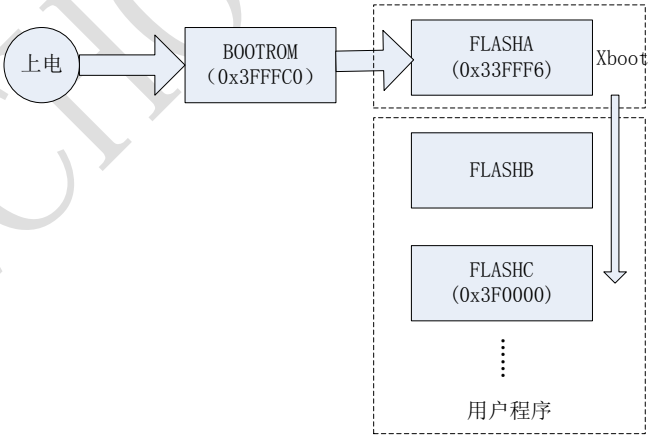

图 2 XBOOT 工作原理

实际中 C2000 系列 DSP 的 FLASH 空间往往都很大，将 FLASH 按照扇区(sector)划分为几部分，FLASHA 包含了 BOOTROM 的跳转地址 0x33FFF6，用来存放 XBOOT 程序；FLASHB 用来模拟内置 EEPROM；剩下的所有 FLASH 存放用户程序。

上电以后，FLASHA 中的 XBOOT 代码首先执行，根据串口或者 CAN 接口数据流判断进入用户程序还是 XBOOT shell，如果 0.5 秒内某个串口上连续收到 5 个字母'e'，进入 XBOOT shell，否则进入用户程序。这样设计会带来上电 0.5 秒的上电延时，一般可以接受，带来的好处是硬件设计无需额外的跳线。XBOOT shell 中可以执行用户固件 IAP 更新功能。

ECIO Studio

3 使用方法

3.1 进入 XBOOT shell 方法

XBOOT 上电启动以后在 0.5 秒内检测串口输入，如果收到连续 5 个字母“e”，便进入 XBOOT shell；否则访问模拟 EEPROM 获取用户程序入口点，如果用户程序有效，执行用户程序，否则进入 XBOOT shell。

使用超级终端连接串口，波特率 115200，8 位数据，1 位停止，无校验，无流控。按住键盘上的字母“e”，点击控制板上的复位按键，或者给控制板重新上电。可以进入 XBOOT shell。

CAN 接口借助 UARTCAN 或者 USBCAN 硬件，设置工作模式为桥接模式，其余使用方法与串口完全相同。

超级终端为 Windows XP 自带程序，通过“开始->所有程序->附件->通讯->超级终端”打开。Windows7 系统不带超级终端，可以将 Windows XP 系统的超级终端软件拷贝过去直接使用。

3.2 更新用户软件详细步骤

首先连接 XBOOT shell，使用 empty 命令查看要写入的 FLASH 是否为空，如果不为空，使用 erase x 命令擦除软件所在 FLASH，其中 x 为 FLASH 名字，取值 a、b、c….

使用 ymodem 命令更新用户软件，请参考 4.4 内容。

注意：erase a 命令将擦除包括 XBOOT 在内所有 FLASH 内容，也包括加密密码，擦除所有 FLASH 内容以后掉电之前，XBOOT 仍然可以运行并且更新程序，如果擦除以后掉电，只能使用仿真器将 XBOOT 写入 DSP。

将用户代码写入未擦除的 FLASH 运行结果将是不确定的。

4 基础命令

本节对基础版 XBOOT 提供的命令进行描述，基础命令足以完成 IAP 功能。

4.1 empty

FLASH 查空命令，无参数，用来检测用户 FLASH 是否为空。

```
empty
 FLASHA is NOT empty @ 0x338000.
 FLASHB is NOT empty @ 0x330000.
 FLASHC is empty @ 0x328000.
 FLASHD is empty @ 0x320000.
 FLASHE is empty @ 0x318000.
 FLASHF is empty @ 0x310000.
 FLASHG is empty @ 0x308000.
 FLASHH is empty @ 0x300000.
```

