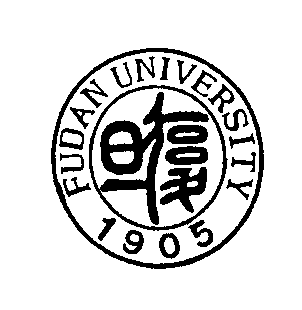
****

**Fudan Switch Simulator Tool**

**测试文档**

目录

[1. 引言 3](#_Toc45219586)

[1.1. 测试目的 3](#_Toc45219587)

[1.2. 参考书籍 3](#_Toc45219588)

[2. 测试环境 3](#_Toc45219589)

[3. 测试方法 3](#_Toc45219590)

[3.1. 单元测试 3](#_Toc45219591)

[3.2. 集成测试 3](#_Toc45219592)

[3.3. 白盒测试 3](#_Toc45219593)

[3.4. 黑盒测试 3](#_Toc45219594)

[4. 测试过程 3](#_Toc45219595)

[4.1. 端口接收数据帧 3](#_Toc45219596)

[4.1.1. 白盒测试 4](#_Toc45219597)

[4.1.2. 黑盒测试 7](#_Toc45219598)

[4.2. 交换机转发数据帧 10](#_Toc45219599)

[4.2.1. 白盒测试 10](#_Toc45219600)

[4.2.2. 黑盒测试 24](#_Toc45219601)

[4.3. MAC地址表清理表项 28](#_Toc45219602)

[4.3.1. 白盒测试 28](#_Toc45219603)

[4.3.2. 黑盒测试 34](#_Toc45219604)

# 引言

## 测试目的

* 通过对测试结果的分析，得到对软件质量的评价
* 分析测试的过程，产品，资源，信息，为以后制定测试计划提供参考
* 评估测试测试执行和测试计划是否符合
* 分析系统存在的缺陷，为修复和预防bug提供建议

## 参考书籍

《软件工程》 钱秋乐，清华大学出版社,2007

# 测试环境

测试环境是Windows10。

# 测试方法

## 单元测试

根据设计描述，对重要的控制路径进行测试，以发现构件或模块内部的错误。

## 集成测试

对单元测试后的模块集成的软件系统进行测试，揭露设计阶段产生的错误。

## 白盒测试

根据程序内部的逻辑结构及有关信息设计测试用例，检查程序中所有逻辑路径是否都按预定的要求正确地工作。

## 黑盒测试

不考虑程序内部的逻辑结构和内部特性，只依据程序的需求规格说明书，检查程序的功能是否符合它的功能需求。

# 测试过程

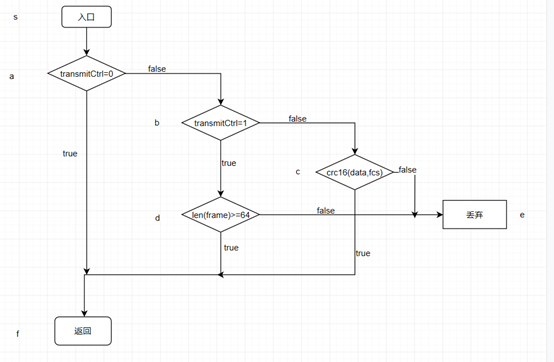
选取交换机实现的核心方法分别进行黑盒测试和白盒测试：

* 端口接收数据帧；
* 交换机转发数据帧；
* MAC地址表定期清理

## 端口接收数据帧

### 白盒测试

测试不同transmitCtri下不同数据帧在端口的接受与发送。分支图如下：



|  |  |  |  |
| --- | --- | --- | --- |
| **测试数据** | **预期效果** | **执行路径** | **覆盖条件**  **abcd** |
| 场景1 | 转发数据帧 | saf | T / / / |
| 场景2 | 转发数据帧 | sabdf | F T / T |
| 场景3 | 丢弃数据帧 | sabde | F T / F |
| 场景4 | 转发数据帧 | sabcf | F F T / |
| 场景5 | 丢弃数据帧 | sabce | F F F / |

1. 直通方式下数据帧转发（两机器分别与端口1，3相连）：

|  |  |  |  |
| --- | --- | --- | --- |
| **测试数据** | **预期效果** | **执行路径** | **覆盖条件** |
| transmitCtrl = 0；  数据帧：b'000000000000  000000001000  0800000oannsh' | 转发数据帧 | saf | tranmitCtrl=0 |

测试结果：



1. 碎片隔离方式下非残帧转发（两机器分别与端口1，3相连）：

|  |  |  |  |
| --- | --- | --- | --- |
| 测试数据 | 预期效果 | 执行路径 | 覆盖条件 |
| transmitCtrl = 1；  数据帧：  b'000000000000  000000001000 0800baca9ea2  hello world!hello world!hello world!' | 转发数据帧 | sabdf | tranmitCtrl=1  len(frame)>=64 |

测试结果：



1. 碎片隔离方式下残帧转发（两机器分别与端口1，3相连）：

|  |  |  |  |
| --- | --- | --- | --- |
| **测试数据** | **预期效果** | **执行路径** | **覆盖条件** |
| transmitCtrl = 1；  数据帧：b'000000000000  000000001000  0800baca9ea2h ' | 丢弃数据帧 | sabde | tranmitCtrl=1  len(frame)<64 |

测试结果：



1. 存储转发方式下正确帧转发（两机器分别与端口1，3相连）：

|  |  |  |  |
| --- | --- | --- | --- |
| **测试数据** | **预期效果** | **执行路径** | **覆盖条件** |
| transmitCtrl = 2；  数据帧：  b'000000000000  000000001000  0800baca9ea2  hello world!hello world!hello world!' | 转发数据帧 | sabcf | tranmitCtrl=2  crc16(data,fcs)= true |

