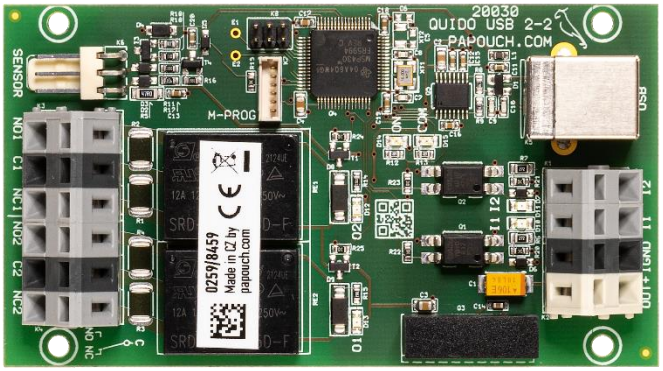
Quido Spinel

Kompletní popis komunikačního protokolu I/O modulů Quido



Quido Spinel

Dokumentace

Vytvořen: 23.11.2005

Poslední aktualizace: 18.10.2023 11:12

Počet stran: 48

This document has been designed using resources from Flaticon.com

---

Papouch s.r.o.

Adresa:
Strašnická 3164/1a
102 00 Praha 10

Telefon:
+420 267 314 268

Web:
papouch.com

Mail:
papouch@papouch.com

OBSAH

Přehled změn v tomto dokumentu

Verze 4.52 [1]

- Do funkce propojení vstupu a výstupu doplněna možnost Set/Reset.
- Do funkce Termostat doplněny další dva režimy hlídání teploty. Viz režimy 7 a 8 v Dodatku 1.

Verze 4.50

- Revize dokumentu.
- Do instrukce čtení statusu přidána možnost zjistit dobu uplynulou od restartu zařízení (run time).
- Možnost strojově přečíst počet vstupů a výstupů.
- Funkce propojení vstupu a výstupu umí reagovat na výpadek spojení.

Verze 3.39 / 4.39

- Přidána funkce pro nastavení samovolného vysílání změn na jednotlivých vstupech. Přibyla díky tomu jedna automatická zpráva s kódem 0x0C, která na rozdíl od 0x0D posílá vždy jen stav konkrétního vstupu, který změnil svůj stav. Nové jsou také konfigurační propojky na nejnovějších verzích Quid.

Verze 3.34

- Upravena funkce *Odečet od čítače* tak, aby bylo možné smazat všechny čítače najednou.

Verze 3.33

- Nová funkce: *Propojení vstupu a výstupu – 0x40/0x41*, která umí nastavit provázání vstupů a výstupů.

Verze 3.30

- Nové funkce: Měření teploty: int, float, string – 0x58, Teplotní jednotka – 0x1C/0x1D, Hlídání teplotních mezí – 0x13/0x14, Timeout komunikace – 0xE5/0xF5, Výchozí nastavení – 0x8F.
- Instrukce Jméno a verze – 0xF3 upravena tak, aby mohla být použita pro vyhledávání zařízení na lince.
- Přidána možnost přepnout komunikační protokol také do režimu, kdy je vypnutý ASCII formát 66 a lze komunikovat jen binárním formátem 97 (*Přepnutí komunikačního protokolu*).
- Quido umí odeslat aktuální komunikační parametry také zkratováním dvojice vývodů na desce elektroniky (viz str. 5).

[1] Verzí se má na mysli *[hw-version].[sw-version]* jak je uvedena v instrukci Jméno a verze – 0xF3. Tj. zařízení s identifikací *Quido USB 4/4; v0253.**04.38**; f66 97; t1* je verze **4.38**.

Verze 2.28

- Rozšíření instrukce Notifikace o změnách: Všechny vstupy – 0x10/0x11 o parametr *mask*. To znamená, že lze nastavit, aby se sledovaly jen některé vybrané vstupy. Funkce se hodí v situaci, kdy na jednom Quidu jsou použity vstupy v režimu čítač (s rychlými změnami) a zároveň vstupy od dveřních kontaktů.

Verze 2.21

- Odstraněna chyba v instrukci 0x33 (Pulz na výstupu – 0x23/0x33). Při zadání parametru (out) 0x00 (tedy čtení nastavení všech výstupů najednou) byla vracena sice korektní data, ale s ACK 0x03.
- Odstraněna chyba v adrese 0xFD, která byla také vyhodnocována jako univerzální adresa.

Verze 2.20

- Úprava v instrukci „Pulz na výstupu“. Jako instrukční kód v protokolu 66 lze použít kromě řetězce „OT“ i „OST“.
- Byly přidány instrukce pro uložení a přečtení uživatelských dat (například názvů) pro každý vstup a každý výstup.
- Nově možnost uložit pro každý výstup délku a polaritu pulzu. Ke spuštění pulzu je určena samostatná instrukce.
- Přidána instrukce pro rychlé zjištění režimu výstupu.
- Přeskupení pořadí instrukcí a jejich tematické rozdělení.

Komunikační parametry modulů Quido

Rozhraní RS232 a RS485

Komunikační rychlost.....nastavitelná 1200 Bd až 230400 Bd

Výchozí komunikační rychlost .....9600 Bd

Počet datových bitů 8

Parita .....bez parity

Počet stopbitů.....1

Rozhraní USB

Komunikační rychlost.....115200 Bd (pevně nastavená)

Počet datových bitů 8

Parita .....bez parity

Počet stopbitů.....1

Rozhraní Ethernet

Komunikační rychlost.....115200 Bd (pevně nastavená)

Počet datových bitů 8

Parita .....bez parity

Počet stopbitů.....1

Jak zjistit aktuální komunikační parametry Quida?

To, jaké komunikační parametry má Quido nastaveno lze zjistit pomocí jedné ze zkratovacích propojek na desce Quida. Postup je uveden na straně 7.

O SPINELU

Dokument popisuje komunikační protokol Spinel I/O modulů řady Quido. Konkrétní počty vstupů a výstupů, rozsahy komunikačních rychlostí a popis rozhraní je popsán v katalogovém listu konkrétního I/O modulu Quido. Případné výjimky jsou popsány u každé konkrétní instrukce.

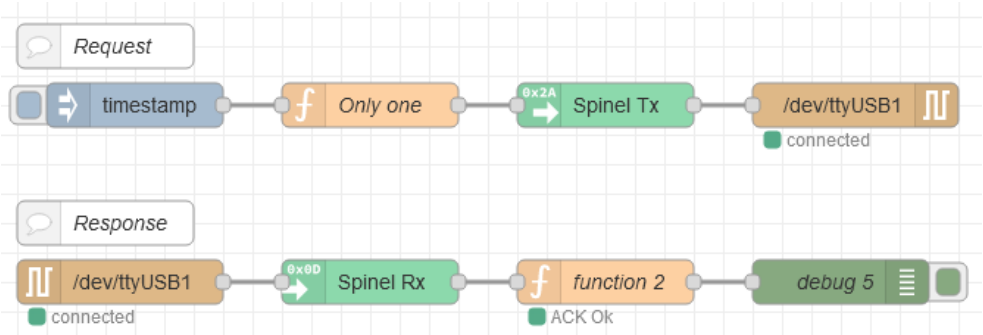
Online parser Spinelu

Na papouch.com/spinel-parser-online je univerzální nástroj k vytváření a parsování paketů protokolu Spinel.

1.	PRE: **2AH**	FRM: **61H**	NUM: **0006H**	ADR: **01H**	SIG: **02H**	INST: **20H**	DATA: [**82H**]	SUM: **C9H**	CR: **0DH**	link copy
2.	PRE: **2AH**	FRM: **61H**	NUM: **0005H**	ADR: **01H**	SIG: **02H**	ACK: **00H**	DATA: [ ]	SUM: **6CH**	CR: **0DH**	link copy
3.	PRE: **2AH**	FRM: **61H**	NUM: **0005H**	ADR: **01H**	SIG: **02H**	ACK: **00H**	DATA: [ ]	SUM: **66H**	CR: **0DH**	link copy

Node-RED

V článku na papouch.com najdete podrobný popis práce se Spinelem v Node-REDu.



Knihovna Spinel.NET v C#

Protokolem Spinel komunikuje také knihovna Spinel.NET pro prostředí .NET, která je zdarma ke stažení na GitHubu. Knihovna má kompletní dokumentaci v češtině.

Příklad sepnutí výstupu OUT 3 na 5 sec (dokumentace zde):

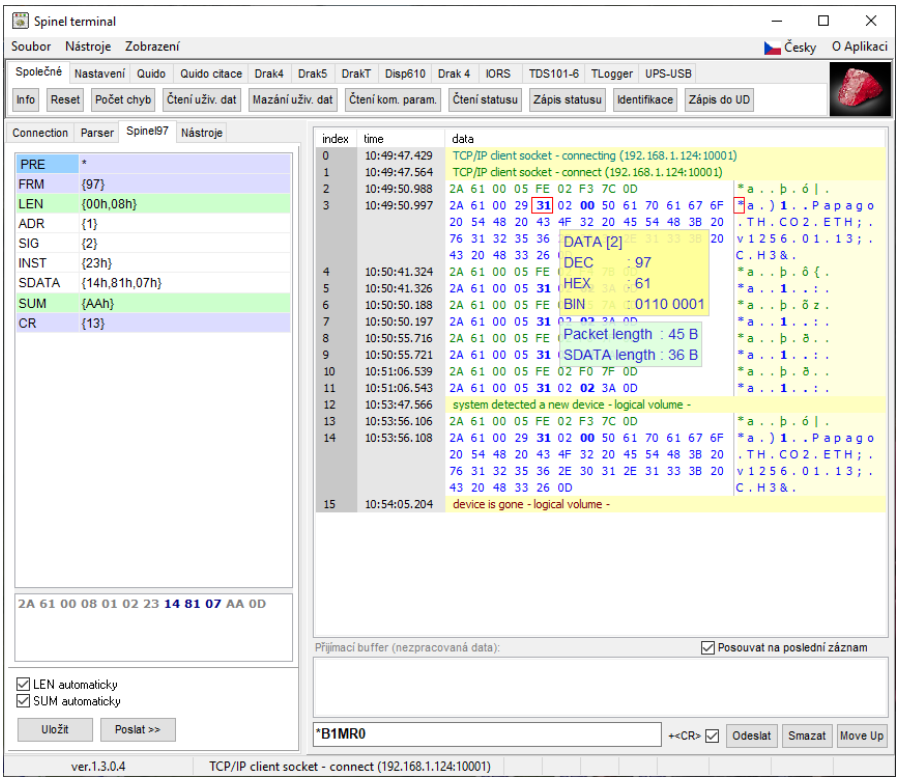
```
if (MyQuido.CmdSetOutput(3, true, 10))
    Console.WriteLine("Ok");
else
    Console.WriteLine("Error");
```

Příklad přečtení počtu jednotek z počítadla na vstupu IN 2 (dokumentace zde):

```
if (MyQuido.CmdGetCounter(2, out int counter))
    Console.WriteLine($"Counter value is {counter}.");
```

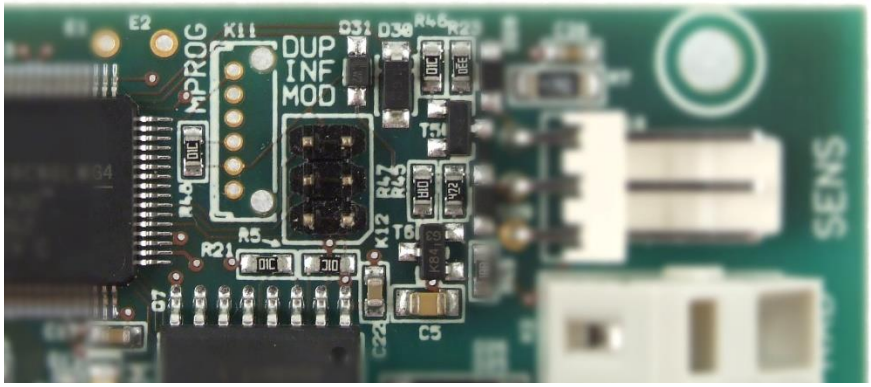
Spinel terminal

Pro snadnější ladění komunikace protokolem Spinel je k dispozici zdarma ke stažení na papouch.com/spinel program Spinel terminal pro Windows. Umožňuje komunikaci přes sériové porty i přes Ethernet, binárním protokolem Spinel (formát 97).



Konfigurační propojky

Na Quidech se od roku 2020 postupně po revizích hardwaru objevují konfigurační propojky, které usnadňují použití Quid v některých typických situacích.



obr. 1 - příklad propojek na Quido RS 2/2

Propojky jsou tři – Duplex, Info a Modbus. (Jak je vidět na obrázku, někdy je popisek v potisku mírně zkrácený.)

