ISO/IEC 14443A 协议防碰撞算法

1 概述

RFID 防碰撞协议/算法用于 reader 从当前射频场中存在的多个 transponders 中选择一个进行通信,且不受其他 transponders 的影响;

防碰撞协议/算法由 reader 和 transponders 之间多次通过命令和响应交互实现,这是一个连续的过程,不能延迟和暂时中断(中断意味着防碰撞算法失败),因此配套试验箱没有(也难以)提供防碰撞算法的实验;

为帮助学习和掌握防碰撞算法,开发了pcdsim和piccsim程序,分别仿真ISO/IEC 14443A协议的 PCD 和 PICC,可演示 ISO/IEC14443A协议防碰撞算法执行过程,以帮助和学习和掌握ISO/IEC14443A防碰撞算法;

2 使用说明

2.1 环境

- 1. OS: Windows 7 later;
- 2. 程序设计语言: C++; IDE: VS2010;
- 3. PCDSim 和 PICCSim 可在同一台计算机上运行,也可运行于网络中的两台计算机上; PCDSim 和 PICCSim 之间使用 udp 进行通信;
 - a) 在运行之前首先需要配置两个程序的 udp 侦听端口(也可使用默认);
 - b) PCDSim 采用组播方式向网络中发送报文, 仿真 PCD 的运行方式: 向空口广播报文;
 - c) PICCSim 采用 udp 方式向 PCDSim 发送数据, 仿真 PICC 的运行方式; RFID 中 PICC 采用 负载调制方式向 PCD"发送"数据,其他 PICC 收不到该数据,相当于单向传送;因此 PICCSim 需要配置 PCD 所在的机器的 ip 地址;

2.2 PCDSim

PCDSim 用于仿真 PCD 的防碰撞过程。

运行开始之前:

- 1. 配置 PCD rx udp port 和 PICC rx udp port;
- 2. 防碰撞过程中,新增加比特的值: 0 or 1;

程序可以单步运行或者连续运行,连续运行和 PCD 的工作模式相同,单步运行可以观察到每个命令发送和响应过程。当然连续运行后,通过界面也可以观察到防碰撞全过程;

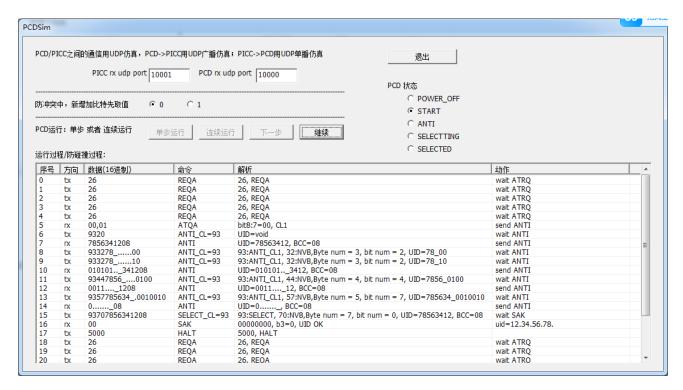


图 1 PCDSim

2.3 PICCSim

PICCSim 用于仿真 PICC 的防碰撞过程。

PIC	C rx po	rt 10001 PCD Ip addr					
与以十	六进制格	8式输入4/7/10字节的PICC 1	D: 12345678	PICC状态 C POWER_OFF			
żπ[I	:#41. ±	テDICC进入DCD射橋は、押					
ス いしナ	下[开始],表示PICC进入PCD射频场,耦合电源成功,开始工作:						
計下[4	宋下[停止],表示PICC案开PCD射频场,停止工作;						
X I LI	-11.3	010 4401 4100-00 17.		停止 C ACTIVE			
値过	程/防碰	撞过程		• HALI			
序号	方向	数据(16进制)	命令	解折	动作		
0	rx	26	REQA	26, REQA	Rx REQA. Tx ATQA.		
1	tx	00,01	ATQA	bit8:7=00, CL1			
2	rx	9320	ANTI_CL=93	UID=void	Rx ANTI, Send ANTI.		
3	tx	7856341208	ANTI	UID=78563412, BCC=08			
4	ΓX	93327800	ANTI_CL=93	93:ANTI_CL1, 32:NVB,Byte num = 3, bit num = 2, UID=78_00	Rx ANTI, not my UID. do nothing.		
5	rx	93327810	ANTI_CL=93	93:ANTI_CL1, 32:NVB,Byte num = 3, bit num = 2, UID=78_10	Rx ANTI, Send ANTI.		
6	tx	010101341208	ANTI	UID=0101013412, BCC=08			
7	rx	934478560100	ANTI_CL=93	93:ANTI_CL1, 44:NVB,Byte num = 4, bit num = 4, UID=7856_0100	Rx ANTI, Send ANTI.		
8	tx	00111208	ANTI	UID=001112, BCC=08			
9	rx	93577856340010010	ANTI_CL=93	93:ANTI_CL1, 57:NVB,Byte num = 5, bit num = 7, UID=785634_0010010	Rx ANTI, Send ANTI.		
10	tx	008	ANTI	UID=0, BCC=08			
11	rx	9370785634120865c7	SELECT_CL=93	93:SELECT, 70:NVB,Byte num = 7, bit num = 0, UID=78563412, BCC=08	Rx SELECT, Tx SAK		
	tx	00	SAK	00000000, b3=0, UID OK			
12	ΓX	5000	HALT	5000, HALT	Rx HALT. Tx nothing.		
12 13				accept in the	Tot Till Lett Till Till Till Till Till Till Till Ti		