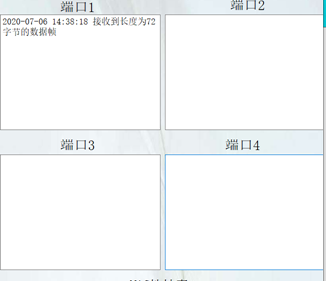
测试结果：



1. 存储转发方式下错误帧转发（两机器分别与端口1，3相连）：

|  |  |  |  |
| --- | --- | --- | --- |
| **测试数据** | **预期效果** | **执行路径** | **覆盖条件** |
| transmitCtrl = 2；  数据帧：  b'000000000000 000000001000 0800000oanns  hello world!hello world!hello world!' | 丢弃数据帧 | sabce | tranmitCtrl=2  crc16(data,fcs) =false |

测试结果：



### 黑盒测试

测试不同Mac地址。等价类划分如下：

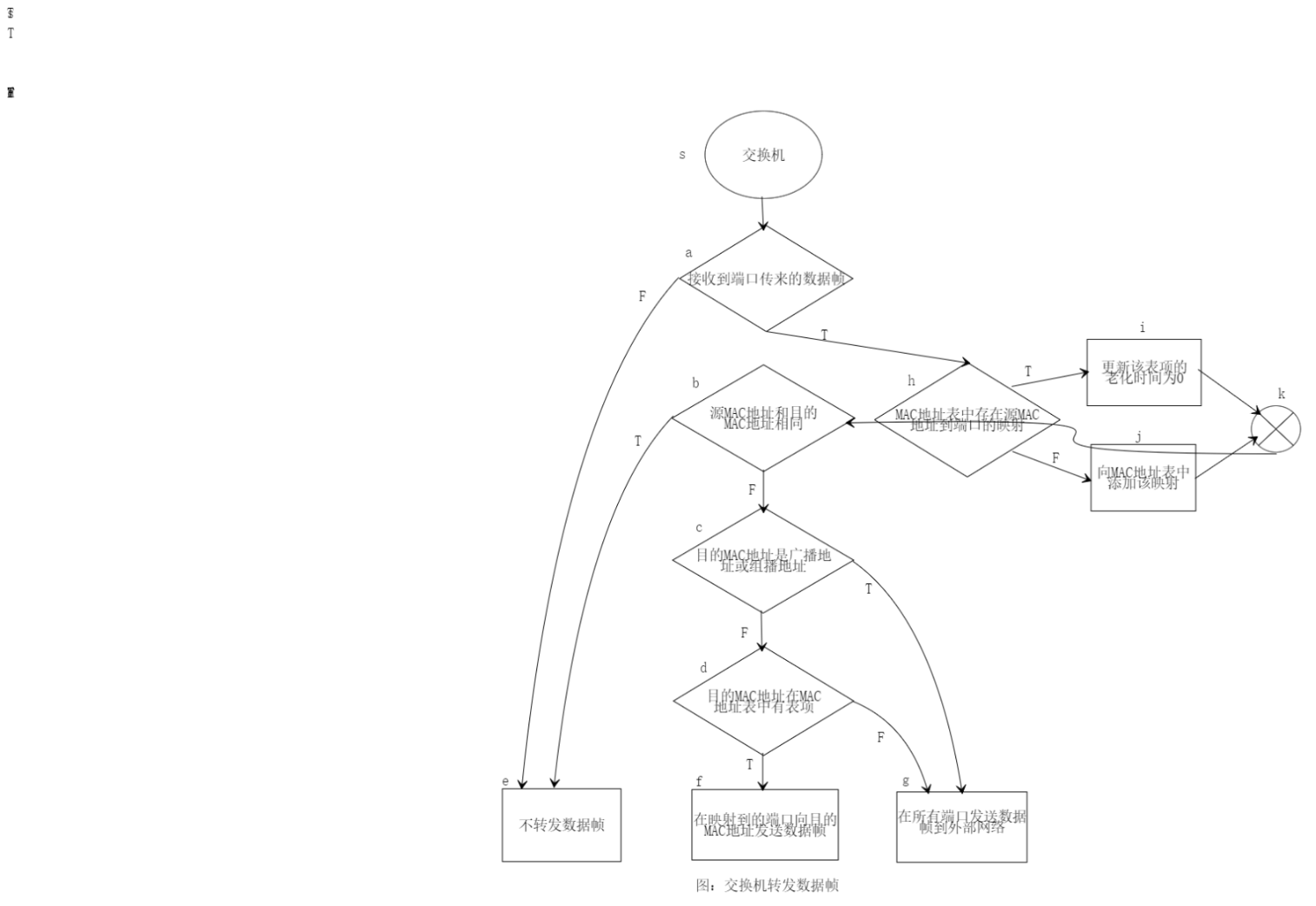
|  |  |  |
| --- | --- | --- |
| 输入条件 | 有效等价类 | 无效等价类 |
| Mac地址：第1、2、4、5、7、8、10、11、13、14、16、17个字符 | 数字（1）字母a~f（2）字母A~F（3） | 非字母数字的字符（4）不等于a~f或A~F的字母（5） |
| Mac地址：第3，6，9，12，15个字符 | ‘ ：’（6） | 数字（7）字母（8）‘ ：’以外的非字母数字的字符（9） |
| 字符数 | 17个（10） | >17或<17个（11） |