Propojka Modbus

Pokud je v okamžiku zapnutí napájení tato propojka zkratována, komunikuje Quido protokolem Modbus bez ohledu na softwarovou konfiguraci.

Propojka Info

Pokud je při připojeném napájení tato propojka krátce zkratována, Quido pošle na sériovou linku aktuální nastavení komunikačních parametrů.[2] Tato informace se posílá vždy v protokolu Spinel. RS verze zařízení posílá informaci na rychlosti 9600 Bd, USB a ETH verze na 115,2 kBd.

Quido odešle nejdříve odpověď na instrukci Jméno a verze – 0xF3 a poté ještě paket, kde je v datech uvedena v ASCII formátu adresa, rychlost a protokol. Příklad:

```
*a?"4N?Address:34 Speed:6 Protocol:1ü?
```

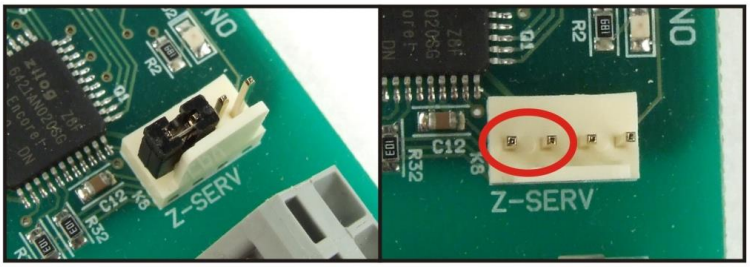
Adresa je hexadecimální, rychlost je kód dle instrukce *Komunikační parametry* a protokol je číslo protokolu podle instrukce *Přepnutí komunikačního protokolu*.

Propojka Info nesmí být zkratována při startu nebo restartu zařízení!

Propojka Duplex

Propojkou se aktivuje režim obousměrného přenosu stavu vstupů a výstupů mezi dvěma Quidy 4/4 nebo 8/8. Sadu je možné objednat jako QuidoDuplexRS nebo QuidoDuplexETH. V manuálech u těchto sad jsou další informace ohledně konfigurace tohoto režimu.

[2] Na dřívějších verzích Quid je možné stejnou akci vyvolat zkratováním dvou pinů na konektoru Z-SERV. Umístění pinů na starších Quidech je patrné z obrázku:



TEXTOVÝ KOMUNIKAČNÍ FORMÁT 66

Jak snadno ovládat Quido

Příklady jsou psány pro jednoduchost v tzv. „formátu 66“, který je vhodný pro seznámení s modulem, ladění a komunikaci pomocí terminálu.

Pro ovládání pomocí Vaší aplikace **v běžném provozu, doporučujeme používat formát 97**, který obsahuje kontrolní součet, a který je blíže popsán v kapitole, začínající na straně 14.

Následující příklady předpokládají komunikaci s modulem ve výchozím nastavení. Ovládacím programem odešlete řetězec uvedený ve sloupci Dotaz. Mezi jednotlivými znaky nesmí být prodleva delší než 5 sec. Pokud je vše v pořádku, modul odpoví tak, jak je uvedeno v následujícím řádku ve sloupci Odpověď.

Sepnutí relé

Následující příklad sepne relé číslo 2 na modulu s adresou 1.

Dotaz	Odpověď		Vysvětlení
*B1OS2H↵		*B	Prefix
		1	Adresa Jako adresu lze také použít znak $. Tento znak je univerzální adresou a funguje, pokud je na lince jen jeden modul.
		OS	Kód instrukce pro změnu stavu výstupu
		2	Číslo výstupu
		H	Kód sepnutí (High)
		↵	Ukončovací znak (enter, 0x0D)
	*B1O↵	*B	Prefix
		1	Adresa modulu
		O	Potvrzení
		↵	Ukončovací znak (enter, 0x0D)

Rozepnutí relé

Následující příklad rozepne relé číslo 4 na modulu s adresou D.

Dotaz	Odpověď		Vysvětlení
*BDOS4L↵		*B	Prefix
		D	Adresa Jako adresu lze také použít znak $. Tento znak je univerzální adresou a funguje, pokud je na lince jen jeden modul.
		OS	Kód instrukce pro změnu stavu výstupu
		4	Číslo výstupu
		L	Kód rozepnutí (Low)
		↵	Ukončovací znak (enter, 0x0D)
	*BDO↵	*B	Prefix
		D	Adresa modulu

		0	Potvrzení
		↵	Ukončovací znak (enter, 0x0D)

Čtení stavu vstupu

Příklad čtení stavu vstupu 3 na jediném připojeném modulu na lince (je použita univerzální adresa).

Dotaz	Odpověď		Vysvětlení
*B$IR3↵		*B	Prefix
		$	Univerzální adresa
		IR	Kód instrukce pro čtení stavu vstupu
		3	Číslo vstupu
		↵	Ukončovací znak (enter, 0x0D)
	*B10H↵	*B	Prefix
		1	Adresa modulu
		0	Potvrzení
		H	Vstup je aktivní
		↵	Ukončovací znak (enter, 0x0D)

Změna adresy

Instrukce změní adresu modulu z **f** na **5**.

Dotaz	Odpověď		Vysvětlení
Nejdříve je nutné povolit speciální instrukcí konfiguraci. Tato instrukce povolí konfiguraci pro bezprostředně následující instrukci. Po jakékoli následující instrukci je konfigurace opět zakázána.			
*BfE↵		*B	Prefix
		f	Adresa
		E	Kód instrukce pro povolení konfigurace
		↵	Ukončovací znak (enter, 0x0D)
	*Bf0↵	*B	Prefix
		f	Adresa modulu
		0	Potvrzení
		↵	Ukončovací znak (enter, 0x0D)
Nyní máme povolenu konfiguraci. Můžeme tedy změnit adresu.			
*BfAS5↵		*B	Prefix
		f	Stará adresa
		AS	Kód instrukce pro změnu adresy
		5	Nová adresa
		↵	Ukončovací znak (enter, 0x0D)
	*Bf0↵	*B	Prefix
		f	Stará adresa
		0	Potvrzení
		↵	Ukončovací znak (enter, 0x0D)

Popis textového formátu 66

Formát 66 používá jen dekadické proměnné nebo znaky, které lze psát na běžné klávesnici. Tento formát je proto vhodný při ladění aplikací se Spinelem. Mezi jednotlivými znaky nesmí být prodleva delší než 5 sec. Instrukce jsou rozděleny na dotaz odpověď:

Struktura

Dotaz:	PRE	FRM	ADR	INST	[DATA]	CR
Odpověď:	PRE	FRM	ADR	ACK	[DATA]	CR

- `PRE` Prefix, znak `*`
- `FRM` Číslo formátu 66 (znak `B`).
- `ADR` Adresa zařízení, na které je odeslán požadavek nebo které odesílá odpověď. (Jeden znak ASCII.)
- `INST` Kód instrukce – jeden ASCII znak z rozsahu `A` až `Z`, a až `z` a číslovky `0` až `9`.
- `ACK` Potvrzení dotazu (Acknowledge), zda a jak byl proveden. ACK je jeden znak z rozsahu `0` až `9` a `A` až `F`.
- `DATA` Přenesená data. Žádný, jeden nebo více ASCII znaků. Doporučujeme přenášet data v běžném textovém tvaru a jednotkách. Data nesmí obsahovat prefix (`*`) ani ukončovací znak (CR).
- `CR` Zakončovaní znak (sufix) [3]

Vysvětlivky

*	B	1	TR	↵
Prefix	Formát 66	Adresa	Kód instrukce	Suffix [3]

ADR – Adresa

Adresa je jeden znak, který jednoznačně určuje konkrétní zařízení mezi ostatními na jedné komunikační lince. Zařízení toto číslo vždy používá pro svou identifikaci v odpovědích na dotazy z nadřazeného systému. Adresou mohou být tyto ASCII znaky: číslice „0“ až „9“, malá písmena „a“ až „z“ a velká „A“ až „Z“. Adresa nesmí být shodná s prefixem nebo CR.

Adresa „%“ je rezervována pro „broadcast“. Pokud je v dotazu adresa „%“, zařízení se chová tak, jako by byla uvedena jeho adresa. Na dotazy s touto adresou se nevrací žádná odpověď.

Adresa „$“ je univerzální adresa. Pokud je v dotazu adresa „$“, zařízení se chová tak, jako by byla uvedena jeho adresa. V odpovědi zařízení uvede skutečnou právě nastavenou adresu. Univerzální adresa se používá jen v případech, kdy je na lince připojené pouze jedno zařízení.

Kód instrukce (INST)

Kód instrukce příslušného zařízení.

Je-li přijata platná instrukce (souhlasí ADR) a je nastaven příznak přijaté zprávy, zařízení na takovou instrukci již musí odpovědět.

[3] Znak „návrat vozíku“ (Carriage Return) s kódem ASCII 13 (dekadicky) nebo 0x0D (hexadecimálně). V příkladech je nahrazen znakem ↵

Potvrzení dotazu (ACK)

ACK informuje nadřazené zařízení o způsobu zpracování přijaté instrukce. Kódy potvrzení:

0.....VŠE V POŘÁDKU
Instrukce byla v pořádku přijata a kompletně provedena.

1.....OBECNÁ CHYBA

2.....NEPLATNÝ KÓD INSTRUKCE

3.....NEPLATNÁ DATA
Data nemají platnou délku nebo obsahují neplatnou hodnotu.

4.....NEPOVOLENO Z NĚKTERÉHO Z TĚCHTO DŮVODŮ
- Pokus o změnu nastavení bez předchozí instrukce Povolení konfigurace.
- Pokus o zápis do nepřístupné paměti.
- Požadovanou funkci zařízení nelze provést, protože pro to nejsou splněny podmínky. Například je vyžadována vyšší komunikační rychlost.
- Přístup do paměti chráněné heslem.

5.....PORUCHA ZAŘÍZENÍ
- Porucha zařízení, vyžadující servisní zásah.
- Chyba vnitřní paměti zařízení nebo paměti nastavení.
- Chyba některé vnitřní periferie zařízení (běhová chyba nebo chyba při inicializaci).
- Jakákoli jiná chyba ovlivňující správnou funkci zařízení.

6.....NEJSOU K DISPOZICI ŽÁDNÁ DATA

D .....AUTOMATICKY VYSLANÁ INSTRUKCE – ZMĚNA STAVU DIGITÁLNÍHO VSTUPU

Data (DATA)

Data instrukce.

Seznam základních instrukcí formátu 66

Popis	Kód [Dotaz] [Odpověď]	Příklad (adresa v příkladu vždy 1)
Čtení vstupu	`*B[adresa]IR[vstup]`[4]↵ `*B[adresa]0[stav]`[5]↵	`*B1IR2`↵ `*B10H`↵
Čtení výstupu	`*B[adresa]OR[výstup]`[6]↵ `*B[adresa]0[stav]`[7]↵	`*B1OR4`↵ `*B10L`↵
Nastavení výstupu	`*B[adresa]OS[výstup]`[6]`[stav]`[7]↵ `*B[adresa]0`↵	`*B1OS3H`↵ `*B10`↵
Nastavení časování výstupu	`*B[adresa]OT[výstup]`[6]`[stav]`[7]`[čas]`[8]↵ `*B[adresa]0`↵	`*B1OT1H20`↵[9] `*B10`↵
Dotaz na jméno a typ zařízení	`*B[adresa]?`↵	

[4] Číslo 0 až 4. Pokud je zadána nula, odešle se stav všech vstupů najednou.

[5] L – vstup neaktivní; H – vstup aktivní

[6] Číslo 1 až 4.

[7] L – rozepnutý kontakt; H – sepnutý kontakt

[8] Doba sepnutí/rozepnutí vybraného výstupu. Je možné zadat číslo 1 – 255. Jednotka je 0,5 sec. Je tedy možné nastavit čas 0,5 sec až 127,5 sec.

[9] Sepnutí výstupu 1 na 10 sec (10 sec = 20 * 0,5).

	*B[adresa]0Quido [line][10] [I/O][11]; v[VC][12]; F66 97↵	
Povolení konfigurace[13]	*B[adresa]E↵ *B[adresa]0↵	*B1E↵ *B10↵
Nastavení adresy[14]	*B[stará adresa]AS[nová adresa]↵ *B[stará adresa]0↵	*B1AS5↵ *B10↵

Poznámky:

[adresa].....Jako [adresa] může být použit také znak $, který představuje univerzální adresu. Lze jej použít, pokud je na lince jen jeden modul. Není jej v tom případě nutné adresovat.

