Fbus V2 protocol

Elie aka Lupin Bursztein
(elie@bursztein.net)

Rev 1.0

Table des matières

1 Abstract

This document focus on how a Nokia phone talk with a P.C. computer. It try to give a comphrensive explantion of how the exchange protocol is build.

1.1 Warnning

Because Nokia keep it protocol specification, This document do not rely on any official source. This document is based on my experience using a Nokia 5510 and some documentation find on the internet. So do not rely on this. During all this document the packets is in hex format.

1.2 Why being interesseted by this protocol ?

Well it depend of the usage you made from your phone. For exemple you may want to write an application that syncronize your phone address book with a web site or more simply you are curious.

1.3 How to figure out if the phone answer correctly ?

Use a serial com logger. That what i did.

1.4 During test is there a risk ?

Since i never broke a phone, i have make it freeze a couple of time, so i guess yes there is a risk. That's why i used a quite old phone.

2 Protocol principle

This protocol is a packet based protocol with a control transmission mecanism. Each packet as a sequence number.

2.1 Protocol exchange layout

Here is a sample of the protocol exchange :

1. P.C send a packet
2. Phone send a Ack packet
3. Phone Send the answer packet
4. P.C send a ack packet

Why this protocol has a Control Transmission mecanism ? Because you could have a collision du to the nature of the cable. For the same reason the Acknoledgement sequence number is here to know which packet have been received.

2.2 Protocol Speed

To properly configure the com port i has to be like this :

speed	115 200
num bits	8
parity	none
Stop bit	1

3 Standard message definition

A standard message look like this one :

```
1E 02 00 04 00 0B 01080002010463020401 40 00 3900
```

which corespond to :

```
[Frame type(1)][Src dev(1)][Dst dev(1)][CMD(1)][Frame type(1)][Len(1)][DATA(X)][Seq(1)][Padd(1 or 0)][Chksum(2)]
```

3.1 Frame type

The frame type indicate which type of protocol is using :

- 1E : Serial Fbus frame
- 1c : Irda Fbus frame

3.2 Src dev and Dst dev

Indicate the source and the destination device

- 02 Phone
- 00 Computer

3.3 CMD

This the command type, it define which type of information is about.

- need a fix

3.4 Frame type

Used if the message exceeded 255 then it give which part is sending. (need to be more detailled)

3.5 Len

The len of the packet. To calculate it : Data + Sequence number so in other word : len = data + 1 (in hex)

3.6 Data

the packets data.

3.7 Seq : Sequence number for regular packet

the sequence number for the standard frame seem's to be between 40 up to 47 So always initialize it to 40 at the beginning seem's to be working

3.8 Padd

Since the packet as to be a odd number, if the len is even it as to be added. The padd is always 00.

3.9 Chksum

The check sum is in fact two different checksum. The first hex represente the Xor of all the odd hex block from the packet, the second represent the Xor of all the even Hex block of the packet.

4 Ack message definition

A standard message look like this one :

```
1E 00 02 7F 00 02 02 01 1E 7C
```

which corespond to :

```
[Frame type(1)][Src dev(1)][Dst dev(1)][CMD(1)][Frame type(1)][Len(1)][DATA(X)][Seq(1)][Chksum(2)]
```

4.1 CMD

the command of this packet is always the 7F.

4.2 Len

The len is always 02 because of the Data (see below)

4.3 Data

Data is alway the CMD of the acknoledged packet.

4.4 Calculing the Sequence number for ack packet

The sequence number for ack packet seem's to be between 00 an 07. For the initialization it has to be synchronise with the seq of the packets issue from the phone. For Exemple

- packet received :1E02 00 04 00 0B [Data] 4400 3D 00 (seq is 44)
- ack packet to send : 1E00 02 7F 00 02 04**04**1879 (seq is 04)

5 Example of a communication

prt	Src	Dst	Cmd	Spc	Len	frm	data	Seq	Pad	cks	action
1E	00	02	04	00	06	00	01000101	40		1D42	
1E	00	02	02	00	09	00	01000D00000201	41	00	5F06	
1E	00	02	02	00	07	1E	0100200201	42	00	5C25	
1E	00	02	02	00	09	00	01000D01000201	43	00	5C06	
1E	00	02	64	00	06	00	01001001	44		1D37	
1E	00	02	D0	00	03	01	01	46	00	5BD2	
1E	00	02	40	00	06	00	01 64 01 01	47		7901	
1E	00	02	40	00	06	00	01 7E F3 01	40		63F4	
1E	00	02	40	00	06	00	01 7E 01 01	41		6307	netmonitor test 1
1E	00	02	40	00	06	00	01 7E 01 01	42		6304	netmonitor test 1
1E	00	02	40	00	06	00	01 7E 02 01	43		6306	netmonitor test 2 ?
1E	00	02	40	00	06	00	01 7E 03 01	44		6300	netmonitor test3

6 conclusion

I hope this short paper will be use full to any devellopper or personn that are curious about gsm phone. A huge Greating to the gnokii (www.gnokii.org) project members which do an impressive work