测试用例：

|  |  |  |
| --- | --- | --- |
| 输入数据 | 预期效果 | 覆盖的等价类 |
| VAR 00:00:00:00:00:00 | 正确标识符 | （1）（2）（3）（6）（10） |
| VAR 00:00:00:00:0g:00 | 报错：mac地址格式错误  正确格式：xx:xx:xx:xx:xx:xx  上式x为16进制数 | （5） |
| VAR 00:00:00:00:0+:00 | 报错：mac地址格式错误  正确格式：xx:xx:xx:xx:xx:xx  上式x为16进制数 | （4） |
| VAR 00000:00:00:00:00 | 报错：mac地址格式错误  正确格式：xx:xx:xx:xx:xx:xx  上式x为16进制数 | （7） |
| VAR 00a00:00:00:00:00 | 报错：mac地址格式错误  正确格式：xx:xx:xx:xx:xx:xx  上式x为16进制数 | （8） |
| VAR 00+00:00:00:00:00 | 报错：mac地址格式错误  正确格式：xx:xx:xx:xx:xx:xx  上式x为16进制数 | （9） |
| VAR 00:00:00:00:00:0 | 报错：mac地址格式错误  正确格式：xx:xx:xx:xx:xx:xx  上式x为16进制数 | （11） |

## 交换机转发数据帧

### 白盒测试

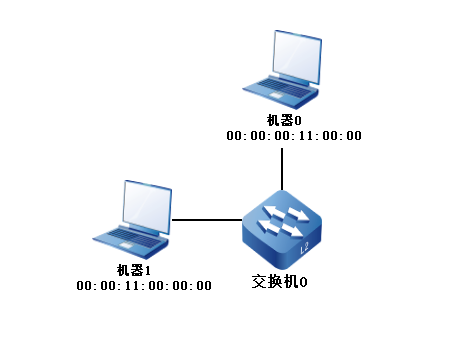


|  |  |  |  |
| --- | --- | --- | --- |
| **测试数据** | **预期结果** | **执行路径** | **覆盖条件** |
|  |  |  | a h b c d |
|  | 不转发数据帧 | sae | F / / / / |
| 场景1，7 | 更新表项的老化时间为0 | sahik | T T / / / |
| 场景0，6 | 向MAC地址表中添加表项 | sahjk | T F / / / |
| 场景2 | 不转发数据帧 | sakbe | T / T / / |
| 场景3，8 | 广播数据帧 | sakbcg | T / F T / |
| 场景4，9 | 单播数据帧 | sakbcdf | T / F F T |
| 场景5，10 | 广播数据帧 | sakbcdg | T / F F F |

注：为了减少白盒测试的分支，我们将测试分解成对几个核心功能的测试。在这里，因为这里测试的主体是交换机，MAC地址的合法性不在对交换机的白盒测试中进行。我们假设通过解析数据帧得到的源MAC地址和目的MAC地址的合法性。并且，在这一部分的UI展示中只展示交换机端口，MAC地址表的部分，不涉及客户机。