[adresa].....Adresou může být také znak %. Pak jde o tzv. broadcast. To znamená, že jsou osloveny všechny moduly na lince, všechny provedou daný příkaz, ale nijak na něj nezareagují, aby nedošlo ke kolizi na lince.

↵.....Ukončovací znak „enter“, dekadicky 13, hexadecimálně 0x0D, escape character \r.

Komunikační rychlost Bd	Kód
1200	3
2400	4
4800	5
9600	6
19200	7
38400	8
57600	9
115200	A
230400	B

[10] Označení komunikačního rozhraní (USB, ETH nebo RS).

[11] Počet vstupů/počet výstupů (například 10/1 pro verzi s deseti vstupy a jedním výstupem).

[12] Výrobní číslo zařízení (například 0227.02.02).

[13] U této instrukce není možné použít universální adresu $.

[14] Této instrukci musí předcházet instrukce Povolení konfigurace

BINÁRNÍ KOMUNIKAČNÍ FORMÁT 97

Zařízení komunikuje binárně – tento způsob se označuje jako tzv. „formát 97“ (úvodní znak má dekadický kód 97). Při komunikaci se používají binární 8-bit znaky (dekadicky v rozsahu 0 až 255, hexadecimálně 0x00 až 0xFF).

Pro vývojáře je určený komfortní program Spinel Terminál (pro Windows) a také online parser a validátor Spinelu.

Následují dva typické příklady struktury dotazu a odpovědi. Na prvním řádku jsou názvy jednotlivých bytů nebo skupin a na druhém řádku je i konkrétní příklad dotazu, resp. odpovědi.

→ Dotaz:

PRE	FRM	NUM	NUM	ADR	SIG	INST	[DATA]	SUM	CR
2A	61	00	07	31	02	61	38, E6	BB	0D

← Odpověď:

PRE	FRM	NUM	NUM	ADR	SIG	ACK	[DATA]	SUM	CR
2A	61	00	05	31	02	00		3C	0D

pole	*délka v bytech*	***popis***
PRE	1	**Prefix. Vždy 0x2A, znak *, dekadicky 42.**
FRM	1	**Číslo formátu 97. Vždy 0x61, znak a, dekadicky 97.**
NUM	2	**Počet bytů v instrukci od následujícího bytu do konce zprávy (tj. ADR až CR).**
ADR	1	**Adresa zařízení, kterému je posílán dotaz nebo který posílá odpověď.**
SIG	1	**Podpis** zprávy. SIG poslané v dotazu, se vrátí v odpovědi. Jde tak poznat ke kterému dotazu patří odpověď.
INST	1	**Kód instrukce** (0x10 až 0xFF).
ACK	1	**Potvrzení dotazu** (0x00 až 0x0F) informuje, zda byl dotaz přijat, nastala chyba, jde o automatickou zprávu atd. Výčet standardních ACK je uveden dále.
DATA	x	**Data zprávy**. Délka pole a obsah se liší podle konkrétní instrukce.[15]
SUM	1	**Kontrolní součet.** Počítá se takto: `SUM = 0xFF – ((PRE + FRM + ((NUM & 0xFF00) >> 8) + (NUM & 0xFF) + ADR + SIG + ACK_INST + DATA) & 0xFF)`
CR	1	**Zakončovaní znak** (Carriage Return, \r). Vždy 0x0D, dekadicky 13.

- **NUM:** Počet bytů od ADR (včetně), až po CR (včetně). Jde o dva byty, takže NUM může být až 65535. Minimum je 5, což odpovídá instrukci, která neobsahuje žádná data. Je-li NUM menší než 5, není paket platný. Horní byte je MSB, dolní je LSB. Je-li počet bytů menší než 256, je horní byte nulový.

[15] **Použití závorek v popisu parametrů:** Pokud parametr v hranatých závorkách **[ ]**, jde o nepovinný parametr. Pokud je jeden nebo více parametrů v kulatých závorkách **( )**, může se takto ohraničená skupina opakovat. **Znak |** mezi dvěma parametry nebo skupinami parametrů znamená nebo – lze zadat jeden nebo druhý parametr (resp. skupinu parametrů).

- **ADR:** Adresa zařízení může být z rozsahu 0x00 až 0xFD (253). Následující adresy jsou rezervovány pro speciální použití:
  - 0xFF (255) je broadcast. To znamená, že pokud zařízení přijme zprávu s touto adresou, zařízení se chová jako by šlo o jeho adresu, instrukci provede, ale nepošle žádnou odpověď. S touto adresou nelze provádět konfiguraci.
  - 0xFE (254) je univerzální adresa. Pokud zařízení přijde zpráva s touto adresou, zařízení se chová jako by šlo o jeho adresu, instrukci provede, a pošle odpověď. Univerzální adresa se dá použít jen v případě, že je na komunikační lince jen jedno zařízení. S touto adresou nelze provádět konfiguraci.
- **ACK** (acknowledge) je v odpovědi na stejném místě jako je v dotazu INST. Je z rozsahu 0x00 až 0x0F. Tímto bytem zařízení informuje o tom, jak dopadlo přijetí poslední přijaté instrukce. Vyhrazené kódy ACK jsou tyto:
  - 0x00 Vše v pořádku: Instrukce byla přijata a provedena.
  - 0x01 Obecná chyba: Blíže nespecifikovaná chyba.
  - 0x02 Neznámý kód INST: Kód instrukce zařízení nezná.
  - 0x03 Chyba v datech: DATA mají nečekanou délku nebo obsahují nečekanou hodnotu.
  - 0x04 Nedovoleno z některého z těchto důvodů:
    - Pokus o změnu nastavení bez předcházejícího *Povolení konfigurace*.
    - Pokus o zápis do nepřístupné paměti.
    - Požadovanou funkci zařízení není možné provést, protože pro to nejsou splněné podmínky. Například je potřeba vyšší komunikační rychlost.
    - Přístup do paměti chráněné heslem.
  - 0x05 Porucha:
    - Zařízení vyžaduje servisní zásah.
    - Chyba vnitřní paměti zařízení nebo paměti nastavení.
    - Chyba některé vnitřní periferie zařízení.
    - Jakákoli jiná chyba ovlivňující správnou funkci zařízení.
  - 0x06 Nejsou k dispozici žádná data: Například krátce po zapnutí zařízení ještě nemusí být k dispozici hodnoty z externích senzorů atp.
  - 0x0A až 0x0F jsou zprávy, které zařízení poslalo automaticky bez dotazu z nadřízeného systému. Například notifikace o změně na vstupu, periodické měření, logy atd.
- **SUM** je kontrolní součet. Na zprávu s chybným kontrolním součtem se neodpovídá. Na příjem CR se čeká, i pokud přijde nesprávný kontrolní součet.

Příklady

V popisech konkrétních instrukcí na dalších stranách jsou příklady uvedeny takto:

Příklad 1:

→ 2A 61 00 08 31 02 **40 00 01 87** 71 0D
← 2A 61 00 05 31 02 **00** 3C 0D

- Příklady jsou v hexadecimálním formátu.[16]
- Šipka → v příkladech znamená dotaz, šipka ← znamená odpověď.

[16] (Pokud není uvedeno jinak.) Hexadecimálně znamená, že dekadické číslo 142 je uvedeno jako 8E, číslo 11 jako 0B. Více o hexadecimálním vyjádření čísel je například ve Wikipedii v článku Šestnáctková soustava.

- Pokud u instrukcí na následujících stranách není uveden příklad odpovědi, jde o standardní potvrzení dotazu s ACK 00 jak je uvedeno v příkladu 1 výše.

INSTRUKCE FORMÁTU 97

Vstupy

Pokud zařízení nemá vstupy, odpovídá ACK 0x02 (neplatná instrukce).

Stav vstupů – 0x31

Instrukce čte aktuální stav vstupů.

Parametry

data	1-13 byte	Každý bit představuje jeden vstup. Počet byte tedy záleží na počtu vstupů. (8 vstupů → 1 byte, 10 vstupů → 2 byte, 16 vstupů → 2 byte, …) Nejnižší bit v každém byte je vstup s nejnižším pořadovým číslem (MSb). Neosazené vstupy jsou vždy 0. • Pokud stačí 1 byte: Osm vstupů v pořadí 87654321. (8 je MSb) • Při dvou bytech: $[^{16}{}_{15}{}^{14}{}_{13}{}^{12}{}_{11}{}^{10}{}_{9}][^{8}{}_{7}{}^{6}{}_{5}{}^{4}{}_{3}{}^{2}{}_{1}]$ • … • Délka 13 byte: $[^{104}{}_{103}{}^{102}{}_{101}{}^{100}{}_{99}{}^{98}{}_{97}]$[12B][11B][10B][9B][8B][7B][6B][5B][4B][3B][2B]$[^{8}{}_{7}{}^{6}{}_{5}{}^{4}{}_{3}{}^{2}{}_{1}]$

Čtení

Struktura:	→ **0x31** ← data
Příklad 1: *Quido 8/8*	→ 2A 61 00 05 01 02 **31** 3B 0D ← 2A 61 00 06 01 02 **00 C2** A9 0D • 0xC2, binárně 11000010 → vstupy 8, 7 a 2 jsou aktivní (tj. log. 1, připojeno napětí, sepnuto apod. dle typu vstupu). • Podle délky dat je vidět, že jde o Quido s osmi vstupy.
Příklad 2: *Quido 10/1*	→ 2A 61 00 05 01 02 **31** 3B 0D ← 2A 61 00 07 01 02 **00 02 C2** A6 0D • 0x02C2, binárně 0000 0010 1100 0010 → vstupy 10, 8, 7 a 2 jsou aktivní • Podle délky dat je vidět, že jde o Quido s maximálně 16ti vstupy.

Čtení 66:

→ `*B1IR29↵`

- 29: Čtení stavu vstupu 29.

← `*B10L↵`

- L: Vstup je v log. 0 (neaktivní). Pokud by byl vstup aktivní, bylo by zde H.

Notifikace o změnách: Všechny vstupy – 0x10/0x11

Tato funkce umožňuje automaticky informovat nadřazený systém o změně stavu některého ze vstupů. V odpovědi je vždy stav všech vstupů. Díky této funkci není nutné periodicky zjišťovat

stav vstupů. Při rychlých změnách na vstupech je třeba počítat také s rychlostí odesílání, aby odesílání informace o změnách bylo technicky možné. (Funkce 0x10/0x11 nijak nesouvisí s funkcemi 0x15/0x16.)

- Quida RS s linkou RS485: Notifikace je možné použít jen pokud je na RS485 jedno zařízení! Ve výchozím nastavení je odesílání notifikací vypnuté.
- Quida USB mají ve výchozím nastavení notifikace vypnuté.
- Quida ETH mají ve výchozím nastavení notifikace zapnuté.

Automatická notifikace se pak posílá ve stejném formátu protokolu Spinel, jako byl formát instrukce, kterou byly notifikace zapnuté.

Parametrem *mask* (jen v binárním formátu 97) lze určit které vstupy způsobí odeslání notifikace a které ne.

Parametry

enable	1 byte	V dotazu:
		• 0x01: Zapnout notifikace
		  ○ Před opětovným zapnutím nebo změnou masky musí být notifikace nejdříve vypnuté!
		  ○ Při zapnutí notifikací bez parametru *mask*, se interně nastaví notifikace ze všech vstupů.
		• 0x00: Vypnout notifikace
		  ○ Vypnutím se současně vynuluje parametr *mask*.
		V odpovědi:
		• 0x00: Notifikace jsou vypnuté
		• 0x42: Notifikace jsou zapnuté instrukcí ve formátu 66
		• 0x61: Notifikace jsou zapnuté instrukcí ve formátu 97

mask	1-13 byte	Bitová maska (pro každý vstup jeden bit) určuje, jestli změna na daném vstupu způsobí odeslání notifikace (1) nebo ne (0). Tento parametr není povinný. Způsob zápisu bitů pro jednotlivé vstupy je shodné s funkcí Stav vstupů – 0x31 na straně 16.
data	1-13 byte	Každý bit představuje jeden výstup. Počet byte tedy záleží na počtu výstupů. (< 8 výstupů → 1 byte, < 16 výstupů → 2 byte, …) Nejnižší bit v každém byte je výstup s nejnižším pořadovým číslem (MSb). Neosazené výstupy jsou vždy 0. • Pokud stačí 1 byte: Osm výstupů v pořadí 87654321. (8 je MSb) • Při dvou bytech: $[^{16}{}_{15}{}^{14}{}_{13}{}^{12}{}_{11}{}^{10}{}_{9}][^{8}{}_{7}{}^{6}{}_{5}{}^{4}{}_{3}{}^{2}{}_{1}]$ • … • Délka 13 byte: $[^{104}{}_{103}{}^{102}{}_{101}{}^{100}{}_{99}{}^{98}{}_{97}]$[12B][11B][10B][9B][8B][7B][6B][5B][4B][3B][2B]$[^{8}{}_{7}{}^{6}{}_{5}{}^{4}{}_{3}{}^{2}{}_{1}]$

Zápis

Struktura:	→ **0x10**, enable, $[\text{mask}]^{15}$

Příklad:	→ 2A 61 00 08 31 02 **10** **01** **1C 03** 09 0D • 0x01: Zapnutí notifikací. • 0x1C03, binárně 0001 1100 0000 0011: Notifikace o stavu všech vstupů bude odeslána jen při změně stavu vstupů 1, 2, 11, 12 a 13.

