Appunti sul protocollo usato da Vimar by-me

Frame

Le informazioni che seguono non hanno nessuna pretesa di esattezza e di corrispondenza. In parte sono tratte da documentazione su konnex, in parte desunte dall'analisi dei frames che circolano sul bus Vimar by-me.

Vengono riportate ed esemplificati unicamente telegrammi di controllo, 9 bytes, lunghezza dati 1 byte: sono quelli che vengono usati per trasmettere semplici comandi tipo accendi/spegni/ecc… sul bus Vimar circolano anche telegrammi più lunghi e complessi, destinati alla configurazione dell'impianto e alla diagnostica.

TOP	SOURCE	DESTIN	CTR	PDU	DATA	CKS

TOP 1 BYTE - info generali del frame - normalmente 0xB4
- Bit 7-6 formato (10)
- Bit 5 ripetizione (0)
- Bit 4 fisso 1
- Bit 3-2 priorità (00=sistema, 01=allarmi, 10=normale, 11=bassa)
- Bit 1-0 fisso 00

SOURCE 2 BYTES indirizzo di provenienza – il byte 1 solitamente vale sempre 0x10
- BYTE 1
  - Bits 7-6-5-4 linea
  - Bits 3-2-1-0 settore
- BYTE 2
  - Bits 7-0 device

DESTIN 2 BYTES indirizzo di destinazione – il byte 1 solitamente vale sempre 0x0B
- BYTE 1 - per gli indirizzi di scenario vale 0x0F
  - Bits 7-6-5-4 linea
  - Bits 3-2-1-0 settore
- BYTE 2
  - Bits 7-0 device

CTR counter – informazioni varie e lunghezza dati
- Bit 7 indirizzo di gruppo (1)
- Bit 6-5-4 routing
- Bit 3-2-1-0 lunghezza dati (da 1 a 8)

PDU

DATA comandi e informazioni (il numero di bytes è indicato in lunghezza dati)

CKS checksum (xor di 0xFF e di tutti i bytes precedenti)

I frame sono trasmessi a 9600baud, ogni byte composto da 1 bit di start, 8 bits di dati, 0 bit di stop. Ogni bit dura 104uS, durata di 1 frame standard circa 12,2mS.

La collisione deve essere intercettata da chi trasmette e corrisponde alla situazione in cui si sta trasmettendo ed il bus assume uno stato non corrispondente a quello atteso in un qualunque istante. In tal caso il frame viene ripetuto dopo un periodo di attesa di bus libero.

Chi riceve il telegramma (il destinatario) conferma la ricezione con un byte di acknowledgement 0xCC. Se non riceve ack il mittente ripete l'invio per altre due volte.

Dal nostro punto di vista (Vimar by-me) per mandare un comando: il byte di TOP vale sempre 0xB4, i bytes di indirizzo SOURCE (chi manda il comando) sono irrilevanti, nell'indirizzo DESTIN il primo byte sarà sempre uguale (probabilmente 0x0B, il secondo byte è importante: è l'indirizzo del dispositivo che comandiamo. Il byte CTR sarà sempre 0xE1 perché la lunghezza dati di comando è sempre 1. Il byte PDU nei comandi vale sempre zero. Segue il byte di comando e il check byte (risultato dagli xor dei bytes precedenti).

Gli indirizzi dei dispositivi attuatori luce sono sempre dispari.

luci
- 0x80 spegni
- 0x81 accendi

Gli attuatori delle tapparelle hanno 2 indirizzi: l'indirizzo di base (di solito dispari) risponde ai comandi effettuati con pressione breve del pulsante di comando: comando 0x80 oppure 0x81 (STOP). L'indirizzo base + 1 (di solito pari) risponde ai comandi effettuati con pressione lunga sul pulsante di comando.

Tapparelle (indirizzo base dispari – es 09)
- 0x80 stop
- 0x81 stop

Tapparelle (indirizzo base+1 – es 0A)
- 0x80 alza
- 0x81 abbassa

Più raramente:

Tapparelle (indirizzo base pari – es 09) – es 0A
- 0x80 stop
- 0x81 stop

Tapparelle (indirizzo base+1 – es 0A) – es 0B
- 0x80 alza
- 0x81 abbassa

Gli attuatori "dimmer" hanno un funzionamento simile: l'indirizzo di base risponde ai comandi effettuati con pressione breve del pulsante di comando: comando 0x80 (spegni) oppure 0x81 (accendi).

La pressione lunga per aumentare/diminuire genera messaggi all'indirizzo successivo (indirizzo base + 1). Il primo messaggio viene generato all'inizio della pressione lunga, il secondo messaggio viene generato al rilascio del pulsante.