分别对有机器连在两台交换机上的情形和机器连在一台交换机的情形进行测试，保证测试的可靠性和全面性，不重不漏，覆盖到每一种情况。

**一台交换机：**

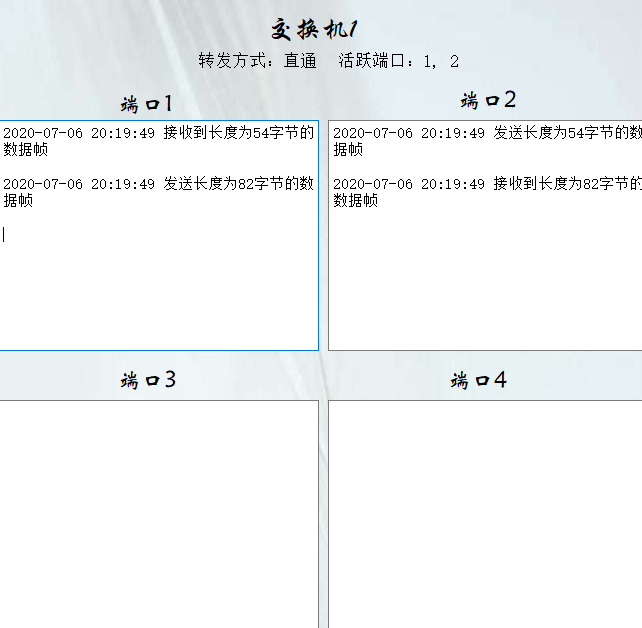


**场景0.**交换机接收到端口传递来的数据帧text1(机器0发送给机器1)，且（源MAC地址，端口号）不在MAC地址表中，则向MAC地址表中添加表项

初始时，MAC地址表为空，还未开始学习。



交换机端口：



MAC地址表：



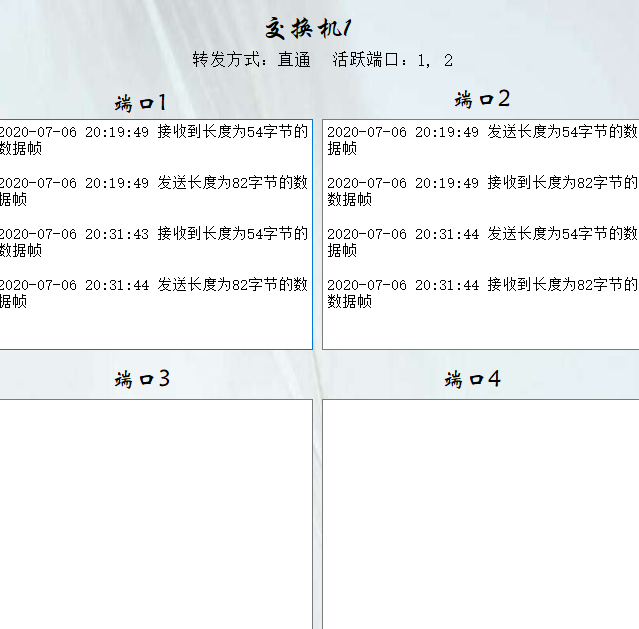
MAC地址表学习到了MAC地址到端口号的映射。

**场景1.**交换机接收到端口传递来的数据帧（机器0发送给机器1text1），且（源MAC地址，端口号）在MAC地址表中，则重置MAC地址表中该表项的老化时间

初始时，MAC地址表中已有表项



交换机端口：



MAC地址表：

****

机器0的表项的老化时间被更新。

**场景2.** 交换机接收到端口传递来的数据帧（机器0给机器0发送text1），源MAC地址与目的MAC地址相同，则不转发

交换机端口：

从端口看出，并没有转发数据帧。

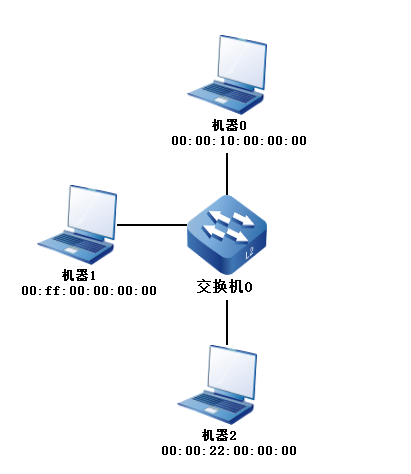


MAC地址表：

****

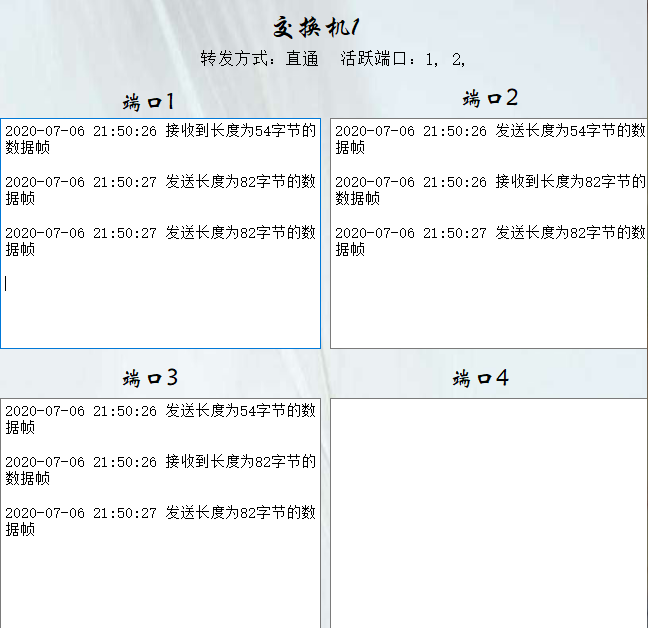
但是，MAC地址表学习到了源MAC地址到端口的映射。

**场景3.** 交换机接收到端口传递来的数据帧，且目的MAC地址是广播/组播地址，则广播数据帧（与该交换机除到来端口外所有端口相连的机器都会收到数据帧）



（机器0广播text1情况）

交换机端口情况：



所有端口都接收到了广播数据帧，转发给客户机，并且发送响应。

MAC地址表：



MAC地址表学习到了所有MAC地址到端口号的映射

（机器0组播text1情况）

交换机端口：

交换机端口情况和广播情况类似。



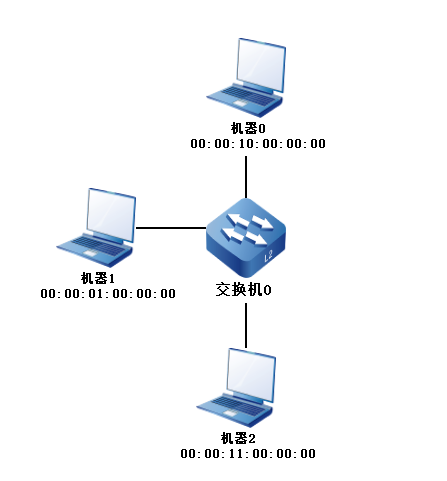
MAC地址表：

与广播情况类似



这里只列出了形如X1:XX:XX:XX:XX:XX的组播地址情况，其余情况也类似。

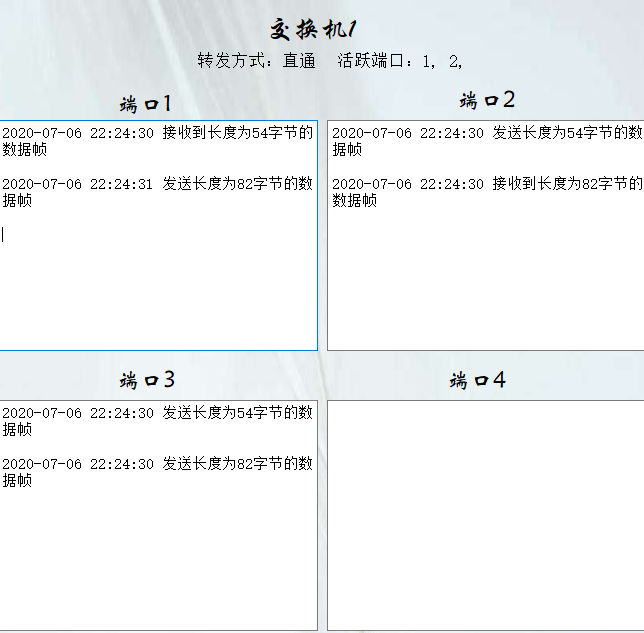
**场景4.** 交换机接收到端口传递来的数据帧（机器0给机器1发送tex1），目的MAC地址是单播地址，在MAC地址表中不存在目的MAC地址到端口号的映射