Čtení

Struktura:	→ **0x11** ← enable, mask
Příklad:	→ 2A 61 00 05 31 02 **11** 2B 0D ← 2A 61 00 07 31 02 **00** **61** **03** D6 0D • 0x61: Automatické vysílaní je zapnuté formátem 97 (tj. 0x61) • 0x03, tj. 00000011: Notifikace o stavu všech vstupů bude odeslána jen při změně na vstupech 1 a 2.

Automatická notifikace

Struktura:	← **0x0D**, data
Příklad:	← 2A 61 00 06 31 02 **0D** **10** 1E 0D • 0x0D: Automatická zpráva o stavu všech vstupů. • 0x10, tj. 00010000: Vstup IN6 je aktivní (tj. v log.1, sepnutý apod.)

Zápis 66:

→ `*B1IS1↵`

- 1: Zapnutí automatických notifikací (0 pro vypnutí).

← `*B10↵`

Čtení 66:

→ `*B1IX↵`

← `*B10B↵`

- B: Automatické notifikace zapnuté formátem 66 (B). Případně „0“, pokud jsou notifikace vypnuté, nebo „a“, pokud byly zapnuté formátem 97.

Automatická notifikace 66:

Příklad notifikace z Quida s osmi vstupy:

← `*B1D LLLLL LHL↵`

- D: Označení automatické notifikace.
- LLLLL LHL: Aktivovaný je jen vstup 7. H znamená aktivní vstup, L znamená neaktivní. Pro lepší čitelnost je po pěti vstupech vložena mezera.

Notifikace o změnách: Jeden vstup – 0x15/0x16

Tato funkce umožňuje automaticky informovat nadřazený systém o změně stavu jednoho vstupu. Notifikace se pošle vždy jen se stavem konkrétního vstupu, který se změnil. (Funkce 0x15/0x16 nijak nesouvisí s funkcemi 0x10/0x11.)

Díky této funkci není nutné periodicky zjišťovat stav vstupů. Při rychlých změnách na vstupech je třeba počítat také s rychlostí odesílání, aby odesílání informace o změnách bylo technicky možné.

Parametry

enable	1 byte	Zapnout (0x01) nebo vypnout (0x00) notifikace. Týká se vždy všech vstupů najednou.
in	1 byte	Číslo vstupu. První je 0x01.
value	1 byte	Vstup je aktivní (0x01) nebo není aktivní (0x00).

Zápis

Struktura:	→ **0x15**, enable
Příklad:	→ 2A 61 00 06 31 02 **15 01** 25 0D • 0x01: Zapnutí notifikací.

Čtení

Struktura:	→ **0x16** ← enable
Příklad:	→ 2A 61 00 05 31 02 **16** 26 0D ← 2A 61 00 06 31 02 **00 00** 3B 0D • 0x00: Notifikace jsou vypnuté.

Automatická notifikace

Struktura:	← **0x0C**, in, value
Příklad:	← 2A 61 00 07 31 03 **0C** **05** **01** 27 0D • 0x0C: Automatická zpráva o změně na jednotlivém vstupu. • 0x05: Vstup IN5. • 0x01: Vstup je aktivní (log. 1, sepnutý).

Vzorkování – 0x62/0x63

Zařízení vzorkuje svoje vstupy každou milisekundu. Pokud je shodný zadaný počet vzorků jdoucích po sobě, je to vyhodnoceno jako změna stavu vstupu. Více vzorků vede k vyšší odolnosti proti zákmitům, ale delší reakční dobu na změnu stavu. Výchozí hodnotou je 20 ms.

Parametry

samples	1 byte	Počet vzorků.

Zápis

Struktura:	→ **0x62**, samples
Příklad:	→ 2A 61 00 06 31 02 **62 0A** CF 0D • 0x0A, tj. dekadicky 10 vzorků.

Čtení

Struktura:	→ **0x63** ← samples
Příklad:	→ 2A 61 00 05 31 02 **63** D9 0D ← 2A 61 00 06 31 02 **00 0A** 31 0D • 0x0A, tj. dekadicky 10 vzorků.

Čtení čítačů – 0x60

Zjištění stavu počítadel změn na vstupech. Čítač umožňuje počítat jednotlivé změny stavu vstupu. Za změnu je považována změna logického stavu (tj. např. stavu připojeného kontaktu). Každý z prvních šedesáti vstupů má vlastní čítač. K hodnotě čítače je přičtena jednička jen při nastaveném typu změny (změna z 1 do 0; změna z 0 do 1; případně obě změny). Stav čítačů není uchován po odpojení od napájení nebo po resetu!

Parametry

counter	1 byte	Bitově orientovaný byte ve tvaru `CXnnnnnn`:
		• C: Pouze čtení beze změny hodnoty čítače (0) nebo vynulování po přečtení (1). Ztrátu pulzů (změn) v krátkém okamžiku mezi čtením a nulováním čítače lze eliminovat postupem uvedeným u instrukce Odečet od čítače – 0x61.
		• nnnnnn: Číslo 0 až 60[17]:
		  ○ 0: Přečtení všech čítačů.
		  ○ Číslo větší než 0 znamená přečtení pouze konkrétního čítače a v takovém případě může v dotazu být parametr *counter* vícekrát.

bits	1 byte	Rozlišení čítačů v bitech. (Většinou 16 bitů.)
value	X byte	Aktuální hodnota počítadla. Dva nebo více byte podle rozlišení čítače (pro 16bit čítače dva byte v pořadí MSB:LSB). Parametrů *value* je v odpovědi tolik, o kolik stavů čítačů bylo požádáno v dotazu. Pořadí odpovídá dotazu. Pokud se posílají stavy všech, jako první se posílá čítač na vstupu IN1.

Čtení

Struktura:	→ **0x60**, (counter) [15] ← bits, (value)
Příklad:	→ 2A 61 00 06 31 02 **60 00** DB 0D • 0x00: Ptáme se na stav všech čítačů, bez nulování hodnot. ← 2A 61 00 1A 31 02 **00** **10** **01 23** **00 00** **00 AC** **00 00** **70 00** **00 31** **AA 00** **00 00** **00 00** **00 00** FC 0D • 0x10, tj. dekadicky 16: 16bit čítače • 0x0123, tj. dekadicky 291: Stav čítače na IN1. • 0x00AC: Stav čítače na IN3.

Čtení 66:

→ `*B1CR15`↵

- 1: Čítač po přečtení vynulovat (případně 0 pro čtení bez nulování).
- 5: Číslo čítače.

← `*B10230`↵

- 230: Čítač napočítal 230 změn.

[17] Quido s více než 60ti vstupy má čítače jen na prvních 60ti vstupech.

Odečet od čítače – 0x61

Instrukce odečte zadanou hodnotu od aktuálního stavu čítače. To vyloučí ztrátu pulzů (změn) v krátkém okamžiku mezi čtením a nulováním čítače, ke které může z principu dojít při použití instrukce 0x60.

Postup, jak předejít ztrátě pulzů:

1) Instrukcí Čtení čítačů – 0x60 přečtěte hodnotu čítače (bez nulování, tj. bit C=0).
2) Přečtenou hodnotu odečtěte touto instrukcí (0x61) od aktuálního stavu čítače. Díky tomuto postupu nebude ztracena žádná změna na vstupu.

Nelze odečíst číslo větší, než je okamžitá hodnota čítače.

Parametry

Pokud je zadaný jeden parametr *counter* a jeden *value*, obojí 0, smažou se všechny čítače najednou. Dvojic *counter+value* může být v dotazu více, maximálně 12.

counter	1 byte	Číslo čítače. Čítač na IN1 má číslo 0x01.[17]
value	X byte	Hodnota k odečtu. Dva nebo více byte podle rozlišení čítače (pro 16bit čítače dva byte v pořadí MSB:LSB).

Zápis

Struktura:	→ **0x61**, (counter, value) [15]
Příklad:	→ 2A 61 00 08 31 02 **61** **02** **00 01** D5 0D • 0x02: Čítač na IN2. • 0x0001: Odečíst jedničku.

Zápis 66:

→ `*B1CD021↵`

- 02: Čítač na IN2. Číslo vstupu je zde vždy dvoumístné.
- 1: Odečíst jedničku.

← `*B10↵`

Režim čítače – 0x6A/0x6B

Funkce nastavuje, jaké změny na vstupech čítač počítá.

Parametry

counter	1 byte	Bitově orientovaný byte ve tvaru `CCnnnnnn`: • CC: Režim čítače: ○ 00: Vypnutý čítač. ○ 10: Přičtení jednotky při změně stavu z log. 0 do 1. ○ 01: Přičtení jednotky při změně stavu z log. 1 do 0. ○ 11: Přičtení jednotky při jakékoli změně stavu. • nnnnnn: Číslo 0 až 60[17]: ○ 0: Nastavení všech čítačů. ○ Číslo větší než 0 znamená nastavení konkrétního čítače a v takovém případě může v dotazu být parametr *counter* vícekrát.

input	1 byte	Číslo vstupu. Hodnota 0 pro všechny čítače, nebo hodnota 1-60 pro konkrétní čítač (a v tom to případě může být v dotazu více parametrů *input* najednou).

Zápis

Struktura:	→ **0x6A**, (counter)
Příklad:	→ 2A 61 00 06 31 02 **6A 80** 51 0D • 0x80, tj. 10000000: Přičtení jednotky při změně stavu z 1 do 0 (CC=10) pro všechny čítače (n=000000).

Čtení

Struktura:	→ **0x6B**, (input) [15] ← (counter)
Příklad:	→ 2A 61 00 09 31 02 **6B** **<u>01</u>** **<u>05</u>** **<u>07</u>** **<u>09</u>** B7 0D • Ptáme se na režim čítačů na vstupech 1, 5, 7 a 9. ← 2A 61 00 09 31 02 **00** **<u>81</u>** **<u>C5</u>** **<u>47</u>** **49** 62 0D • 0x81: Čítač IN1 počítá změny z log. 1 do 0. • 0xC5: Čítač IN5 počítá všechny změny. • 0x47: Čítač IN7 počítá změny z log. 0 do 1.

Zápis 66:

→ `*B1CO15↵`

- 1: Počítání změn z 0 do 1. Lze také zadat 0 (vypnutý čítač), 2 (počítá změny z 1 do 0) nebo 3 (počítá všechny změny).
- 5: Číslo vstupu (čítače).

← `*B10↵`

Čtení 66:

→ `*B1CX3↵`

- 3: Ptáme se na nastavení čítače na vstupu 3.

← `*B11↵`

- 1: Počítají se změny z 0 do 1. (Další možnosti viz Zápis.)

Výstupy

Pokud zařízení nemá výstupy, odpovídá ACK 0x02 (neplatná instrukce).

Ovládání výstupů – 0x20

Základní instrukce pro ovládání výstupů – tedy okamžité sepnutí nebo rozepnutí.

Parametry

set	1 byte	Bitově orientovaný byte ve tvaru Soooooo: • S: Sepnout výstup (1) nebo rozepnout výstup (0). • ooooooo: Číslo výstupu z rozsahu 1 až 127.

Zápis

Struktura:	→ **0x20**, (set) [15]
Příklad:	→ 2A 61 00 06 01 02 **20** **82** C9 0D • 0x82, tj. 10000010: Sepnout (bit 7=1) výstup 2.

Čtení 66:

→ `*B1OS25H↵`

- 25: Nastavení výstupu 25.
- H: Sepnout (L pro rozepnout).

← `*B10↵`

Čtení výstupů – 0x30

Instrukce čte aktuální stav výstupů (relé).