图 3 empty 命令执行结果

注意：28335 的 FLASH 最小擦除单位是一个扇区，写入之前必须确保为空，否则需要先进行擦除。

4.2 erase

FLASH 擦除命令。

erase 命令接受一个字符串参数：abcdefgh，分别对应 8 个 FLASH 扇区。可以同时指定

ECHO Studio

多个 FLASH 扇区，如 erase bcd 将会擦除三个 flash 扇区。

注意：erase a 命令将会擦除 FLASHA，也就是 XBOOT 本身所在的 FLASH 扇区，擦除 XBOOT 以后，芯片成为空片，DSP 也将解除加密，如果掉电，只能使用仿真器将 XBOOT 写入 DSP。

```
erase b
 erase flash sector 0x02 OK.
empty
 FLASHA is NOT empty @ 0x338000.
 FLASHB is empty @ 0x330000.
 FLASHC is empty @ 0x328000.
 FLASHD is empty @ 0x320000.
 FLASHE is empty @ 0x318000.
 FLASHF is empty @ 0x310000.
 FLASHG is empty @ 0x308000.
 FLASHH is empty @ 0x300000.
```

图 4 erase 命令执行结果

4.3 reboot

重启系统。带一个延时参数，单位 ms。执行 reboot 1000 即延时 1 秒以后重新启动，无参数默认 10ms 延时以后重新启动。

4.4 ymodem

使用 ymodem 协议更新用户软件。

执行 ymodem 命令，超级终端显示字符“C”以后，选择菜单发送->发送文件，弹出图 5 对话框，文件名选择用户程序的 hex 文件，协议选择 Ymodem，点击发送即可，命令执行成功以后如图 7 所示。

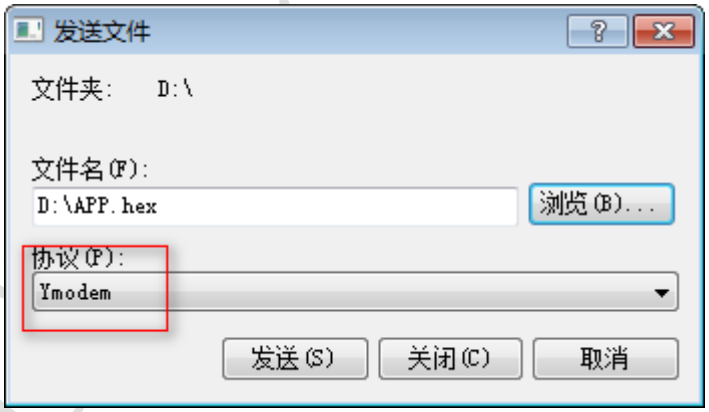

图 5 选取发送的 hex 文件

EC Studio

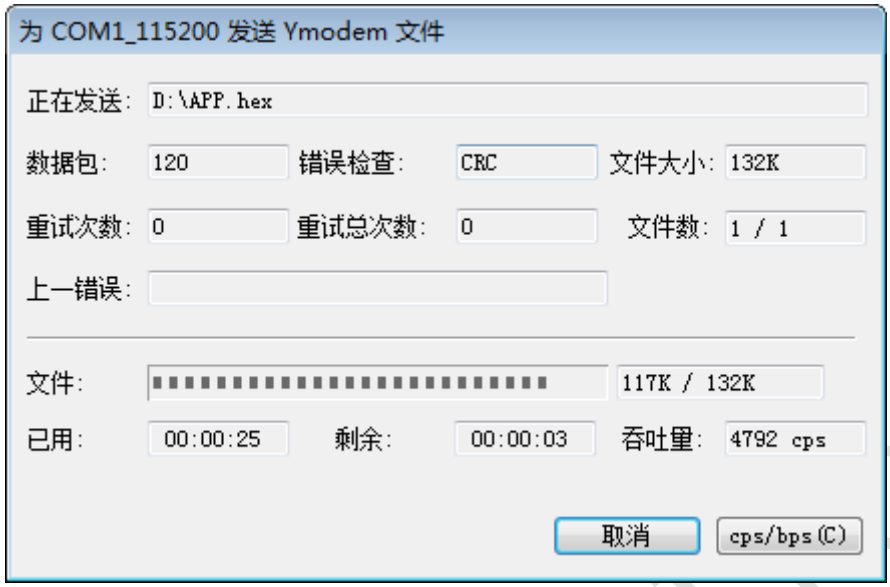

图 6 APP 固件发送过程

```
ymodem
 warning: flash address 0x330000 is not empty, do not use it.
 ymodem update firmware, press A to abort ...
 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
  Update firmware OK!
  File Name: APP.hex
  File Size: 134957
  End Addr:  0x32ED73
```

