

PAN1070 eFuse 例程说明文档

PAN-CLT-VER-A0, Rev 1.0

PANCHIP

PanchipMicroelectronics

www.panchip.com

修订历史

版本	修订日期	描述
V1.0	2023-10-23	初始版本创建

PANCHIP

目录

第 1 章 测试内容	4
1.1 测试内容	4
1.2 环境准备	4
1.2.1 软件环境	4
1.2.1.1 待测代码	4
1.2.1.2 软件工具	4
1.2.2 硬件环境	4
第 2 章 测试流程	6
2.1 环境配置	6
2.1.1 测试程序编译烧录	6
2.1.2 硬件接线	6
2.2 eFuse 工作流程	6
2.3 测试程序初始化	6
2.4 基本功能验证	6
2.4.1 eFuse 所有寄存器默认状态	6
2.4.2 读写 1 字节数据	7
2.4.3 读写多个字节数据	7
2.4.4 读写用户区域 (0x7C ~ 0x7F) 的数据	8
2.4.5 遍历读取整块 eFuse	9
第 3 章 使用注意事项	11

第1章 测试内容

1.1 测试内容

PAN1070 包含 1 个 128 Byte 的物理 eFuse，在普通模式下，eFuse 有些区域是只读的、有些区域是可读写的，有些区域是不可读写的，具体如下：

Address	Permission	Description
0x00 ~ 0x1A	不可读写	Secure 相关内容
0x1B	只读	Secure 控制开关
0x1C	只读	Panchip 保留区域
0x1D ~ 0x7B	只读	Panchip CP/FT 数据区域
0x7C ~ 0x7F	可读写	用户自定义区域 (4 Bytes)

本测试例程 cases 如下：

1. 寄存器默认值
2. 读 1 字节数据
3. 读多个字节数据
4. 读写用户区域的数据
5. 遍历读取整块 eFuse

1.2 环境准备

1.2.1 软件环境

1.2.1.1 待测代码

测试工程文件：

<PAN1070-DK>\03_MCU\mcu_samples\EFUSE\keil\EFUSE.uvprojx

测试源文件目录：

<PAN1070-DK>\03_MCU\mcu_samples\EFUSE\src

1.2.1.2 软件工具

- 1、SecureCRT（用于显示 PC 与 Test Board 的交互过程，打印 log 等）

1.2.2 硬件环境

- 1、PAN1070 COB 板 1 块
 - a) UART0（测试交互接口，TX：P16，RX：P17，波特率：921600）
 - b) SWD（用来调试和烧录程序，SWDCLK: P00，SWDIO: P01）
- 2、JLink（SWD 调试与烧录工具）

PANCHIP

第2章 测试流程

2.1 环境配置

2.1.1 测试程序编译烧录

打开测试工程，确保可以编译通过。

2.1.2 硬件接线

无需特别接线，将 Log UART 连至 PC 即可。

2.2 eFuse 工作流程

参考 User Manual 文档。

2.3 测试程序初始化

硬件连线完成并烧录测试程序后，Test Board 上电，观察 Debug Port 是否正常打印测试主菜单。

CPU @ 48000000HZ

```
-----
PAN1070 eFuse Sample Code.
-----
Press key to start specific testcase:
Input '0'   Testcase 0: Register Default value check.
Input '1'   Testcase 1: Read One Byte from eFuse.
Input '2'   Testcase 2: Read Multiple Bytes from eFuse.
Input '3'   Testcase 3: Read data from User Area of eFuse.
Input '4'   Testcase 4: write data to User Area of eFuse.
Input '5'   Testcase 5: Read the whole eFuse table.
-----
```

2.4 基本功能验证

2.4.1 eFuse 所有寄存器默认状态

在主菜单下，输入 **'0'** 命令 打印所有寄存器默认值：

测试目的：

检查所有 eFuse 相关寄存器复位 Default 值状态。

测试预期：

寄存器默认值应和 Datasheet 上 eFuse 模块默认值一致。

测试现象：

```
0
eFuse Register Default Values:
-----
EF_CTL           = 0x00000000
EF_ADDR          = 0x00000000
EF_DAT           = 0x00000000
EF_VDD           = 0x00000000
EF_CMD           = 0x00000002
EF_TRG           = 0x00000000
EF_PROG_TIM1     = 0x00104e06
EF_PROG_TIM2     = 0x009c0884
EF_PROG_TIM3     = 0x00000c27
EF_READ_TIM4     = 0x0030180f
EF_READ_TIM5     = 0x003c100c
EF_OP_ERROR      = 0x00000000
EF_VERIFY_DEBUG1 = 0x00000000
EF_VERIFY_DEBUG2 = 0x00000000
EF_VERIFY_DEBUG3 = 0x00000000
EF_VERIFY_DEBUG4 = 0x00000000
EF_FLASH_PERMISSION = 0x00000000
-----
EFUSE Test OK, Success case: 0
```

测试分析:

参考 eFuse 设计手册对比寄存器信息，均与手册一致，符合预期。

2.4.2 读写 1 字节数据

测试目的:

验证使用 EFUSE_ReadByte() API 读 1 字节数据是否正常。

测试预期:

能够成功读取 eFuse 区域的数据。

测试现象:

Test Board 端，输入 ‘1’ 命令，从 SecureCRT 界面可看到弹出输入读取地址的提示，输入 ‘0x1B’ 后按回车键，可以看到成功读取 eFuse 0x1B 地址的数据，其值也为 0x00:

```
1
Read one byte of data from eFuse.
Now input address (in hexadecimal format, e.g. 0x7C):
0x1b
eFuse Read One Byte Success!
      address = 0x1b, data = 0x00
EFUSE Test OK, Success case: 1
```

测试分析:

eFuse 0x1B 地址位于只读区域，因此读取数据可以成功，现象符合预期。

2.4.3 读写多个字节数据

测试目的:

验证使用 EFUSE_Read() API 读多字节数据是否正常。

测试预期:

能够成功读取 eFuse 只读区域的多个字节数据。

测试现象:

Test Board 端, 输入 ‘2’ 命令, 从 SecureCRT 界面可看到弹出输入读取起始地址和长度的提示, 输入地址 ‘0x1E’ 后按回车键, 再输入长度 ‘0x10’ 后按回车键, 可以看到成功读取 eFuse 0x1E ~ 0x2D 地址共 16 字节的数据, 值为全 0:

```
2
Read multiple data from eFuse.
Now input start address (in hexadecimal format, e.g. 0x1E):
0x1e
Now input number of bytes to read (in hexadecimal format, e.g. 0x10):
0x10
eFuse Read Multiple Data Success!
    start address = 0x1e, length = 0x10
    data = 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
EFUSE Test OK, Success case: 2
```

测试分析:

eFuse 0x1E ~ 0x2D 地址位于只读区域, 因此读取数据可以成功, 现象符合预期。

2.4.4 读写用户区域 (0x7C ~ 0x7F) 的数据

测试目的:

验证使用 EFUSE_UserRead() 和 EFUSE_UserWrite() API 读写可读写区域行为是否正常。

测试预期:

能够成功读写 eFuse 用户区域的数据。

测试现象:

Test Board 端, 输入 ‘3’ 命令, 从 SecureCRT 界面可看到弹出输入读取地址的提示, 输入地址 ‘0x7C’ 后按回车键, 再输入读取长度 ‘0x04’ 后按回车键, 可以看到成功读取到数据, 值为全 0:

```
3
Read multiple data from User Area in eFuse.
Now input start address (in hexadecimal format, e.g. 0x7C):
0x7c
Now input number of bytes to read (in hexadecimal format, e.g. 0x04):
0x04
eFuse Read User Area Success!
    start address = 0x7c, length = 0x04
    data = 0x00 0x00 0x00 0x00
EFUSE Test OK, Success case: 3
```

再输入 ‘4’ 命令, 从 SecureCRT 界面可看到弹出输入写入地址和数据的提示, 输入地址 ‘0x7C’ 后按回车键, 然后输入数据 ‘01020304’ 后再按回车键, 可以看到写入成功的提示:


```
4
Write multiple data to User Area in eFuse.
Now input start address (in hexadecimal format, e.g. 0x7C):
0x7c
Now input data to write to eFuse (in hexadecimal format, e.g. abCDfe99):
01020304
    eFuse Write, start address = 0x7c, length = 0x04
    data = 0x01 0x02 0x03 0x04
eFuse Write User Area Success!
EFUSE Test OK, Success case: 4
```

再次输入 ‘3’ 命令，再次读取 ‘0x7C ~ 0x7F’ 地址的内容，可以看到读取到的数据与写入的一致：

```
3
Read multiple data from User Area in eFuse.
Now input start address (in hexadecimal format, e.g. 0x7C):
0x7c
Now input number of bytes to read (in hexadecimal format, e.g. 0x04):
0x04
eFuse Read User Area Success!
    start address = 0x7c, length = 0x04
    data = 0x01 0x02 0x03 0x04
EFUSE Test OK, Success case: 3
```