Parametry

data	1-13 byte	Každý bit představuje jeden výstup. Nejnižší bit v každém byte je výstup s nejnižším pořadovým číslem. Neosazené výstupy jsou vždy 0. • Délka 1 byte: Výstupy 87654321. (8 je MSb) • Délka 2 byte: $[^{16}_{15}{}^{14}_{13}{}^{12}_{11}{}^{10}_{9}][^{8}_{7}{}^{6}_{5}{}^{4}_{3}{}^{2}_{1}]$ • … • Délka 13 byte: $[^{104}_{103}{}^{102}_{101}{}^{100}_{99}{}^{98}_{97}]$[12B][11B][10B][9B][8B][7B][6B][5B] [4B][3B][2B]$[^{8}_{7}{}^{6}_{5}{}^{4}_{3}{}^{2}_{1}]$

Čtení

Struktura:	→ **0x30** ← data
Příklad:	→ 2A 61 00 05 01 02 **30** 3C 0D ← 2A 61 00 06 01 02 **00** **11** 5A 0D • 0x11, binárně 00010001 → výstupy 5 a 1 jsou sepnuté

Čtení 66:

→ `*B1OR14↵`

- 14: Čtení stavu výstupu 14.

← `*B10H↵`

- H: Výstup je sepnutý. Pokud by byl výstup rozepnutý, bylo by zde L.

Pulz na výstupu – 0x23/0x33

Sepne nebo rozepne výstupy na zadanou dobu. Parametry pulzu se zadávají při spuštění pulzu. Pulz se spustí okamžitě po přijetí této instrukce.

Opětovné spouštění pulzu, když ještě neskončil předchozí, je možné. Opakované spouštění pulzu tedy je možné použít například v situaci, kdy není žádoucí, aby po výpadku spojení zůstal výstup sepnutý. Pokud bude na výstupu např. každou vteřinu spouštěn pulz na dobu 5 sec, po výpadku spojení relé nejpozději za 5 sec rozepne.

Zde nastavený čas nesouvisí s délkou pulzu uloženou instrukcí „Pulz na výstupu: Oddělené spouštění – 0x26/0x36“ na straně 25.

Parametry

time	1 byte	V dotazu: Čas, po který mají být nastaveny výstupy uvedené dále v parametrech *out*. Rozsah 1 až 255, jednotka je 0,5 sec. V odpovědi: Zbývající čas pulzu na výstupu. Nula znamená, že čas vypršel.
set	1 byte	Bitově orientovaný byte ve tvaru Soooooo: • S: Sepnout výstup (1) nebo rozepnout výstup (0). • oooooo: Číslo výstupu z rozsahu 1 až 127.
out	1 byte	Číslo výstupu: • 1 – 255: Číslo konkrétního výstupu. • 0: Dotaz na všechny výstupy na zařízení najednou.

Zápis

V jedné instrukci lze poslat maximálně dvanáct parametrů *set*.

Struktura:	→ **0x23**, time, (set) [15]
Příklad:	→ 2A 61 00 08 35 02 **23** **04** **81** **84** 09 0D • 0x04: Čas 2 sec (4 × 0,5 sec) • 0x81, tj. 10000001: Sepnout výstup 1. • 0x84, tj. 10001000: Sepnout výstup 4.

Čtení

Sekvencí *(set,time)* je v odpovědi tolik, kolik bylo v dotazu zadáno výstupů, respektive tolik, kolik je výstupů, pokud byla v dotazu zadána 0.

Struktura:	→ **0x33**, out ← (set, time)
Příklad:	→ 2A 61 00 06 31 02 **33** **00** 08 0D • 0x00: Ptáme se na všechny výstupy. ← 2A 61 00 0B 31 02 **00** **81 1B** **02 00** **83 09** 0C 0D • 0x811B: 0x81 (10000001) Výstup 1 bude sepnutý ještě 13,5 (0x1B × 0,5) sekund. • 0x201B: 0x20 (00000010) Výstup 2 je rozepnutý a nemá nastaven čas. • 0x8309: 0x83 (10000011) Výstup 3 bude sepnutý ještě 4,5 (9 × 0,5) sekund.

Zápis 66:

→ `*B1OST5H20↵`

- 5: Výstup 5.
- H: Sepnout (L pro rozepnout).
- 20: Čas v násobcích 0,5 sec. Rozsah 1 až 255 tedy znamená 0,5 až 127,5 sec.

← `*B10↵`

Čtení 66:

→ `*B1ORT3↵`

- 3: Dotaz na průběh pulzu na výstupu 3.

← `*B10H9↵`

- H: Sepnuto (L pro rozepnuto).
- 9: Výstup bude sepnutý ještě 4,5 sec (9 × 0,5 sec).

Pulz na výstupu: Oddělené spouštění – 0x26/0x36/0x25

Sepne nebo rozepne výstupy na zadanou dobu. Parametry pulzu se nastavují odděleně. Pulz se spouští samostatnou instrukcí. Parametry pulzu jsou uloženy v paměti i při výpadku napájení.

Zde nastavený čas nesouvisí s délkou pulzu uloženou instrukcí „Pulz na výstupu – 0x23/0x33“ na straně 25.

Parametry

out	1 byte	Číslo výstupu z rozsahu 1 až 255. (Při čtení může být zadána 0, což znamená přečtení nastavení všech výstupů.)
type	1 byte	Typ pulzu na výstupu: • 0x00: Žádný • 0x02: Kladný pulz (L → H → L) • 0x03: „Záporný“ pulz (H → L → H)
time	1 byte	Délka pulzu jako číslo 1 až 255, jednotka 0,5 sec. Při čtení nastavení se zde posílá nastavený čas, ne zbývající čas pulzu. (Zbývající čas probíhajícího pulzu se čte pomocí instrukce 0x33.)

Zápis (nastavení)

Instrukce může obsahovat více sekvencí *out, type, time* (na pořadí nezáleží). V jedné instrukci lze poslat těchto sekvencí maximálně dvanáct.

Struktura:	→ **0x26**, (out, type, time)
Příklad:	→ 2A 61 00 08 31 02 **26** **04** **02** **04** 09 0D • 0x04: Výstup 4. • 0x02: Kladný pulz. • 0x04: Čas 2 sec (4 × 0x,5 sec)

Čtení (čtení nastavení)

Instrukce může obsahovat více sekvencí *type, time*, v závislosti na počtu dotazovaných vstupů.

Struktura:	→ **0x36**, (out) ← (type, time)

Příklad:	→ 2A 61 00 06 31 02 **36 00** 05 0D • 0x00: ptáme se na nastavení všech výstupů. ← 2A 61 00 0D 31 02 **00** **03 14** **02 14** **00 00** **02 04** 01 0D • 0x0314: První v režimu „záporný pulz“ (0x03), doba 10 sec (0x14 × 0,5). • 0x0214: Druhý v režimu „kladný pulz“ (0x02), doba 10 sec (0x14 × 0,5). • 0x0000: Třetí výstup bez nastavení pulzu. • 0x0204: Čtvrtý v režimu „kladný pulz“ (0x02), doba 2 sec (0x04 × 0,5).

Spuštění pulzu

Pulz na výstupu se provede jen v případě, že na výstupu není aktivována funkce termostatu. Pokud je na výstupu aktivován termostat, lze pulz spustit jen, když je teplota mezi *tempx* a *tempy*. Jinak Quido odpoví ACK 0x03.

Instrukce může obsahovat více sekvencí *out, type, time* (na pořadí nezáleží). V jedné instrukci lze poslat těchto sekvencí maximálně dvanáct.

Struktura:	→ **0x25**, (out)
Příklad:	→ 2A 61 00 07 31 02 **25** **02** **04** 0F 0D • 0x02, 0x04: Spustit pulz na výstupech 2 a 4.

Zjištění režimu výstupu – 0x38

Tato instrukce umožňuje zjistit, v jakém režimu výstup právě pracuje.

Parametry

out	1 byte	Číslo výstupu z rozsahu 1 až 255 nebo 0 pro všechny výstupy.
flags	1 byte	Bitově orientovaný byte (bit 7 = MSB); význam jednotlivých bitů je následující:

7 A	**6** M1	**5** M0	**4**	**3** S3	**2** S2	**1** S1	**0** S0	**Bit** Název bitu	**Význam**
1	0	1	0	0	*S*	*S*	*K*		Výstup je v režimu termostatu. Bity SSK odpovídají bitům SSK z funkce termostatu (str. 31).
0	0	0	0	0	0	1	0		Výstup je v manuálním režimu a je nastavena délka kladného pulzu.
0	0	0	0	0	0	1	1		Výstup je v manuálním režimu a je nastavena délka záporného pulzu.
0	0	0	0	0	0	0	0		Výstup je v manuálním režimu a není nastaven žádný pulz.

Čtení

Instrukce může obsahovat více parametrů *out*. Podobně pak v odpovědi více parametrů *flags*.

Struktura:	→ **0x38**, (out) ← (flags)
Příklad:	→ 2A 61 00 06 31 02 **38** **00** 03 0D • 0x00: Ptáme se na všechny výstupu. ← 2A 61 00 09 31 02 **00** **A0** **02** **03** **A0** F3 0D • OUT1: statický režim

	• OUT2: kladný pulz • OUT3: záporný pulz • OUT4: hlídání teploty (bity SSK = 0)

Propojení vstupu a výstupu – 0x40/0x41

Umožňuje nastavit provázání vstupů spolu s výstupy. Aktivací vstupu aktivovat některý výstup apod. Tato funkce je dostupná pouze pro Quido 2/2, 4/4 a 8/8.

Parametry

in	1 byte	Číslo vstupu. První vstup má číslo 0x01.
out	1 byte	Číslo výstupu. První výstup má číslo 0x01.
fn	1 byte	Typ propojení vstupu a výstupu: • 0x00: Žádný. • 0x01: Kopírování: Stav vstupu se kopíruje na výstup aktivní vstup = sepnutý výstup). • 0x02: Spuštění pulzu: Hranou na vstupu (okamžikem změny) se spustí pulz na výstupu. • 0x03: Změna stavu hranou: Funguje jako dělička dvěma, tedy první hrana na vstupu sepne výstup, druhá rozepne výstup. • 0x04: Reakce na výpadek spojení: Sepnutí/rozepnutí výstupu. • 0x05: Set/Reset: Hrana na vstupu výstup sepne *nebo* rozepne. Jeden vstup může spínat nebo rozpínat, ne obojí.
edge	1 byte	Typ hrany (jen pro *fn*=2, *fn*=3 a *fn*=5): • 0x00: Náběžná hrana. • 0x01: Sestupná hrana. Délka výpadku spojení (jen pro *fn*=4): • Číslo z rozsahu 0x01 až 0x03 jako čas v sekundách.
param	1 byte	Inverze stavu (jen pro *fn*=1): Umožňuje invertovat (0x01) vstupní úroveň. Sepnutí vstupu rozepne výstup a obráceně. Prodlužování pulzu (jen pro *fn*=2): • 0x00: Opakovaná hrana na vstupu během pulzu na výstupu nemá na dobu trvání pulzu žádný vliv. • 0x01: Opakování hrany na vstupu prodlužuje délku sepnutí výstupu. Typ reakce na událost (jen pro *fn*=4 a *fn*=5): • 0x00: Rozepnout výstup. • 0x01: Sepnout výstup.

Zápis

Struktura:	→ **0x40**, in, out, fn, edge, param
Příklad:	→ 2A 61 00 0A 31 02 **40** **02** **02** **03** **00** **00** F0 0D • 0x02: Vstup IN2. • 0x02: Výstup OUT2. • 0x03: Funkce změny stavu hranou. • 0x00: Reaguje se na náběžnou hranu.

Čtení

Struktura:	→ **0x41**, in ← in, out, fn, edge, param
Příklad:	→ 2A 61 00 06 31 02 **41** **03** F7 0D • 0x03: Ptáme se na 3. vstup. ← 2A 61 00 0A 31 02 **00** **03** **04** **01** **00** **01** 2E 0D • 0x03: Vstup IN3. • 0x04: Výstup OUT4. • 0x01: Kopírování stavu 0x01: Inverze.

Měření a hlídání teploty

Upozornění: Pokud zařízení neumožňuje připojení teploměru, odpovídá na následující instrukce ACK 0x02 (neplatná instrukce).

Měření teploty – 0x51

Vrací teplotu naměřenou připojeným teploměrem. Teplota je vracena v nastavené teplotní jednotce.

Parametry

id	1 byte	Číslo teploměru (první má číslo 0x01). 0x00 znamená dotaz na všechny teploměry. Parametr může být v dotazu maximálně dvanáctkrát.
int	2 byte	Naměřená teplota. Hodnota MSB:LSB ve formátu signed int. Teplota se z hodnoty získá vydělením deseti. [18]

Čtení

Pokud je teplota mimo rozsah, odpovídá se ACK 0x05 (porucha zařízení). Při chybě senzoru je vyhlášeno ACK 0x05 po cca deseti sekundách trvání chyby.