图 7 ymodem 命令执行结果

执行 ymodem 命令以后，超级终端显示 C 时，可以按字母 a 取消命令。

用户发送的 APP 固件，不能使用 FLASHA 或者 FLASHB，否则将会损坏 XBOOT 固件。

4.5 help

显示 XBOOT 帮助信息。

```
help
 empty -> empty Check flash sector is empty or not.
 erase -> erase [abcdefgh] Erase flash sector.
 entry -> entry [hex addr] Get/Set user app entry.
 memrd -> memrd [hex addr] [hex len] Show Memory.
 goto -> goto [hex addr] to execute.
 eeprom -> eeprom [load|save|read|write] [addr] [data]
 uuid -> uuid [128bits hex string] Show/Setup UUID.
 reboot -> reboot [delay ms] Restart xboot.
 ymodem -> update user APPs.
 help -> help Info.
 version -> version display XBOOT Info.
```

图 8 help 命令输出

注意：基础版本不支持 entry、memrd、goto、eeprom、uuid 命令。

4.6 version

显示 XBOOT 版本和版权信息。

```
version
 xboot v16.08.12 SN: 66E4DA6F807F4B0AA66DB40077386E0C
 for TMS320F28335 by ECHO Studio <echo.xjtu@gmail.com>.
 All Rights Reserved.
```

图 9 version 命令输出

注意：没有写入唯一 ID 的 DSP 芯片，序列号 SN 内容为全 FF。

5 高级命令

定制版 XBOOT 可以支持高级功能，通过本节提供的高级命令进行支持。

5.1 entry

设置、查看用户程序入口点。

默认无参数查看当前用户程序入口点，带一个参数为设置用户程序入口点。

```
entry
 user app entry is 0x328000.
entry 320000
 set user app entry to 0x320000 .
entry
 user app entry is 0x320000.
```

图 10 entry 命令执行结果

5.2 memrd

读取 DSP 内存。内存地址包括 SRAM 或者 FLASH。

memrd 命令接受两个参数，第一个为要读取的内存地址，第二个为要读取的数据长度，默认 128。全部为十六进制。

```
memrd 338000

0x338000 0000 0030 8000 0000 8000 0030 8000 0000
0x338008 0000 0031 8000 0000 8000 0031 8000 0000
0x338010 0000 0032 8000 0000 8000 0032 8000 0000
0x338018 0000 0033 8000 0000 8000 0033 8000 0000
0x338020 0080 0040 0020 0010 0008 0004 0002 0001
0x338028 FE12 1E46 A044 A842 8A56 7640 84AE 964D
0x338030 5200 EC03 FFEF 009E 284D 03E7 0642 0746
0x338038 1901 1E48 C442 8F30 0000 A8A9 0FA6 660D
0x338040 C448 8F33 FFFF A8A9 0FA6 6807 0642 FF0F
0x338048 0060 1E4A 2B4E 6F16 C442 8F38 0400 A8A9
0x338050 0FA6 660E C448 8F38 07FF A8A9 0FA6 6808
0x338058 0642 FF0A 0E01 1E4A 56BF 104E 6F03 9A0C
0x338060 6F70 7640 8219 7622 9A03 F4A9 0A8D 761A
0x338068 0201 1E50 0646 0F50 685C 5C4E 064A FF69
0x338070 7640 8290 964B 2B51 9251 522D 6728 BE00
0x338078 761F 0333 0624 0FA6 EC03 C524 3E67 8A44
```

图 11 memrd 命令执行结果

5.3 goto

跳转到新地址执行。

接受一个复位向量作为参数。28335 平台下 goto 33fff6 将会重启 XBOOT，如图 12。

```
goto 33FFF6
 goto address 0x33FFF6 to execute...
```

图 12 goto 命令实现 XBOOT 重启

5.4 eeprom

读写 DSP 内部模拟 EEPROM。带四个子命令：read、write、load、save。

读 eeprom：eeprom read [地址] [长度]，其中地址为必要参数，长度可不填，默认 128 字，从内部缓存读取数据。

写 eeprom：eeprom write [地址] [数据]，其中地址和数据均为必要参数，数据写入内部

缓存。

加载 eeprom：eeprom load，数据从 FLASH 加载到内部缓存。

存储 eeprom：eeprom save，eeprom 数据从内部缓存写入 FLASH，掉电保存。

读取内部 eeprom 内容执行结果参考图 13。

内置 eeprom 总共 255 个 16 位字，前 16 个字预留给 XBOOT 自身使用，其余用户程序可用。

```
eeprom read 0

0x0000  8000 0032 FFFF 01F4 FFFF FFFF FFFF FFFF
0x0008  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0010  0002 0001 C200 0001 4B00 0000 FFFF FFFF
0x0018  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0020  0002 0001 C200 0001 4B00 0000 FFFF FFFF
0x0028  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0030  007D FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0038  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0040  0001 0001 C200 0001 4B00 0000 FFFF FFFF
0x0048  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0050  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0058  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0060  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0068  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0070  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
0x0078  FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF
```