I comandi da me loggati:

pressione breve (indirizzo base)

- 0x80 spegni
- 0x81 accendi

Pressione lunga in alto: (indirizzo base+1)

- 0x89 aumenta intensita
- 0x88 fine aumento (rilascio pulsante)

Pressione lunga in basso: (indirizzo base+1)

- 0x81 diminuisci intensita
- 0x80 fine diminuzione (rilascio pulsante)

Scenari:

nei comandi di scenario l'indirizzo di destinazione (linea e settore) è impostato a 0x0F, l'indirizzo di dispositivo è il numero di scenario. Inoltre il byte di lunghezza vale 0xE2 e di conseguenza ci sono 2 bytes di dati: il primo è il classico comando 0x80-0x81, il secondo ripete il numero di scenario.

Esempi di comandi loggati

Accendi luce

B4 10 29 0B 65 E1 00 81 7C

B4: prefisso (TOP)
10 29: indirizzo dispositivo mittente
0B 65: indirizzo dispositivo destinatario (0B linea/settore, 65 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
81: accendi
7C: check byte (FF xor B4 xor 10 xor 29 xor 0B xor 65 xor E1 xor 00 xor 81)

Spegni luce

B4 10 29 0B 65 E1 00 80 7D

B4: prefisso (TOP)
10 29: indirizzo dispositivo mittente
0B 65: indirizzo dispositivo destinatario (0B linea/settore, 65 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
80: spegni
7D: check byte

Accendi dimmer

B4 10 15 0B 41 E1 00 81 64

B4: prefisso (TOP)
10 15: indirizzo dispositivo mittente
0B 41: indirizzo dispositivo destinatario (0B linea/settore, 41 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
81: accendi
64: check byte

Aumenta luce dimmer

B4 10 15 0B 42 E1 00 89 6F Ad inizio pressione prolungata

B4: prefisso (TOP)
10 15: indirizzo dispositivo mittente
0B 42: indirizzo PARI dispositivo destinatario (0B linea/settore, 42 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
89: aumenta
6F: check byte

B4 10 15 0B 42 E1 00 88 6E A fine pressione prolungata

B4: prefisso (TOP)
10 15: indirizzo dispositivo mittente
0B 42: indirizzo PARI dispositivo destinatario (0B linea/settore, 42 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
88: ferma l'aumento
6E: check byte

Diminuisci luce dimmer

B4 10 15 0B 42 E1 00 81 67 Ad inizio pressione prolungata

B4: prefisso (TOP)
10 15: indirizzo dispositivo mittente
0B 42: indirizzo PARI dispositivo destinatario (0B linea/settore, 42 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
81: diminuisci
67: check byte

B4 10 15 0B 42 E1 00 80 66 A fine pressione prolungata

B4: prefisso (TOP)
10 15: indirizzo dispositivo mittente
0B 42: indirizzo PARI dispositivo destinatario (0B linea/settore, 42 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
80: ferma la diminuzione
66: check byte

Spegni dimmer

B4 10 15 0B 41 E1 00 80 65

B4: prefisso (TOP)
10 15: indirizzo dispositivo mittente
0B 41: indirizzo dispositivo destinatario (0B linea/settore, 41 dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
80: spegni
65: check byte

Alza tapparella

B4 10 2F 0B 0E E1 00 80 10

B4: prefisso (TOP)
10 2F: indirizzo dispositivo mittente
0B 0E: indirizzo PARI dispositivo destinatario (0B linea/settore, 0E dispositivo)
E1: ctr – lunghezza dati 1
00: TPU
80: spegni
10: check byte

Ferma tapparella (pulsante su)

B4 10 2F 0B 0D E1 00 81 12

Oppure con pulsante giu

B4 10 2F 0B 0D E1 00 80 13 1101

Abbassa tapparella

B4 10 2F 0B 0E E1 00 81 11 1110

Esempi di comandi di gruppo (di ambiente?) (di scenario?)