图 2 PICCSim

运行开始之前:

- 1. 配置 PCD ip 地址、rx udp port 和 PICC rx udp port;
- 2. 输入 PICC UID;

PCD/PICC之间的通信用UDP仿真,PCD->PICC用UDPC	`播仿真;PICC->PCD用UDP单播仿真
PICC rx port 10001 PCD Ip addr 127 . 0	. 0 . 1 PCD rx port 10000

请以十六进制格式输入4/7/10字节的PICC ID: 12345678

按下开始按钮后,程序连续运行,通过界面也可以观察到防碰撞全过程;按下停止按钮,相当于 PICC 离开的射频场;

3 防碰撞算法

以 PICC 的 UID 为 0x12345678, 说明防碰撞算法执行过程:

PCDSim	PICCSim
0-3, 发送 REAQ, 等待 ATQA;	
4, 发送 REAQ,等待 ATQA;	0,收到 REQA
	1,发送 ATQA
5, 收到 ATQA	
6,发送 ANTI, 0x93,20,无任何 UID 信息;	
	2, 收到 ANTI
	3, 发送 ANTI, 0x78563412, BCC;
7, 收到 ANTI, 0x78563412, BCC;	
8, 仿真发生了碰撞: 随机位置: 10	
发送 ANTI: 934278,00,取 bit9=0	
	4,收到,但不响应:0x56:0101,0110,最低两个
	比特为 10,
9, 发送 ANTI: 934278,10, 取 bit9=1	
	5. 收到,响应,发送: 10,1010,3412BCC
10, 收到 0101,013412BCC,	

15. 收到所有 uid,发送 SEL	
	11. 受到 SEL
	12. 发送 SAK
16, 收到 SAK	
17, 发送 HALT	
	13. 收到 HALT
18-发送 REQA	

注意:图 1 和 2 的界面显示中,如果显示一个字节的比特,是按从高到低顺序显示的,实际是按从低到高 传送的。

4 其他

PCDSim 和 PICCSim 运行时,分别在各自目录下生成了 pcdesp 和 piccdesp 文件,记录了防碰撞过程。

PCDSim 还生成了 pcddebug 文件,更详细记录了算法执行过程,特别注意接收到的比特序!.

如下是 pcddebug 文件的内容样例: 其中比特是按从 L 到高显示的.

Tx REQA. waiting ATQA. =====Begin =====

Tx REQA. waiting ATQA. =====Begin =====

Tx REQA. waiting ATQA.

Rx ATQA.

Tx SELECT (anticollision), waiting anticollision.

wait

Rx UID:0001.1110.0110.1010.0010.1100.0100.1000.

Rx anticollision.

Tx anticollision(new bit=0), waiting anticollision.

UID[0x93]: 0001.1110.00

```
Tx anticollision(new bit=1), waiting anticollision.
UID[0x93]: 0001.1110.01
wait[0x93]
Rx UID:10.1010.0010.1100.0100.1000.
Rx anticollision.
Tx anticollision(new bit=0), waiting anticollision.
UID[0x93]: 0001.1110.0110.1010.0010.
wait[0x93]
Rx UID:1100.0100.1000.
Rx anticollision.
Tx anticollision(new bit=0), waiting anticollision.
UID[0x93]: 0001.1110.0110.1010.0010.1100.0100.100
wait[0x93]
Rx UID:0.
Rx anticollision.
Tx SELECT, waiting SAK
UID[0x93]:0001.1110.0110.1010.0010.1100.0100.1000.
wait[0x93]
Rx SAK[0x00].
End. ==== Begin =====
```

Tx REQA. waiting ATQA. ==== Begin =====

wait[0x93].