图 13 读取内部模拟 eeprom

5.5 uuid

查看或者设置芯片的唯一 ID。芯片的唯一 ID 可以用作芯片序列号、单板序列号、设备序列号。还可以用来加密固件。许多现代 MCU 如 STM8、STM32、STC 本身提供了唯一 ID，TI C2000 系列 MCU 没有提供唯一 ID，XBOOT 利用 OTP 存储空间来实现唯一 ID 功能。

不带参数的 uuid 命令可以显示当前单板的 uuid，如下所示：

```
uuid
BBF2CE53C8B54062B42C3E8423AD9E88
```

如果单板没有设置 uuid，uuid 命令无输出。

未写入 uuid 的单板，可以使用 uuid 命令写入 uuid，如下所示。

```
uuid BBF2CE53C8B54062B42C3E8423AD9E88
```

uuid 介绍及生成方法参考 7。

注意：一块单板 uuid 只能写入一次，要保证写入的 uuid **每次都是重新生成**的，避免重复。uuid 写入以后，重新擦写 FLASH 均不受影响，更换 uuid 的唯一方法是更换新的芯片。

6 用户软件编写指南

6.1 复位向量

TMS320F28335 FLASH 入口地址为 0x33FFF6，这个地址位于 FLASHA，已经被 XBOOT 占用。用户代码需要使用另外的复位向量。XBOOT 默认会将入口地址设置到 FLASHC 开头两个字，地址 0x328000，上电后 XBOOT 可以跳转到 FLASHC 执行用户代码。

用户代码需要将复位向量放到所在 FLASH 扇区开始两个字。如图 14 所示，所有用户代码放在 FLASHC，复位向量占用 FLASHC 开始两个字。

```
85    FLASHC      : origin = 0x328002, length = 0x007FFE
86    BEGIN       : origin = 0x328000, length = 0x000002
87    FLASHB      : origin = 0x330000, length = 0x008000
88    FLASHA      : origin = 0x338000, length = 0x007F80
89 /* BEGIN       : origin = 0x33FFF6, length = 0x000002 */
90    CSM_RSVD    : origin = 0x33FF80, length = 0x000076
```

图 14 用户软件复位向量（28335）

定制版 XBOOT 固件可以使用 entry 命令重新设置复位向量，实现多 APP 支持。

由于 XBOOT 本身使用 FLASHA 和 FLASHB，用户程序不能占用这两处 FLASH，编写 cmd 文件时需要注意，用户程序编译后注意检查 map 文件，确认两块 FLASH 没有使用，见图 15。

```
12         name            origin    length      used     unused
13 ----------------------  --------  ---------  --------  --------
14 PAGE 0:
15   ZONE0                 00004000  00001000   00000000  00001000
16   RAML0L1               00008000  00002000   000013f8  00000c08
17   RAML2L3               0000a000  00002000   00000000  00002000
18   ZONE6                 00100000  00100000   00000000  00100000
19   ZONE7A                00200000  0000fc00   00000000  0000fc00
20   FLASHH                00300000  00008000   00000000  00008000
21   FLASHG                00308002  00007ffe   00000000  00007ffe
22   FLASHF                00310000  00008000   00000000  00008000
23   FLASHE                00318002  00007ffe   00000000  00007ffe
24   FLASHD                00320000  00008000   00000000  00008000
25   BEGIN                 00328000  00000002   00000002  00000000
26   FLASHC                00328002  00007ffe   000066fd  00001901
27   FLASHB                00330000  00008000   00000000  00008000
28   FLASHA                00338000  00007f80   00000000  00007f80
```