测试分析：

eFuse 0x7C ~ 0x7F 地址位于可读写区域，供用户自定义使用，读写均可以成功，现象符合预期。

2.4.5 遍历读取整块 eFuse

测试目的：

列出整块 eFuse Table 的内容，并标注可读与不可读区域。

测试预期：

能够正确列出 eFuse Table 及各个地址的读取权限。

测试现象：

Test Board 端，输入 ‘5’ 命令，由 Log 看到列出了整块 eFuse Table 的内容及读取权限情况：

```

5
eFuse full table:
Addr:Data      Addr:Data      Addr:Data      Addr:Data
0x00:0x00(N/A) 0x01:0x00(N/A) 0x02:0x00(N/A) 0x03:0x00(N/A)
0x04:0x00(N/A) 0x05:0x00(N/A) 0x06:0x00(N/A) 0x07:0x00(N/A)
0x08:0x00(N/A) 0x09:0x00(N/A) 0x0a:0x00(N/A) 0x0b:0x00(N/A)
0x0c:0x00(N/A) 0x0d:0x00(N/A) 0x0e:0x00(N/A) 0x0f:0x00(N/A)
0x10:0x00(N/A) 0x11:0x00(N/A) 0x12:0x00(N/A) 0x13:0x00(N/A)
0x14:0x00(N/A) 0x15:0x00(N/A) 0x16:0x00(N/A) 0x17:0x00(N/A)
0x18:0x00(N/A) 0x19:0x00(N/A) 0x1a:0x00(N/A) 0x1b:0x00(OK)
0x1c:0x00(OK)  0x1d:0x00(OK)  0x1e:0x00(OK)  0x1f:0x00(OK)
0x20:0x00(OK)  0x21:0x00(OK)  0x22:0x00(OK)  0x23:0x00(OK)
0x24:0x00(OK)  0x25:0x00(OK)  0x26:0x00(OK)  0x27:0x00(OK)
0x28:0x00(OK)  0x29:0x00(OK)  0x2a:0x00(OK)  0x2b:0x00(OK)
0x2c:0x00(OK)  0x2d:0x00(OK)  0x2e:0x00(OK)  0x2f:0x00(OK)
0x30:0x00(OK)  0x31:0x00(OK)  0x32:0x00(OK)  0x33:0x00(OK)
0x34:0x00(OK)  0x35:0x00(OK)  0x36:0x00(OK)  0x37:0x00(OK)
0x38:0x00(OK)  0x39:0x00(OK)  0x3a:0x00(OK)  0x3b:0x00(OK)
0x3c:0x00(OK)  0x3d:0x00(OK)  0x3e:0x00(OK)  0x3f:0x00(OK)
0x40:0x00(OK)  0x41:0x00(OK)  0x42:0x00(OK)  0x43:0x00(OK)
0x44:0x00(OK)  0x45:0x00(OK)  0x46:0x00(OK)  0x47:0x00(OK)
0x48:0x00(OK)  0x49:0x00(OK)  0x4a:0x00(OK)  0x4b:0x00(OK)
0x4c:0x00(OK)  0x4d:0x00(OK)  0x4e:0x00(OK)  0x4f:0x00(OK)
0x50:0x00(OK)  0x51:0x00(OK)  0x52:0x00(OK)  0x53:0x00(OK)
0x54:0x00(OK)  0x55:0x00(OK)  0x56:0x00(OK)  0x57:0x00(OK)
0x58:0x00(OK)  0x59:0x00(OK)  0x5a:0x00(OK)  0x5b:0x00(OK)
0x5c:0x00(OK)  0x5d:0x00(OK)  0x5e:0x00(OK)  0x5f:0x00(OK)
0x60:0x00(OK)  0x61:0x00(OK)  0x62:0x00(OK)  0x63:0x00(OK)
0x64:0x00(OK)  0x65:0x00(OK)  0x66:0x00(OK)  0x67:0x00(OK)
0x68:0x00(OK)  0x69:0x00(OK)  0x6a:0x00(OK)  0x6b:0x00(OK)
0x6c:0x00(OK)  0x6d:0x00(OK)  0x6e:0x00(OK)  0x6f:0x00(OK)
0x70:0x00(OK)  0x71:0x00(OK)  0x72:0x00(OK)  0x73:0x00(OK)
0x74:0x00(OK)  0x75:0x00(OK)  0x76:0x00(OK)  0x77:0x00(OK)
0x78:0x00(OK)  0x79:0x00(OK)  0x7a:0x00(OK)  0x7b:0x00(OK)
0x7c:0x01(OK)  0x7d:0x02(OK)  0x7e:0x03(OK)  0x7f:0x04(OK)
EFUSE Test OK, Success case: 5

```

测试分析:

eFuse 0x00 ~ 0x1A 是 Secure 相关的地址，不可读取；除此以外的所有地址均可读取，由 Log 可以看出打印的结果符合预期。

第3章 使用注意事项

1. eFuse 是一种 otp (One-Time-Program) 存储器，其任意 bit 写为 1 后将会无法恢复为 0，这一点需要特别注意。
2. 如 1.1 小节介绍，正常情况下 eFuse 可写区域比较小，这是为了保护 eFuse 不被误写导致芯片工作异常；若确实需要写 eFuse 中的其他区域（如 SWD Debug Key 区域），则需额外配置以提升 eFuse 操作权限：

```
EFUSE->EF_FLASH_PERMISSION |= EFUSE_FLASH_PERMISSION_CTRL_Msk;
```

请注意，一般来说写 eFuse 中的 SWD Debug Key 区域应在用户产品的量产阶段，通过量产烧录工具 PANLINK 来操作，而不是直接在 C 代码中操作。

3. 任何情况下都请不要对 eFuse 0x1C~0x7B 地址中的任何地方进行写操作（读操作是允许的），这是用于 Panchip 存放芯片正常工作所需的基础配置信息的区域，不合理的修改将会导致芯片工作异常！