Struktura:	→ **0x51**, (id) [15] ← (id, value)
Příklad:	→ 2A 61 00 06 31 02 **51** **01** E9 0D • 0x01: ptáme se na 1. teploměr. ← 2A 61 00 08 31 02 **00** **01** **00 F6** 42 0D • 0x01: První teploměr. • 0x00F6 je dekadicky 246, tj. teplota 24,6°

Čtení 66:

→ `*B1TR1`↵

- 1: Číslo jednoho teploměru, který má být přečten.

← `*B10+029.1C`↵

- +029.1C: Teplota jako ASCII řetězec (vždy 7 znaků zarovnaných doprava). Nepoužité znaky jsou vyplněny nulou (0x30). Jako oddělovač desetinných míst je použita tečka (0x2E).

[18] Skutečná přesnost použitého teplotního senzoru je uvedena v dokumentaci ke konkrétnímu modulu Quido.

- Je-li teploměr mimo rozsah nebo není možné načíst teplotu, odpovídá se ACK 5 (porucha zařízení).

Měření teploty: int, float, string – 0x58

Teplotu naměřenou připojeným teploměrem vrací (1) jako celé číslo vynásobené deseti, (2) jako plovoucí desetinné číslo a (3) jako ASCII řetězec.

Parametry

id	1 byte	Číslo teploměru (první má číslo 0x01). 0x00 znamená dotaz na všechny teploměry. Parametr může být v dotazu maximálně dvanáctkrát.

status	1 byte	Bitově orientovaný byte ve tvaru: `V00000HL` (V je MSb)
		• V: Platnost:
		  ○ 1 = Teplota je platná
		  ○ 0 = Teplota je neplatná!
		• H: 1 = Teplota nad horní mezí
		• L: 1 = Teplota pod dolní mezí

int	2 byte	Naměřená teplota. Platnost údaje určuje *status*! Hodnota MSB:LSB ve formátu signed int. Teplota se z hodnoty získá vydělením deseti.
float	4 byte	Naměřená teplota jako údaj s plovoucí řádovou čárkou dle IEEE 754. Platnost údaje určuje *status*!
string	10 byte	Naměřená teplota jako řetězec. Například „24,6“. Platnost údaje určuje *status*!

Čtení

Při chybě senzoru je v parametru *status* nastaven bit V=0 a jako teplota je nastaveno číslo -9999 po cca deseti sekundách trvání chyby.

Struktura:	→ **0x58**, (id) ← (id, status, int, float, string)
Příklad:	→ 2A 61 00 06 B1 02 **58 00** 63 0D ← 2A 61 00 17 B1 02 **00** **01** **80** **01 10** **41 DA 00 00** **20 20 20 20 20 20 32 37 2E 32** 74 0D • 0x01: První teploměr • 0x80: Naměřená teplota je platná, a je v mezích. • 0x0110, tj. dekadicky 272: Teplota 27,2° • 0x41DA0000: 2.725e+1 • 0x20,0x20,0x20,0x20,0x20,0x20,0x32,0x37,0x2E,0x32, tj. „ 27,2“

Teplotní jednotka – 0x1C/0x1D

Hlavní teplotní jednotka, se kterou zařízení pracuje.

Parametry

id	1 byte	Vždy 0x00 v dotazu a 0x01 v odpovědi.
unit	1 byte	Kód jednotky: • 0x00: stupně Celsia • 0x01: stupně Fahrenheita • 0x02: Kelvin

Zápis

Struktura:	→ **0x1C**, id, unit
Příklad:	→ 2A 61 00 07 B1 02 **1C 00 01** 9D 0D • 0x01: Nastavení stupňů Fahrenheita

Čtení

Struktura:	→ **0x1D** ← id, unit
Příklad:	→ 2A 61 00 05 B1 02 **1D** 9F 0D ← 2A 61 00 07 B1 02 **00 01 01** B8 0D • 0x01: Nastaveny stupně Fahrenheita

Hlídání teplotních mezí – 0x13/0x14

Pokud naměřená teplota opustí dvojici nastavených teplotních mezí odešle zařízení automatickou zprávu. Při komunikaci přes RS485 lze tuto funkci použít jen pokud je na lince jen jedno zařízení!

Parametry

Parametry, které mají před délkou uvedenou konstantu, nemusejí být uvedené v instrukci všechny najednou. V datech je nejdřív jeden byte s konstantou a pak následuje parametr. Podle konstanty zařízení jednoznačně identifikuje, o jaký parametr jde.

tid	1 byte	Číslo teploměru z intervalu 0x01 až 0x08.
enable	0x01 + 1 byte	Hlídání teploty zapnuto (0x01) nebo vypnuto (0x00).
highInt	0x02 + 2 byte	Horní mez teploty. Hodnota MSB:LSB ve formátu signed int. Teplota se z hodnoty získá vydělením deseti.
lowInt	0x03 + 2 byte	Dolní mez teploty. Hodnota MSB:LSB ve formátu signed int. Teplota se z hodnoty získá vydělením deseti.
period	0x04 + 2 byte	Pokud je teplota mimo meze, automatická zpráva se bude odesílat tak často, jak je nastaveno zde. Údaj je v sekundách.
highStr	0x05 + X byte	Horní mez teploty jako řetězec. Například „24,6“.
lowStr	0x06 + X byte	Dolní mez teploty jako řetězec. Například „-21,7“.

status	1 byte	Bitově orientovaný byte ve tvaru: `V00000HL` (V je MSb)
		• V: Platnost:
		  ○ 1 = Teplota je platná
		  ○ 0 = Teplota je neplatná!
		• H: 1 = Teplota nad horní mezí
		• L: 1 = Teplota pod dolní mezí

Zápis

Struktura:	→ **0x13**, tid, enable, highInt, lowInt, period, highStr, lowStr
Příklad:	→ 2A 61 00 11 31 02 **13 01 01 01 02 01 36 03 00 FA 04 00 01** DF 0D • 0x01: První teploměr. • 0x01, 0x01: Hlídání teploty zapnuto. • 0x02, 0x0136: Horní mez 310, tj. 31,0° • 0x03, 0x00FA: Dolní mez 250, tj. 25,0°

	• 0x04, 0x0001: Perioda 1 sec.

Čtení

Struktura:	→ **0x14**, tid ← tid, enable, highInt, lowInt, period, highStr, lowStr
Příklad:	→ 2A 61 00 06 31 02 **14** **01** 26 0D • 0x01: ptáme se na 1. teploměr. ← 2A 61 00 27 31 02 **00** **01** **01 01** **02 01 36** **03 00 FA** **04 00 01** **05 20 20 20 20 20 20 33 31 2E 30** **06 20 20 20 20 20 20 32 35 2E 30** CA 0D • 0x01: První teploměr. • 0x01, 0x01: Hlídání teploty zapnuto. • 0x02, 0x0136: Horní mez 310, tj. 31,0° • 0x03, 0x00FA: Dolní mez 250, tj. 25,0° • 0x04, 0x0001: Perioda 1 sec.

Automatická zpráva

Struktura:	← (0x0158, 0x02, tid, 0x03, status, 0x04, int, float, string)
Příklad:	← 2A 61 00 1C 31 C0 **0F** **01 58** **02** **01** **03** **82** **04** **01 39** **41 FB 00 00** **20 20 20 20 20 20 33 31 2E 33** 78 0D • 0x0158: Konstanta. • 0x0F je ACK značící automatickou zprávu. • 0x01: První teploměr. • 0x82: Platná hodnota, teplota nad horní mezí. • 0x0139: Hodnota 313 (signed int) znamená 31,3° • 0x41FB0000: Formát float (IEEE 754). • Ostatní byty jsou ASCII řetězec s teplotou.

Termostat – 0x1A/0x1B

Každý z výstupů umí automaticky reagovat na teplotu naměřenou teplotním senzorem. Podrobný popis funkce termostatu je v Dodatku 1 na straně 43 tohoto dokumentu.

Parametry

out	1 byte	Číslo výstupu. První má číslo 0x01. Hodnota 0x00 je neplatná.

flags	1 byte	Bitově orientovaný byte ve tvaru: FSSKSTTT (F je MSb)
		• F: Výstup *out* řízen termostatem:
		  ○ 1 = ano
		  ○ 0 = ne
		• SSxS: Akce výstupu, která se má při nastavené teplotě provést
		  ○ 00x0 = sepnout
		  ○ 01x0 = rozepnout
		  ○ 10x0 = sepnout na dobu *time* („kladný pulz“)
		  ○ 11x0 = rozepnout na dobu *time* („záporný pulz“)
		  ○ 00x1 = sepnout v mezích *tempx* a *tempy*
		  ○ 01x1 = sepnout mimo meze *tempx* a *tempy*
		• K: Kritická teplotní tendence (týká se jen SSxS=10x0 a SSxS=11x0):
		  ○ 0 = vzestup teploty

		○ 1 = pokles teploty • TTT: Číslo teplotního senzoru (většinou 1).
tempx	2 byte	Vyšší teplota. Hodnota MSB:LSB ve formátu signed int. Teplota se z hodnoty získá vydělením deseti.
tempy	2 byte	Nižší teplota. Hodnota MSB:LSB ve formátu signed int. Teplota se z hodnoty získá vydělením deseti.
time	1 byte	Délka změny stavu výstupu ve vteřinách (týká se jen SSxS=10 a SSxS=11x0).
error	1 byte	Jak reagovat na chybu teploměru (přerušený/odpojený senzor): • 0: žádná reakce • 1: rozepnout výstup • 2: sepnout výstup

Zápis

V dotazu může být sekvence parametrů v závorce () uvedena maximálně dvanáctkrát za sebou. Termostat tedy jde nastavit až u dvanácti výstupů najednou.

Struktura:	→ **0x1A**, (out, flags, tempx, tempy, time, err) [15]
Příklad:	→ 2A 61 00 0D 31 02 **1A** **01** **81** **01 0E** **01 0E** **05** **00** 75 0D • 0x01: První výstup. • 0x81 → binárně 10000001: Termostat zapnutý (F=1), Akce sepnutí (SSxS=00x0), Teploměr 1 (TTT=1). • 0x010E → dekadicky 270: Vyšší teplota je 27,0 °C • 0x010E → dekadicky 270: Nižší teplota je 27,0 °C • 0x05: čas 5 sec (vzhledem k SSxS=00x0 nemá význam) • 0x00: při chybě teploměru žádná reakce

Čtení

Parametr *out* v dotazu nemusí být vůbec (v odpovědi budou všechny výstupy) nebo až dvanáctkrát. Tomu bude odpovídat počet sekvencí () v odpovědi.

Struktura:	→ **0x1B**, out ← (out, flags, tempx, tempy, time, err)
Příklad:	→ 2A 61 00 05 31 02 **1B** 21 0D • Dotaz na všechny výstupy. ← 2A 61 00 15 31 02 **00** **01** **81** **01 0E 01 0E 05 00** **02** **00** **27 0F D8 F1 00 00** 86 0D • 0x01: První výstup. ○ 0x81 → binárně 10000001: Termostat zapnutý (F=1), Akce sepnutí (SSxS=00x0), Teploměr 1 (TTT=1). • 0x02: Druhý výstup. ○ 0x00 → binárně 00000000: Termostat vypnutý (F=0).

Konfigurace komunikační linky a nastavení adresy

Povolení konfigurace – 0xE4

Povoluje provedení některých klíčových instrukcí. U těchto instrukcí je potřeba povolení konfigurace výslovně uvedena. Povolní konfigurace musí každé z těchto instrukcí bezprostředně předcházet. Povolení konfigurace platí vždy jen pro jednu následující instrukci. Povolení konfigurace není možné použít s univerzální adresou.

Zápis

Struktura:	→ **0xE4**
Příklad:	→ 2A 61 00 05 31 02 **E4** 58 0D

Zápis 66:

→ `*B1E↵`

← `*B10↵`

Komunikační parametry – 0xE0/0xF0

Nastavení adresy a komunikační rychlosti.

Parametry

addr	1 byte	Hodnota 0x00 – 0xFD. Pokud pro komunikaci se zařízením používáte i ascii formát 66, je nutné použít jen takové adresy, které je možné vyjádřit jako zobrazitelný ASCII znak (tzn. znaky s kódy 0x20 až 0x7E).
baud	1 byte	Komunikační rychlost [Bd]: • 1200 ..... 0x03 • 2400 ..... 0x04 • 4800 ..... 0x05 • 9600 ..... 0x06 • 19200 ..... 0x07 • 38400 ..... 0x08 • 57600 ..... 0x09 • 115200..... 0x0A • 230400..... 0x0B **Quida ETH a Quida USB** mají neměnnou rychlost 115200 Bd. Jako parametr *baud* musí být u těchto Quid zadán vždy kód pro rychlost 115200 Bd, jinak modul odpoví ACK 0x03 (neplatná data). **Quida RS** mají výchozí rychlost 9600 Bd.