图 15 用户软件编译后不能占用 FLASHA 和 FLASHB

6.2 生成 hex 文件

使用 hex2000 命令行工具将 CCS 编译生成的.out 文件转化为 hex 文件。命令如下：

hex2000 --intel -romwidth 16 -memwidth 16 TEST.out

其中 TEST.out 为需要转化的.out 文件，根据实际需要进行替换。

.hex 文件实际为文本文件，可以使用文本编辑器打开查看。

CCS5 可以设置编译完成自动输出 hex 文件，设置方法如下：

在工程名上按 Alt+Enter 打开项目属性设置页，然后按照图 16 设置即可。

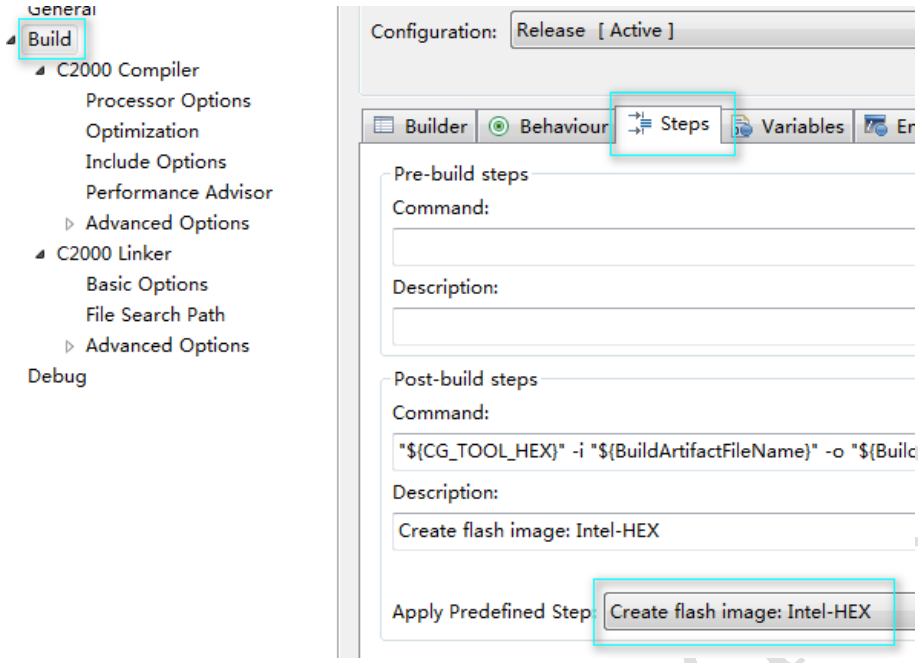

图 16 CCS5.5 设置自动输出 hex 文件

7 uuid 生成指南

UUID 含义是通用唯一识别码 (Universally Unique Identifier)，是指在一台机器上生成的数字，它保证对在同一时空中的所有机器都是唯一的。通常使用一个 16 进制字符串表示，如：BBF2CE53C8B54062B42C3E8423AD9E88。

7.1 Linux

Linux 平台下可以使用 uuidgen 命令生成 uuid，见图 17，注意输入时去掉中间的'-'分隔符。

```
echo@ubuntu:~$ uuidgen
148e3f58-7d9b-4fa8-9d45-6671210e24ca
```

图 17 Linux 下使用 uuidgen 命令生成 uuid

7.2 Windows

Windows 平台下可以使用 PDFFactory 软件来生成 uuid，见图 18。

ECIO studio

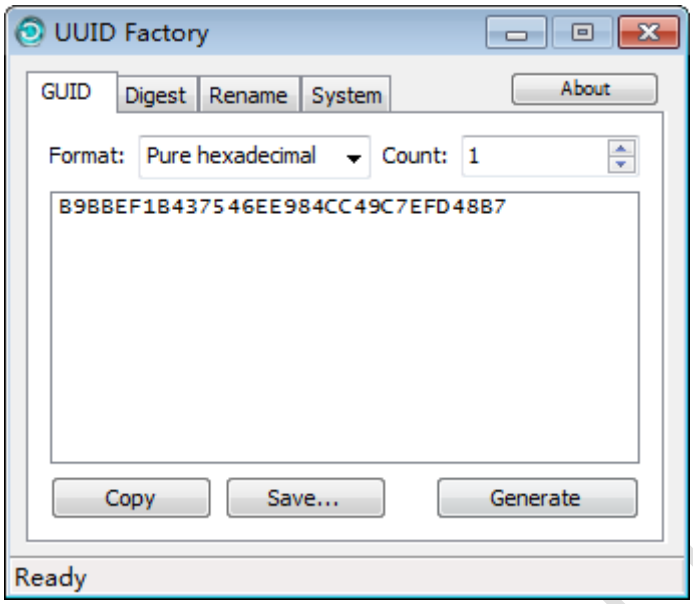

图 18 Windows 下使用 UUIDFactory 生成 UUID

8 免责条款

8.1 基础版

仅仅用来评估，使用方法请参考本文档，作者不提供任何技术支持与保证。

8.2 定制版

可用于商业场合。使用方法请参考本文档，同时作者会尽力满足客户需求，并提供有限的技术支持。

技术支持仅仅限于 XBOOT 固件本身，用户需要对最终产品测试负责，作者不提供用户产品的技术支持与保证。

用户使用 XBOOT 默认接受以上条款。

ECHO studio