****

一开始，MAC地址表中不存在表项

****

交换机端口：

****

**MAC地址表：**

****

机器1会通过端口发送响应，从而让MAC地址表学习到映射关系；

**场景5.** 交换机接收到端口传递来的数据帧（机器0给机器1发送text），目的MAC地址是单播地址，在MAC地址表中存在目的MAC地址到端口号的映射

初始MAC地址表：

****

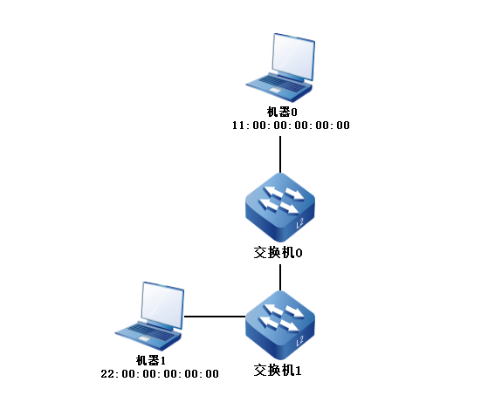
交换机端口：



MAC地址表：



**两台交换机：**



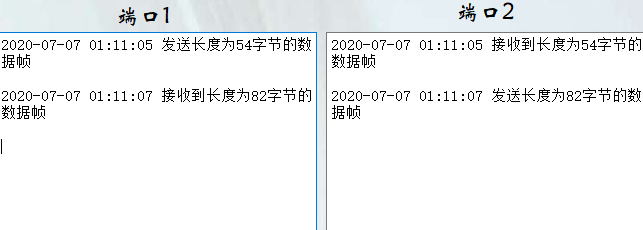
**场景6.**交换机接收到端口传递来的数据帧（机器0给机器1发送text1），且（源MAC地址，端口号）不在MAC地址表中，则向MAC地址表中添加表项

初始：

交换机1，0的MAC的地址表为空

交换机0：

(端口)

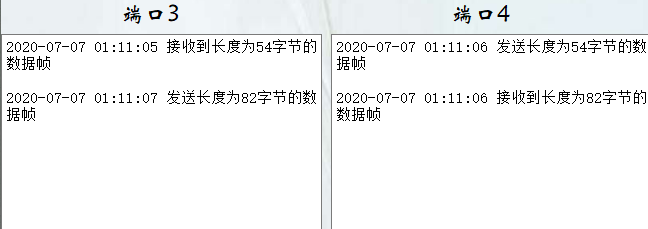


（MAC地址表）



交换机1：

（端口）



(MAC地址表)