Zápis

Po změně parametrů dojde k restartu zařízení a tím i k vynulování čítačů a statusu! K zápisu není možné použít univerzální adresu 0xFE. Pokud neznáte adresu zařízení, můžete nastavit novou adresu instrukcí Nastavení adresy sériovým číslem (str. 34). Zápisu musí předcházet Povolení konfigurace – 0xE4.

Struktura:	→ **0xE0**, addr, baud
Příklad:	→ 2A 61 00 07 01 02 **E0** **<u>02</u>** **<u>0A</u>** 7E 0D • 0x02: nová adresa • 0x0A: rychlost ← 2A 61 00 05 01 02 **00** 6C 0D • Nová adresa a komunikační rychlost se nastaví až po odeslání odpovědi.

Čtení

Čtení komunikačních parametrů má smysl v případě, že není známa adresa zařízení. Čtení se provede s univerzální adresou 0xFE. Pokud není známa ani komunikační rychlost je možné využít Konfigurační propojky anebo vyzkoušet všechny komunikační rychlosti. K nalezení zařízení s neznámými komunikačními parametry je možné použít Modbus Configurator.

Struktura:	→ **0xF0** ← addr, baud
Příklad:	→ 2A 61 00 05 FE 02 **F0** 7F 0D • Dotaz s univerzální adresou 0xFE. ← 2A 61 00 07 04 02 **00** **04** **06** 5D 0D • 0x04: adresa 0x04 • 0x06: rychlost 9600 Bd

Zápis 66:

Ve formátu 66 se adresa a rychlost nastavuje dvěma oddělenými instrukcemi:

→ `*B1AS4↵`

- 4: Nastavení adresy 4.

← `*B10↵`

- Adresa bude změněna až po odeslání odpovědi.

→ `*B1SS7↵`

- 7: Nastavení komunikační rychlosti 19200 Bd.

← `*B10↵`

- Rychlost bude změněna až po odeslání odpovědi.

Čtení 66:

→ `*B1CP↵`

← `*B10B7↵`

- B: Adresa B.
- 7: Komunikační rychlost 19200 Bd.

Nastavení adresy sériovým číslem – 0xEB

Instrukce umožňuje nastavit adresu podle unikátního sériového čísla zařízení. Tato instrukce je praktická v případě, že nadřazený systém nebo obsluha ztratí adresu zařízení, které je na stejné komunikační lince s dalšími zařízeními.

Parametry

newaddr	1 byte	Nová adresa z rozsahu 0x00 – 0xFD.

sn	2+2 byte	Sériové číslo složené z výrobního a produktového čísla. S/N je uvedené na štítku na zařízení a jde přečíst také pomocí instrukce Výrobní údaj.  *obr. 2 - sériové číslo zařízení*

Zápis

Struktura:	→ **0x24**, newaddr, sn
Příklad:	→ 2A 61 00 0A FE 02 **EB** **<u>32</u>** **<u>01 3B</u>** **<u>04 F9</u>** 14 0D • 0x32: nová adresa • 0x013B: produkt číslo 315 • 0x04F9: kus číslo 1273 ← 2A 61 00 05 <u>32</u> 02 **00** 3B 0D • Odpověď již s novou adresou 0x32.

Ostatní

Jméno a verze – 0xF3

Podle typu dotazu vrací počty vstupů, výstupů a teploměrů, jméno přístroje, verzi vnitřního software a seznam možných formátů komunikace. Umí také zjistit adresu pomocí sériového čísla.

Parametry

getio	1 byte	Je-li zde uvedeno 0x01, odpověď není standardní textová, ale instrukce v odpovědi vrací *ios* se strojově čitelným počtem I/O.
ios	3 byte	Počty I/O: • 1. byte: Počet vstupů • 2. byte: Počet výstupů • 3. byte: Počet teploměrů
sn	2+2 byte	Sériové číslo složené z výrobního a produktového čísla. S/N je uvedené na štítku na zařízení (viz str. 35). Je-li na jedné komunikační lince více zařízení se stejnou adresou, lze takto pomocí univerzální adresy 0xFE a sériového čísla oslovit jedno konkrétní zařízení.
string	x byte	Řetězec s identifikací zařízení ve tvaru: *[device-type]; v[device-number].[hw-version].[sw-version];*

Čtení

Struktura:	→ **0xF3**, [getio] | [sn] [15] ← string | ios
Příklad 1: *Identifikační řetězec*	→ 2A 61 00 05 FE 02 **F3** 7C 0D • Dotaz bez parametru. ← 2A 61 00 2B 31 02 **00** **51 75 69 64 6F 20 55 53 42 20 34 2F 34 3B 20 76 30 32 35 33 2E 30 34 2E 34 38 3B 20 66 36 36 20 39 37 3B 20 74 31** CF 0D • Řetězec s identifikací zařízení: *Quido USB 4/4; v0253.04.48; f66 97; t1*
Příklad 2: *Počty I/O*	→ 2A 61 00 06 FE 02 **F3** **01** 7A 0D • 0x01: Dotaz na počty vstupů, výstupů a teploměrů. ← 2A 61 00 08 31 02 **00** **04 04 01** 30 0D • 0x040401: Dotaz na počty vstupů, výstupů a teploměrů. ○ 0x04: Počet vstupů. ○ 0x04: Počet výstupů. ○ 0x01: Počet teploměrů.
Příklad 3: *Oslovení pomocí S/N*	→ 2A 61 00 09 FE 02 **F3** **00 FD 08 8F** E4 0D • Pomocí univerzální adresy 0xFE, pošleme dotaz zařízení se sériovým číslem 0253/2191 (0x00FD/0x088F). ← 2A 61 00 2B 31 02 **00** **51 75 69 64 6F 20 55 53 42 20 34 2F 34 3B 20 76 30 32 35 33 2E 30 34 2E 34 38 3B 20 66 36 36 20 39 37 3B 20 74 31** CF 0D • Řetězec s identifikací zařízení: *Quido USB 4/4; v0253.04.48; f66 97; t1*

Čtení 66:

```
→ *B1?↵

← *B10Quido ETH 4/4; v0254.02.07; f66 97; t1↵
```

Výrobní údaje – 0xFA

Instrukce přečte výrobní údaje ze zařízení.

Parametry

type	2 byte	Typové číslo výrobku. Z řetězce 0254/0001 jde o číslo 254.
item	2 byte	Číslo kusu. Z řetězce 0254/0001 jde o číslo 1.
factory	4 byte	Výrobní údaje.

Čtení

Struktura:	→ **0xFA** ← type, item, factory
Příklad:	→ 2A 61 00 05 FE 02 **FA** 75 0D ← 2A 61 00 0D 35 02 **00** **00 C7** **00 65** **20 05 09 23** B3 0D • 0x00C7: Typ 199. • 0x0065: Kus 101. • 0x20050923: Výrobní údaje.

Uživatelská data – 0xE2/0xF2

Prostor pro uživatelská data je paměť, do které si může uživatel uložit libovolná data, která si bude zařízení pamatovat i po vypnutí napájení nebo resetu. Tento prostor je vhodný například pro pojmenování měřicího místa.

Parametry

offset	1 byte	Adresa paměti, kam se mají data uložit. Číslo z rozsahu 0x00 – 0x0F. V případě zápisu od offsetu např. 0x0C, lze zapsat max. 4 byte.
data	1-16 byte	Uživatelská data.

Zápis

Struktura:	→ **0xE2**, offset, data
Příklad:	→ 2A 61 00 10 FE 02 **E2** **00** **53 74 6F 72 61 67 65 20 34 32** 27 0D • 0x00: Offset 0 – zápis od začátku. • Text *Storage 42* (10 znaků).

Čtení

Struktura:	→ **0xF2** ← data
Příklad:	→ 2A 61 00 05 FE 02 **F2** 7D 0D ← 2A 61 00 16 31 02 **00** **53 74 6F 72 61 67 65 20 34 32 20 20 20 20 20 20 20** F0 0D • Text „Storage 42 “ (16 znaků)

Zápis 66:

→ `*B1DW0Storage 42↵`

- 0: Pozice v paměti, na kterou se bude zapisovat. Znak z intervalu 0-9 nebo A-F.
- Storage 42: 1 až 16 znaků.

← `*B10↵`

Čtení 66:

→ *B1DR↵

← *B10Storage 42↵

Názvy vstupů – 0x2B/0x3B

Umožňuje pro každý vstup uložit jedinečný řetězec znaků. Tato funkce se hodí pro pojmenování vstupů.

Toto paměťové místo využívá ovládací software, který je dodáván zdarma k modulům Quido. Také je využit v Ethernetových verzích modulů Quido pro uložení názvů vstupů a výstupů. Z těchto důvodů nedoporučujeme manipulovat s tímto paměťovým místem při použití s naším standardním softwarem nebo při použití standardního webového rozhraní v Ethernetových modulech Quido.

Pokud není na zařízení žádný vstup, odpovídá se ACK 0x02 (neplatná instrukce).

Parametry

input	1 byte	Číslo vstupu. První vstup má číslo 0x01.
data	21 byte	Libovolná uživatelská data.

Zápis

Struktura:	→ **0x2B**, input, data
Příklad:	→ 2A 61 00 1B 31 02 **2B** **01** **30 4B 6F 74 65 6C 6E 61 00 00 00 00 00 00 00 00 00 00 00 00 00** FC 0D • Uložení názvu "0Kotelna" ke vstupu 1 (0x01). (Nevyužité bajty jsou vyplněny nulami.)

Čtení

Struktura:	→ **0x3B**, input ← data
Příklad:	→ 2A 61 00 06 31 02 **3B** **01** FF 0D • 0x01: ptáme se na 1. vstup. ← 2A 61 00 1A 31 02 **00** **30 4B 6F 74 65 6C 6E 61 00 00 00 00 00 00 00 00 00 00 00 00 00** 29 0D • V datech je uložen text "0Kotelna". (Nevyužité bajty jsou vyplněny nulami.)

Názvy výstupů – 0x2A/0x3A

Umožňuje pro každý výstup uložit jedinečný řetězec znaků. Tato funkce se hodí pro pojmenování výstupů.

Toto paměťové místo využívá ovládací software, který je dodáván zdarma k modulům Quido. Také je využit v Ethernetových verzích modulů Quido pro uložení názvů vstupů a výstupů. Z těchto důvodů nedoporučujeme manipulovat s tímto paměťovým místem při použití s naším standardním softwarem nebo při použití standardního webového rozhraní v Ethernetových modulech Quido.

Pokud není na zařízení žádný výstup, odpovídá se ACK 0x02 (neplatná instrukce).

Parametry

output	1 byte	Číslo výstupu. První výstup má číslo 0x01.
data	21 byte	Libovolná uživatelská data.

Zápis

Struktura:	→ **0x2A**, output, data
Příklad:	→ 2A 61 00 1B 31 02 **2A** **04** **30 53 69 72 65 6E 61 00 00 00 00 00 00 00 00 00 00 00 00 00 00** 66 0D • Uložení názvu "0Sirena " k výstupu 4 (0x04). (Nevyužité bajty jsou vyplněny nulami.)

Čtení

Struktura:	→ **0x3A**, output ← data
Příklad:	→ 2A 61 00 06 31 02 **3A** **04** FD 0D • 0x04: ptáme se na 4. výstup. ← 2A 61 00 1A 31 02 **00** **30 53 69 72 65 6E 61 00 00 00 00 00 00 00 00 00 00 00 00 00 00** 95 0D • V datech je uložen text "0Sirena". (Nevyužité bajty jsou vyplněny nulami.)

Status a Run time – 0xE1/0xF1

Status je jeden byte paměti určený pro uživatelské označení stavu zařízení. Po zapnutí zařízení a po resetu (i softwarovém) je automaticky nastaven na 0x00.

Při čtení lze kromě statusu přečíst také dobu od zapnutí zařízení v sekundách.

Parametry

status	1 byte	Po zapnutí nebo resetu je nastavena automaticky hodnota 0x00.
runtime	4 byte	Doba od zapnutí nebo resetu zařízení v sekundách.

Zápis

Struktura:	→ **0xE1**, status
Příklad:	→ 2A 61 00 06 01 02 **E1** **12** 78 0D • Nastavení statusu 0x12.

Čtení

0x31 je nepovinný parametr, který označuje požadavek na vyčtení *runtime*.

