[illegible]
21281280

北京交通大學

作业4

1. a) block 0 类型为 ~~write~~ read/write block
   block 1 和 2 类型为 value block.

   b) ~~100 = 0x64~~ ~~200 = 0xC8~~

   10000 = 0x00002710　　20000 = 00004E20

block 1:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
value				value				value				adr	adr	adr	adr
10	27	00	00	EF	D8	FF	FF	10	27	00	00	01	FE	01	FE

block 2:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
value				value				value				adr	adr	adr	adr
20	4E	00	00	DF	C1	FF	FF	20	4E	00	00	02	FD	02	FD

   c) byte 0 为 0x08；byte 1 设置为 0x01（题目未给存取控制位置）
   byte 2-3 为 0x04D2（假设题中的 id 为十进制）
   byte 4-7 为 byte 4-5：0x07E8，byte 6 为 0x05，byte 7 为 0x1A

综上：

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
08	01	D2	04	E8	07	05	1A	00	00	00	00	00	00	00	00

   d) block 0 可读可写，故 C1C2C3 = 100 或 011
   block 1 和 2 是 value block，可读/增/减，故 110
   但由于是收费系统，不可写. 可设为 001，充值时临时改为 110.

2. a) 门禁. 每条记录 16 bytes. 共用 4个 sectors 存12条记录.

4 bytes	2 bytes	1 byte	1 byte
时间	Reader ID	出/入	RFU

b) 钱包. 每条交易记录占 1个 block. 共用 4个 sectors 存 12条记录.

1-4 bytes	5-7 bytes	8-9 bytes	10 byte	11-12 bytes	13-16 bytes
交易时间	原金额	交易金额	交易类型	交易机器	RFU.

钱包信息占 1个 sector, 其中使 1个 block 记录信息:

1-3 bytes	4-6 bytes	7-8 bytes	9-12 bytes	13-14 bytes	15-16
原金额	余额	充值员 ID	充值时间	充值机编号	RFU

其余 2个 block 设置为 value block 用于存余额.

c) 图书 每本书占 1个 sector. 共 16×3=48 bytes.

前 8 bytes: 借出日期 4 bytes 归还日期 4 bytes

后 40 bytes: 书名/书 ID.

共占 5个 sector.

综上. 门禁 1-4 sectors
图书 10-14 sectors } C1C2C3=100. Read: KeyA 或 KeyB
Write: KeyB.
门禁有 KeyB. 用户有 KeyA.

钱包 5 sector 钱包. → C1C2C3=001, KeyA/KeyB 可读可消费.
C1C2C3= 110, 仅 KeyB 充值.

6-9 sectors 记录交易 → ~~C1C2C3=010. 只读.~~
~~充值/消费 KeyB 和用户 KeyA 均可读.~~
C1C2C3=100.
充值/消费 KeyB 可读可写. 用户 KeyA 只读.

剩下 sector 15, RFU.

Inventory: inventory

- request

字段名称	比特数
SOF	
Flags	8 bits
Inventory	8 bits
Optional AFI	8 bits
Mask length	8 bits
Mask value	0 – 64 bits
CRC16	16 bits
EOF	

- responses

字段名称	比特数
SOF	
Flags	8 bits
DSFID	8 bits
UID	64 bits
CRC16	16 bits
EOF	

- 函数

```
int inventory(
    uint8 u8Flag,
    uint8 u8AFI,
    uint8 u8MaskLen, // bitlen
    uint8 au8MaskValue[8], // .
    uint8 *pu8RxFlags,
    uint8 *pu8RxDSFID,
    uint8 au8RxUID[8]
);
```

返回：

- `1` 表示清晰收到一个 VICC 的应答；
- `0` 表示没有 VICC 应答；
- `<0` 表示有多个 VICC 应答；

如果返回 `< 0`，或者：VCD 发送 EOF，继续点名，或者：修改 masklen & maskvalue，重新发送 inventory()

Stay quiet:

- request

字段名称	比特数
SOF	
Flags	8 bits
Stay quiet	8 bits
UID	64 bits
CRC16	16 bits
EOF	

- response

  无

- 函数

```
int stayQuiet(
    uint8 u8Flags,
    uint8 au8UID[8]
);
```

Read Single Block

- request

字段名称	比特数
SOF	
Flags	8 bits
Read single block	8 bits
UID	64 bits
Block number	8 bits
CRC16	16 bits
EOF	

request 中，是否存在 UID，由 flags 决定（address mode or select mode）。

- response

字段名称	比特数
SOF	
Flags	8 bits
Error code	8 bits
CRC16	16 bits
EOF	

字段名称	比特数
SOF	
Flags	8 bits
Block security status	8 bits
Data	
Block length	
CRC16	16 bits
EOF	

- 函数

```
int readSingleBlk(
    uint8 u8Flags,
    uint8 au8UID[8],
    uint8 u8BlkNo,
    uint8 *pu8RxFlags,
    uint8 *pu8ErrCode
    uint8 *pu8BlkSecurityStatus,
    uint8 au8Dat[]
);
```

返回值：flags.
是否返回：pu8BlkSecurityStatus，要看 u8Flags 中 optional_flag。

或

```
// 返回：0，正确，否则 Flags.
Uint8 readSingleBlk(
    uint8 u8VisitMode, // 1 for address mode, 0 for select mode
    uint8 u8Optionalflag; // 1 return security status;
    uint8 au8UID[8],
    uint8 u8BlkNo,
    uint8 *pu8BlkSecurityStatus, // optionalflag = 1 时，有效；
    uint8 au8Dat[]
);
```

Write Single Block

- request

字段名称	比特数
SOF	
Flags	8 bits
Write single block	8 bits
UID	64 bits
Block number	8 bits
Data	
Block length	
CRC16	16 bits
EOF	

- response
  - 有 Error_flag

字段名称	比特数
SOF	
Flags	8 bits
Error code	8 bits
CRC16	16 bits
EOF	

  - 无 Error_flag

字段名称	比特数
SOF	
Flags	8 bits
Error code	8 bits
CRC16	16 bits
EOF	

- 函数

```
int writeSingleBlk(
    uint8 u8Flags,
    uint8 au8UID[8],
    uint8 u8BlkNo,
    uint8 au8Dat[],
    uint8 *pu8RxFlags,
    uint8 *pu8ErrCode
);
```

Read multiple Blocks

- request

字段名称	比特数
SOF	
Flags	8 bits
Read multiple block	8 bits
UID	64 bits
First block number	8 bits
Number of blocks	8 bits
CRC16	16 bits
EOF	

- response
  - 有 Error_flag

字段名称	比特数
SOF	
Flags	8 bits
Error code	8 bits
CRC16	16 bits
EOF	

  - 无 Error_flag

字段名称	比特数
SOF	
Flags	8 bits

字段名称	比特数
Block security status	8 bits
Data	
Block length	
CRC16	16 bits
EOF	

注: Data 和 Block length 的内容根据需要重复。

- 函数

```
int readMultipleBlks(
    uint8 u8Flags,
    uint8 au8UID[8],
    uint8 u8BlkFirstNo,
    uint8 u8BlkNums,
    uint8 *pu8RxFlags,
    uint8 *pu8ErrCode
    uint8 au8BlkSecurityStatus[],
    uint8 au8Dat[]
);
```

Write multiple Blocks

- request

字段名称	比特数
SOF	
Flags	8 bits
Write multiple block	8 bits
UID	64 bits
First block number	8 bits
Number of blocks	8 bits
Data	
Block length	
CRC16	16 bits
EOF	

注: Data 和 Block length 的内容根据需要重复。

- response

- 有 Error_flag

字段名称	比特数
SOF	
Flags	8 bits
Error code	8 bits
CRC16	16 bits
EOF	

- 无 Error_flag

字段名称	比特数
SOF	
Flags	8 bits
CRC16	16 bits
EOF	

- 函数

```
int writeMultipleBlks (
    uint8 u8Flags,
    uint8 au8UID[8],
    uint8 u8BlkFirstNo,
    uint8 u8BlkNums,
    uint8 au8Dat[],
    uint8 *pu8RxFlags,
    uint8 *pu8ErrCode
);
```