**场景7.**交换机接收到端口传递来的数据帧（机器0传text1给机器1），且（源MAC地址，端口号）在MAC地址表中，则重置MAC地址表中该表项的老化时间

初始MAC地址表

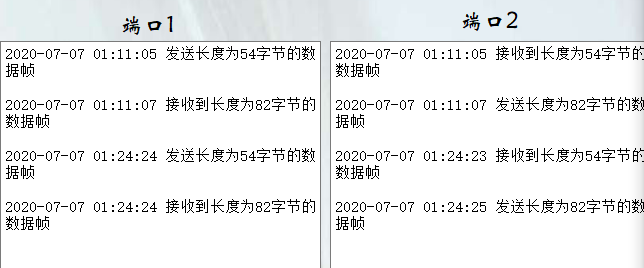
（交换机0）



（交换机1）

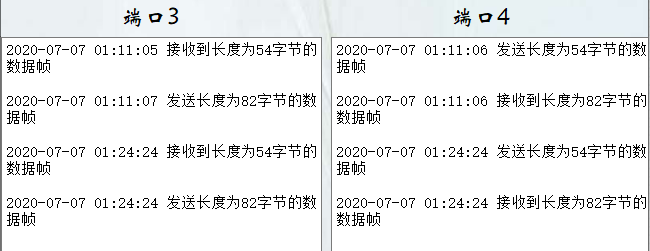


交换机1端口，MAC地址表





交换机0端口，MAC地址表



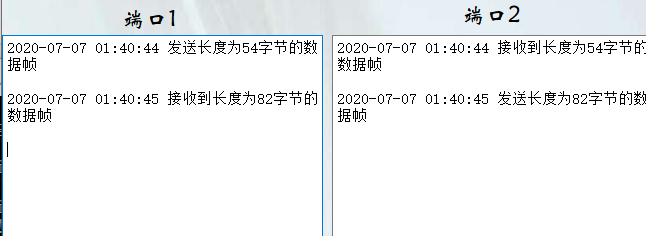


两台交换机的相应端口接收到数据帧，并且更新相应表项的老化时间为0。

**场景8.** 交换机接收到端口传递来的数据帧，且目的MAC地址是广播/组播地址，则广播数据帧（与该交换机除到来端口外所有端口相连的机器都会收到数据帧）

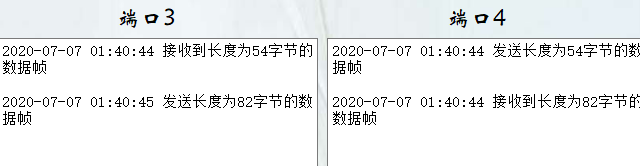
机器0广播数据帧

交换机0





交换机1





**两台交换机的MAC地址表都重置了表项的老化时间。**

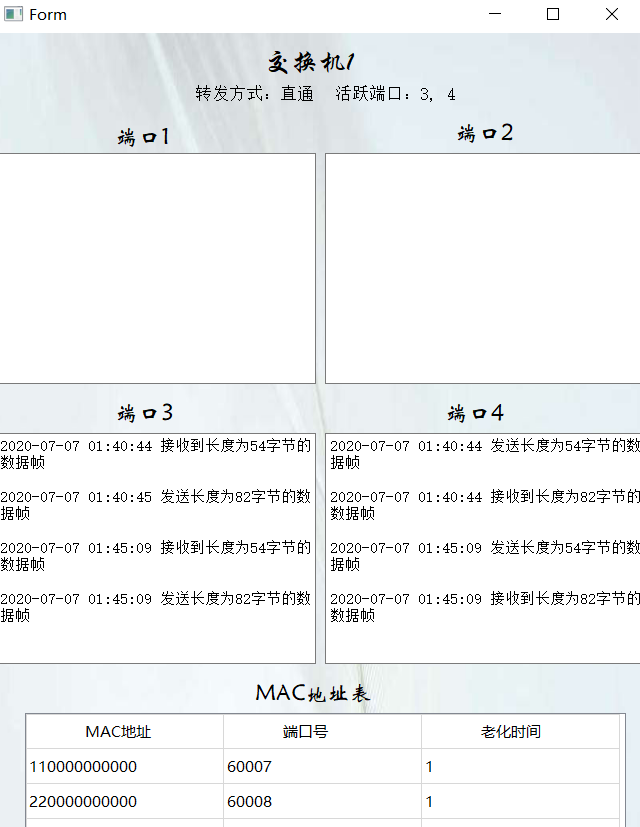
组播情况类似广播情况：

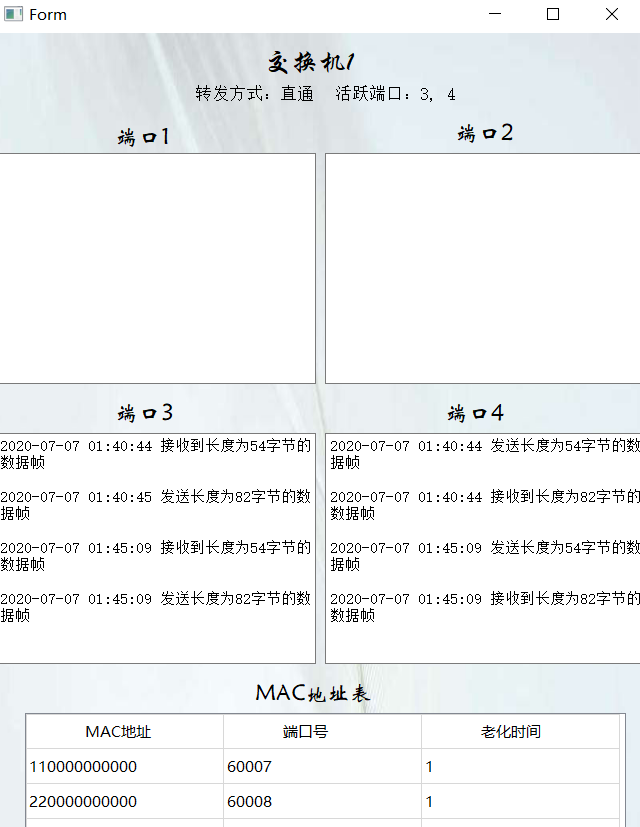
交换机0端口&MAC地址表

****

****

交换机1端口&MAC地址表

****

****

**场景9.** 交换机接收到端口传递来的数据帧（机器0给机器1发送text1

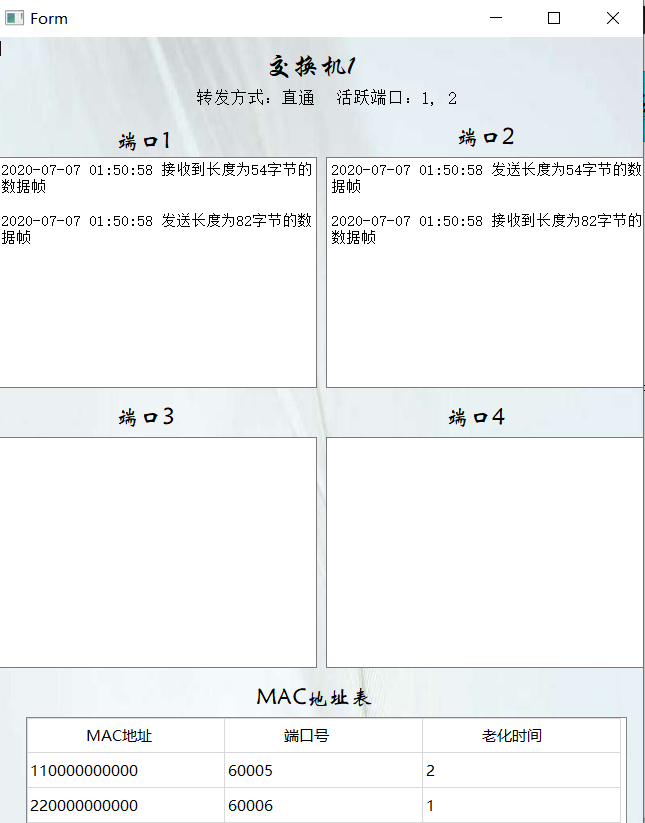
），目的MAC地址是单播地址，在MAC地址表中不存在目的MAC地址到端口号的映射

初始时，MAC地址表都为空

交换机0端口&MAC地址表



交换机1端口&MAC地址表



MAC地址表会学习到源MAC地址到端口的映射和目的MAC地址到端口的映射。

**场景10.** 交换机接收到端口传递来的数据帧（机器0给机器1发送text1

），目的MAC地址是单播地址，在MAC地址表中存在目的MAC地址到端口号的映射

交换机0端口&MAC地址表



交换机1端口&MAC地址表



与场景9的区别是，这里会更新对应表项的老化时间。

### 黑盒测试

等价类划分如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入条件 | 编号 | 有效等价类 | 编号 | 无效等价类 | 备注 |
| MAC地址 | (1) | 按字节以冒号隔开的十六进制数 | (2) | 违反规定格式的输入 | MAC地址标识了转发的目标机器 |
| 传送文件 | (3) | 可读文件 | - | - | 端口模块将文件内容作为数据打包成数据帧 |