Struktura:	→ **0xF1**, [0x31] [15] ← status
Příklad 1: *Jen status*	→ 2A 61 00 05 01 02 **F1** 7B 0D ← 2A 61 00 06 01 02 **00** **12** 59 0D • Ve statusu je číslo 0x12.
Příklad 2: *Status + runtime*	→ 2A 61 00 06 FE 02 **F1** **31** 4C 0D ← 2A 61 00 0A 31 02 **00** **12** **00 00 00 DD** 48 0D • 0x12: Ve statusu je dekadické číslo 18. • 0x000000DD: Od resetu uplynulo 221 sekund.

Zápis 66:

→ `*B1SWA↵`

- A: znak z intervalu „mezera“ až „~“ (32 – 126)

← `*B10↵`

Čtení 66:

→ `*B1SR↵`

← `*B10A`

Chyby komunikace – 0xF4

Instrukce vrací počet chyb komunikace, které se vyskytly od zapnutí přístroje, nebo od posledního čtení chyb komunikace.

Parametry

errors	1 byte	Počet chyb komunikace, které se vyskytly od zapnutí přístroje, nebo od posledního čtení. Za chyby komunikace jsou považovány následující události: • Je očekáván prefix, ale přijde jiný byte. • Nesouhlasí kontrolní součet SUMA. • Zpráva není kompletní – tj. vyprší timeout komunikace.

Čtení

Struktura:	→ **0xF4** ← errors
Příklad:	→ 2A 61 00 05 01 02 **F4** 78 0D ← 2A 61 00 06 01 02 **00 05** 66 0D • 0x05: Pět chyb.

Kontrolní součet – 0xEE/0xFE

Kontrolní součet (angl. checksum) je ochranou proti poškození dat při přenosu. Tato funkce umožňuje kontrolu checksumu vypnout je určená pouze pro ladění aplikací! Při ručním zadávání instrukcí prostřednictvím terminálu tak není nutné správně zadávat kontrolní součet (předposlední byte). Kontrola je z výroby zapnuta.

Parametry

state	1 byte	• 0x01: Zapnuto. • 0x00: Vypnuto.

Zápis

Struktura:	→ **0xEE**, state
Příklad:	→ 2A 61 00 06 01 02 **EE 01** 7C 0D • 0x01: zapnutí kontroly checksumu

Čtení

Struktura:	→ **0xFE** ← state

Příklad:	→ 2A 61 00 05 01 02 **FE** 6E 0D ← 2A 61 00 06 01 02 **00 01** 6A 0D • 0x01: kontrola checksumu je zapnuta

Timeout komunikace – 0xE5/0xF5

Umožňuje nastavit timeout komunikace. Timeout je doba, která se měří po přijetí každého byte, a po které je komunikace považována za přerušenou (viz Chyby komunikace – 0xF4).

Parametry

ms	1 byte	Čas v desítkách milisekund, tj. 10 – 2550 ms. Výchozí čas je 1 sec.

Zápis

Struktura:	→ **0xE5**, ms
Příklad:	→ 2A 61 00 06 B1 02 **E5 20** B6 0D • 0x20: nastavení timeoutu 32 ms

Čtení

Struktura:	→ **0xF5** ← ms
Příklad:	→ 2A 61 00 05 B1 02 **F5** C7 0D ← 2A 61 00 06 B1 02 **00 20** 9B 0D • 0x20: timeout je 20 ms

Reset – 0xE3

Provede reset zařízení. Zařízení se dostane do shodného stavu jako po zapnutí napájení. Reset se provede až po odeslání odpovědi.

Zápis

Struktura:	→ **0xE3**
Příklad:	→ 2A 61 00 05 01 02 **E3** 89 0D

Zápis 66:

→ `*B1RE↵`

← `*B10↵`

Výchozí nastavení – 0x8F

Nastaví všechny parametry zařízení do výchozího nastavení. Instrukci musí předcházet instrukce Povolení konfigurace popsaná na straně 33.

Zápis

Struktura:	→ **0x8F**
Příklad:	→ 2A 61 00 05 B1 02 **8F** 2D 0D

Přepnutí komunikačního protokolu – 0xED

Instrukci musí předcházet Povolení konfigurace popsané na straně 33. K přepnutí protokolu lze použít například program Modbus Configurator.

Parametry

protocol	1 byte	Identifikační číslo protokolu: • 0x01: Spinel, formát 97 (binární) a 66 (ascii) • 0x02: Modbus RTU (jen pro Quida RS a Quida USB) • 0x0A: Spinel, pouze formát 97 (binární)

Zápis

Struktura:	→ **0x2A**, protocol
Příklad:	→ 2A 61 00 06 31 02 **ED 02** 4C 0D • Přepnutí do protokolu Modbus RTU.

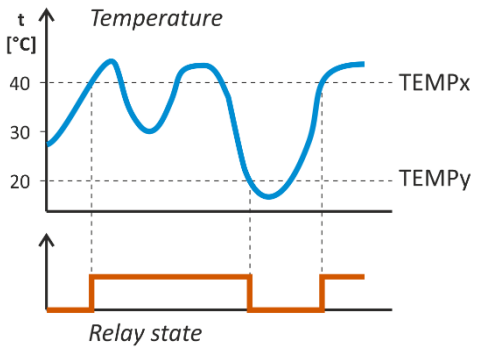
DODATEK 1: TERMOSTAT

V tomto dodatku jsou popsány režimy termostatu v modulech Quido s připojeným teplotním senzorem. Označení teplot a parametru *flags* a dalších proměnných na obrázcích a v textu je shodné s označením parametrů instrukce pro nastavení termostatu.

Režim 1

Výstup sepne při překročení teploty TEMPx a rozepne při poklesu pod teplotu TEMPy.[19]

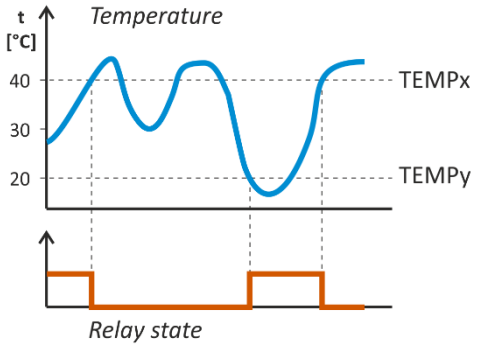
- Stav výstupu v klidu: OFF
- Bity v parametru *flags*:
  - SSxS: 00x0 – sepnout výstup



Režim 2

Výstup rozepne při překročení teploty TEMPx a sepne při poklesu pod teplotu TEMPy.[19]

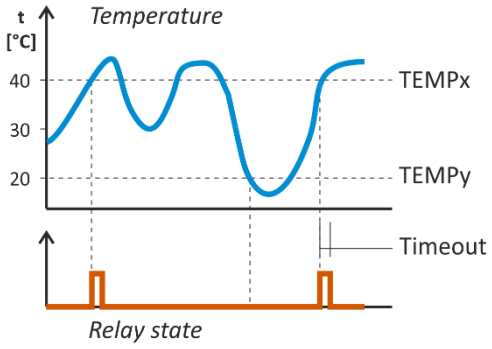
- Stav výstupu v klidu: ON
- Bity v parametru *flags*:
  - SSxS: 01x0 – rozepnout výstup



Režim 3

Výstup sepne na nastavenou dobu při překročení teploty TEMPx. Znovu může sepnout, až pokud teplota klesne pod TEMPy a poté znovu vzroste na TEMPx.[19]

- Stav výstupu v klidu: OFF
- Bity v parametru *flags*:
  - SSxS: 10x0 – sepnout na nastavenou dobu
  - K: 0 – vzestup teploty

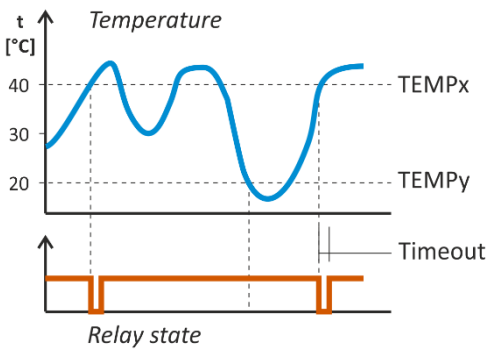


[19] Tím je zavedena v řízení teploty tzv. hystereze. Obě teploty je možné nastavit na stejnou hodnotu a tím hysterezi zrušit.

Režim 4

Výstup rozepne na nastavenou dobu při překročení teploty TEMPx. Znovu rozepne, až pokud teplota klesne pod TEMPy a poté znovu vzroste na TEMPx.[19]

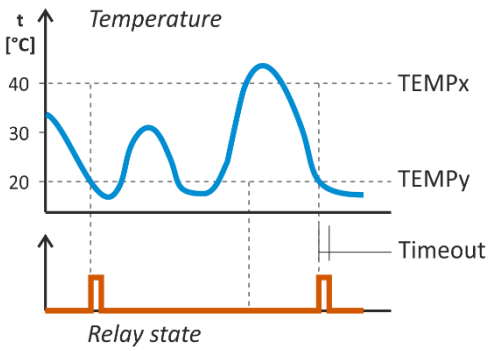
- Stav výstupu v klidu: ON
- Bity v parametru *flags*:
  - SSKS: 11x0 – rozepnout na nastavenou dobu
  - K: 0 – vzestup teploty



Režim 5

Výstup sepne na nastavenou dobu při poklesu teploty pod TEMPy. Znovu může sepnout, až pokud teplota stoupne nad TEMPx a poté znovu klesne pod TEMPy.[19]

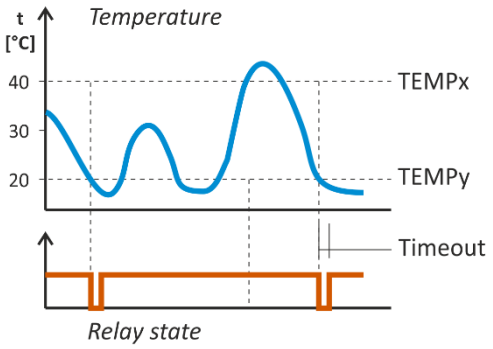
- Stav výstupu v klidu: OFF
- Bity v parametru *flags*:
  - SSxS: 10x0 – sepnout na nastavenou dobu
  - K: 1 – pokles teploty



Režim 6

Výstup rozepne na nastavenou dobu při poklesu pod teplotu TEMPy. Znovu může rozepnout, až pokud teplota stoupne nad TEMPx a poté znovu klesne na TEMPy.[19]

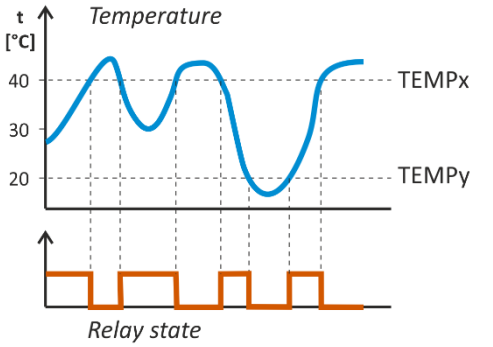
- Stav výstupu v klidu: ON
- Bity v parametru *flags*:
  - SSxS: 11x0 – rozepnout na nastavenou dobu
  - K: 1 – pokles teploty



Režim 7

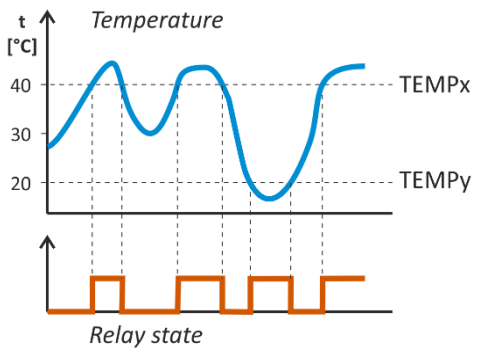
Výstup je sepnutý, pokud je teplota mezi TEMPx a TEMPy.

- Bity v parametru *flags*:
  - SSxS: 00x1 – sepnout v mezích



Režim 8

Výstup je sepnutý, pokud je teplota mimo meze TEMPx a TEMPy.



- Bity v parametru *flags*:
  - SSxS: 01x1 – sepnout mimo meze

Papouch s.r.o.

Přenosy dat v průmyslu, převodníky linek a protokolů, RS232, RS485, RS422, USB, Ethernet, LTE, WiFi, měřicí moduly, inteligentní teplotní čidla, I/O moduly, zakázkový vývoj a výroba.

Adresa:

Strašnická 3164/1a
102 00 Praha 10

Telefon:

+420 267 314 268

Web:

papouch.com

Mail:

papouch@papouch.com