Attiva gruppo/scenario 1

B0 10 01 0F 04 E2 00 80 04 33

B0: prefisso (TOP)
10 01: indirizzo dispositivo mittente
0F 04: indirizzo dispositivo destinatario (0F=scenario, 04 numero scenario)
E2: ctr – lunghezza dati 2
00: TPU
80: attiva scenario di spegnimento (?)
04: numero scenario
7C:

Abbassa le tapparelle dell'ambiente 1

Alza le tapparelle dell'ambiente 1

Esempi di comandi globali

Spegni tutte le luci

Appunti di Simone

VIMAR usa l'APCI 1111 (escape) con valori estesi per leggere e scrivere blocchi di 4 byte nella memoria di un device.

```
Valori estesi (6 bit):
- xx010101: Lettura
- xx010110: Risposta
- xx010111: Programmazione / scrittura
```

Successivamente al byte con il valore esteso ci sono sempre 3 byte 0x01, 0xC9, 0x40, tuttavia non ho capito cosa indicano: in lettura cambiando i primi 2 il device non risponde nulla, con il terzo continua a rispondere come se lo ignorasse.
Dopo questi 3 byte c'è un byte che indica invece l'indirizzo da leggere o scrivere e successivamente a lui se è una scrittura vengono passati i 4 byte da scrivere.

```
Strutta del telegramma di lettura:
00: Control Field
01: Source Address High
02: Source Address Low
03: Destination Address High
04: Destination Address Low
05: Routing Field
06: Command Field High
07: Command Field Low + Extended APCI
08: Data (Unknown meaning)
09: Data (Unknown meaning)
10: Data (Unknown meaning)
11: Data (Address to read)
12: Checksum
```

```
Strutta del telegramma di scrittura:
00: Control Field
01: Source Address High
02: Source Address Low
03: Destination Address High
04: Destination Address Low
05: Routing Field
06: Command Field High
07: Command Field Low + Extended APCI
```

```
08: Data (Unknown meaning)
09: Data (Unknown meaning)
10: Data (Unknown meaning)
11: Data (Address to write)
12: Data (1st byte to write)
13: Data (2nd byte to write)
14: Data (3rd byte to write)
15: Data (4th byte to write)
16: Checksum
```

```
L'indirizzo da leggere o scrivere sembra essere una cosa del tipo:
0x01: Dati blocco funzionale 1
0x05: Dati blocco funzionale 2
0x09: Dati blocco funzionale 3
0x0D: Dati blocco funzionale 4
```

Inoltre ho notato, ad esempio, che se vado avanti ancora di 4 e leggo l'indirizzo 0x11 per un attuatore a 4 uscite 01851.2 il primo byte (se non ricordo male) è una bitmask che mi indica quali attuatori sono attivi.

Ci ho più o meno preso? Puoi integrare in qualche modo queste informazioni?

Sebbene questo sia vero per l'attuatore a 4 uscite, non sono invece riuscito a capire lo stato con i seguenti device:

- 20512: Modulo con 2 pulsanti basculanti (Stato reale del led)
- 20527: Attuatore per tapparelle
- 01855: Gestione carichi (So come leggere la potenza, ma non so se ci sono altri dati utili e il loro significato, ad esempio lo stato dei vari carichi gestiti e / o se è possibile ottenere la loro potenza attiva)

Sarebbe molto utile anche il protocollo usato dai moduli della climatizzazione e quello per la programmazione di device, in quanto darebbe la possibilità di creare un device "virtuale" che fa da tramite con i moduli Wi-Fi dei condizionatori.

Ho invece visto che esiste un APCI 1100 (MaskVersionRead) a cui corrisponde un APCI 1101 (MaskVersionResponse), ma anche qui VIMAR non rispetta lo standard KNX e ritorna molti più dati di quelli previsti, ho visto anche che in realtà questo è l'APCI usato quando si lancia la diagnositca dei dispostivi da centrale.

Vimar by-me – lettura degli stati

Similmente a KNX ogni dispositivo contiene più blocchi funzionali e di ciascun blocco possono essere lette e/o modificate le proprietà, attraverso messaggi di property-read e property-write. La documentazione Vimar si ferma però a livello "alto" così come si utilizza in ETS.

Frame di richiesta stato

Le informazioni che seguono non hanno nessuna pretesa di esattezza e di corrispondenza. In parte sono tratte da documentazione su konnex, in parte desunte dall'analisi dei frames che circolano sul bus Vimar by-me.

I telegrammi di **richiesta stato** hanno un formato di questo tipo:

TOP	SOURCE	DESTIN	CTR	PDU	ID_O	ID_P	NP	INDEX	CKS

TOP 1 BYTE – control field – 0xB0
SOURCE 2 BYTES indirizzo di provenienza
DESTIN 2 BYTES indirizzo di destinazione
CTR lunghezza dati – vale 0x05
PDU 2 bytes – vale 0x03D5
ID_O 1 byte : ID oggetto che si interroga
ID-P 1 byte : ID proprietà dell' oggetto che si interroga
NP 4 bit : numero di elementi richiesti
INDEX 12 bit : indirizzo elemento
CKS checksum (xor di 0xFF e di tutti i bytes precedenti)

I telegrammi di **risposta alla richiesta stato** hanno un formato di questo tipo:

TOP	SOURCE	DESTIN	CTR	PDU	ID_O	ID_P	NP	INDEX	DATA	CKS

TOP 1 BYTE – control field
SOURCE 2 BYTES indirizzo di provenienza
DESTIN 2 BYTES indirizzo di destinazione
CTR lunghezza dati
PDU 2 bytes – vale 0x03D6
ID_O 1 byte : ID oggetto che si interroga
ID-P 1 byte : ID proprietà dell' oggetto che si interroga
NP 4 bit : numero di elementi richiesti
INDEX 12 bit : indirizzo elemento
DATA VALORE LETTO

CKS checksum (xor di 0xFF e di tutti i bytes precedenti)

Le informazioni O_ID, P_ID, N sono tipiche di ciascun tipo di dispositivo, riporto solo quelle relative ad alcuni dispositivi da me conosciuti:

Attuatore 14535 (ed altri) con uscita ad 1 relè:

- O_ID: 00
- P_ID: CA
- N: 1
- INDEX: 0 01
- VALORE LETTO (1 byte): 0=spento 1=acceso

Attuatore 01851 (ed altri) con uscita a 4 relè:

- O_ID: 00
- P_ID: CA
- N: 4
- INDEX: 0 01
- VALORI LETTO (4 bytes): 0=spento 1=acceso (un byte per ogni rele)

Attuatore per 2 tapparelle 01852 (ed altri) :

- O_ID: 00
- P_ID: CA
- N: 2
- INDEX: 0 05
- VALORI LETTI (2 byte): 0=ultimo comando SU 1=ultimo comando GIU (ogni byte una tapparella)

Comando / attuatore tapparelle 14527 (ed altri) :

- O_ID: 00
- P_ID: CA
- N: 1
- INDEX: 0 0B
- VALORE LETTO (1 byte): 0=ultimo comando SU 1=ultimo comando GIU

Modulo controllo carichi 01855 :

- O_ID: 01
- P_ID: C9
- N: 4
- INDEX: 0 01
- VALORE LETTO (4 byte) relativi alle prese 1-4: i bit da 0 a 6 indicano il tipo di controllo, il bit 7 indica se la presa è attivata o disattivata

  Le prese 5-8 si interrogano impostando in INDEX il valore 0 05.
  Lo stato degli assorbimenti correnti impostano in INDEX il valore 0 09.

Esempi – query:

B0 10 02 10 03 05 03 D5 00 CA 10 0B 4C

Risposta:

B0 10 03 10 02 66 03 D6 00 CA 10 0B 01 2D

Termostati

Le informazioni che seguono non hanno nessuna pretesa di esattezza e di corrispondenza. Sono desunte dall'analisi dei frames che circolano sul bus Vimar by-me. Non è chiaro se il termostato abbia un indirizzo proprio o se faccia riferimento ad un indirizzo globale relativo alla termoregolazione.

I telegrammi di **richiesta stato** hanno un formato di questo tipo:

TOP	SOURCE	DESTIN	CTR	PDU	ID_O	CKS

TOP 1 BYTE – control field – 0xB0
SOURCE 2 BYTES indirizzo di provenienza
DESTIN 2 BYTES indirizzo di destinazione
PDU 2 bytes – vale 0xE100
ID_O 1 byte : ID oggetto che si interroga – vale 00
CKS checksum (xor di 0xFF e di tutti i bytes precedenti)

I telegrammi di **risposta alla richiesta stato** hanno un formato di questo tipo:

TOP	SOURCE	DESTIN	CTR	PDU	ID_O	DATA	CKS

TOP 1 BYTE – control field
SOURCE 2 BYTES indirizzo di provenienza
DESTIN 2 BYTES indirizzo di destinazione
PDU 2 bytes – vale 0xE500
ID_O 1 byte : ID oggetto – 0x80
DATA 4 bytes:

stato	1 byte: modo di funzionamento
temperatura impostata	1 byte
temperatura corrente	2 bytes: il bit 15 va tolto e spostato sul bit 8 – dal valore convertito in decimale va sottratto 100 ed il valore risultante va considerato in decimi di grado centigrado.

CKS checksum (xor di 0xFF e di tutti i bytes precedenti)

Alcuni termostati rispondono anche alle query standard di interrogazione con:

- O_ID: 01
- P_ID: C9
- N: 5
- INDEX: 0 001
- VALORE LETTO (5 bytes): …

O anche:

- O_ID: 01
- P_ID: CB
- N: 1-5
- INDEX: 0 001 – 0 019 (esadecimale)
- VALORE LETTO (5 bytes): …