数据用例：

|  |  |  |
| --- | --- | --- |
| 输入数据 | 预期结果 | 覆盖的等价类 |
| MAC地址为00:00:00:00:00:00 | 正确 | (1) |
| MAC地址为KK:00:00:00:00:00 | 提示MAC地址格式错误 | (2) |
| 传送文本文件data.txt（1000字节）和long\_data.txt（1500字节） | 正确 | (3) |

测试用例：

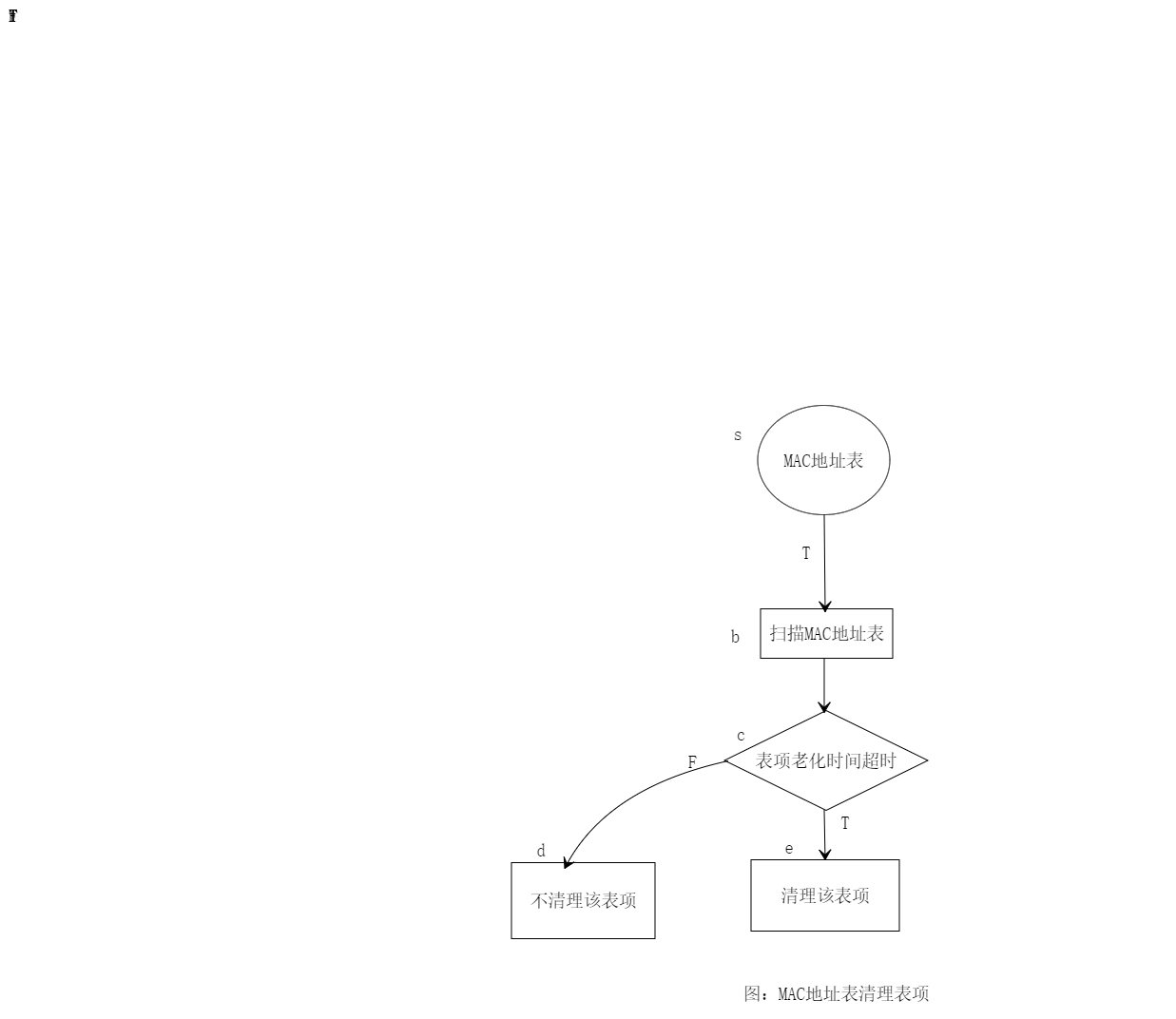
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 用例描述 | 前置条件 | 输入 | 预期结果 |
| (2.1) | 客户机向另一台客户机发送消息 | 已经添加了两台客户机，而且它们在交换机网络中存在通路 | 发送端为客户机为0，接收端为客户机1，发送的文件为data.txt | 客户机1接收到数据帧 |
| (2.2) | 客户机向自己发送消息 | 存在一台客户机 | 发送端和接收端都是客户机0，发送的文件为data.txt | 不转发数据帧 |
| (2.3) | 客户机向不存在的客户机发送消息 | 存在若干台客户机，而且它们互相连通 | 发送端为客户机0，目的地址为网络中不存在的MAC地址，发送的文件为data.txt | 除了发送端的客户机都接收到数据帧 |
| (2.4) | 客户机向另一台客户机发送空消息 | 存在两台客户机，而且它们互相连通 | 发送端为客户机为0，接收端为客户机1，发送的文件为空文件 | 交换机丢弃残帧 |
| （2.5） | 客户机向另一台客户机发送长消息 | 存在两台客户机，而且它们互相连通 | 发送端为客户机为0，接收端为客户机1，发送的文件为long\_data.txt | 客户机1接收到数据帧 |
| (2.6) | 客户机广播消息 | 存在若干台客户机，而且它们互相连通 | 发送端为客户机0，接收端为其余2台客户机，发送的文件为data.txt | 客户机1和客户机2都接收到数据帧 |
| (2.7) | 客户机组播消息 | 存在若干台客户机，而且它们互相连通 | 发送端为客户机0，接收端为其余2台客户机，发送的文件为data.txt | 客户机1和客户机2都接收到数据帧 |

测试结果：

|  |  |  |
| --- | --- | --- |
| 编号 | 测试结果截图 | 结论 |
| (2.1) |  | 测试通过 |
| (2.2) |  | 测试通过 |
| (2.3) |  | 测试未通过，客户机1未收到广播的数据帧 |
| (2.4) |  | 测试未通过，残帧未被交换机丢弃 |
| (2.5) |  | 测试未通过，客户机1未收到长数据帧的所有分段 |
| (2.6) |  | 测试未通过，客户机1,2未收到广播的数据帧 |
| (2.7) |  | 测试通过 |

## MAC地址表清理表项

### 白盒测试

****

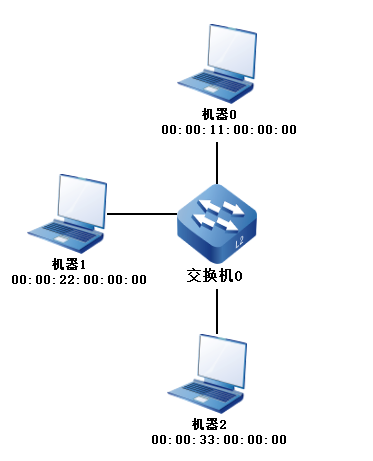
|  |  |  |  |
| --- | --- | --- | --- |
| **测试数据** | **预期结果** | **执行路径** | **覆盖条件** |
|  |  |  | c |
| 场景0，2 | 不清理该表项 | sabcd | F |
| 场景1，3 | 清理该表项 | sabce | T |

其中，场景0，1对应单台交换机的情况；场景2，3对应两台交换机的情况。

**场景0. 机器通过单台交换机连接，**表项老化时间没有超过阈值，则不清理该表项

设置的阈值为10，每查找一次MAC地址表，老化时间增加1；如果老化时间没有超过阈值，表项会继续保留在MAC地址表中。

机器0先给机器1发送tex1，然后再连续给机器2发送9次tex1，机器1到对应端口的映射一直保存在MAC地址表中，老化时间递增至10.





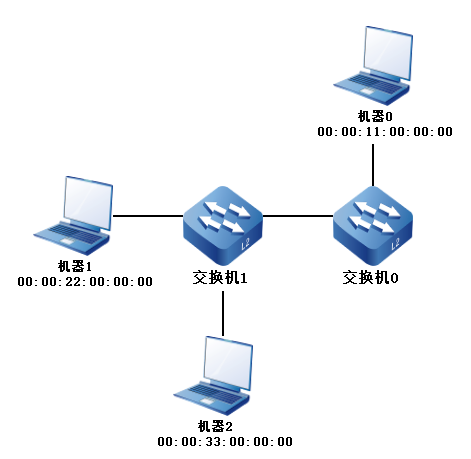


**场景1.** 表项老化时间超过阈值，则清理该表项

机器0先给机器1发送tex1，然后再连续给机器2发送10次tex1，此时，机器1到对应端口的映射的表项的老化时间变为11，超过阈值10，该表项被清理。



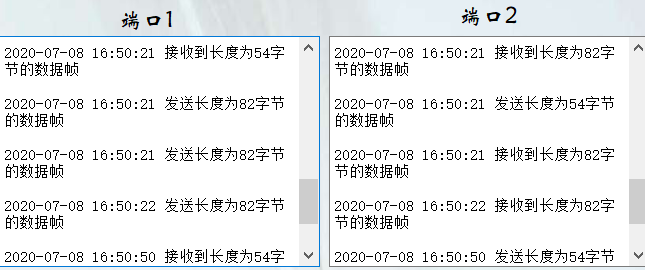
**场景2.** 机器通过多台交换机连接，对多台交换机都满足：表项老化时间没有超过阈值，则不清理该表项



设置的阈值为10，每查找一次MAC地址表，老化时间增加1；如果老化时间没有超过阈值，表项会继续保留在MAC地址表中。

机器0先给机器1发送tex1，然后再连续给机器2发送9次tex1，机器1到对应端口的映射一直保存在交换机0和交换机1的MAC地址表中，老化时间递增至10.

交换机0：





交换机1：





**场景3.** 机器通过多台交换机连接，对多台交换机都满足：表项老化时间超过阈值，则清理该表项

机器0先给机器1发送tex1，然后再连续给机器2发送10次tex1，此时，机器1到对应端口的映射的表项的老化时间变为11，超过阈值10，该表项在交换机0的MAC地址表和交换机1的MAC地址表中都被清理。

交换机0：



交换机1：



### 黑盒测试

等价类划分：

|  |  |  |
| --- | --- | --- |
| 输入条件 | 有效等价类 | 无效等价类 |
| MAC地址表的老化时间 | 0~10（1） | >11（2） |

测试用例格式：

|  |  |  |
| --- | --- | --- |
| 输入数据 | 预期结果 | 覆盖的等价类 |
| 0，1，2，3，4，5，6，7，8，9，10 | 表项保留在MAC地址中 | （1） |
| 11 | 表项被清理 | （2） |