

Testování

Datum: 27.2.2024, commit 27f5877e3eee4d5ca314fab6c9243825900d0cf5

Prostředí: Windows 10, Qt6, mingw_64

Testované komponenty:

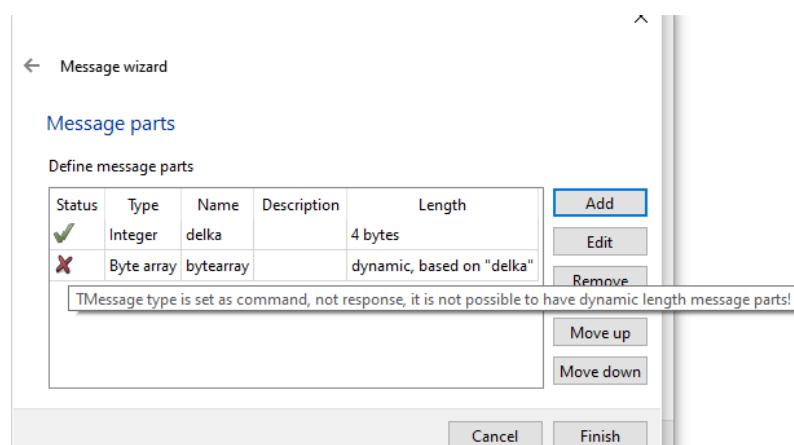
- Protocol, message, message part
- IO Device (FilePlugin)

-
- V protocol manager dvojklik pro úpravu protokolu funguje. Mohl by se přidat ten samý dvojklik i na protokoly v záložce Project?
 - Když otevřu okno message part details, tak není správně vykreslený formulář (co se týče skrytých polí apod.). Hned jak pozměním libovolné pole (na obrázku jsem dal znak do pole Name) se vše správně aktualizuje:

Message part details	
Set message part name, type and length	
Name:	<input type="text"/>
Description:	<input type="text"/>
Data type:	String
Is payload:	<input type="checkbox"/>
Value:	<input type="text"/>
Interpret as:	<input type="radio"/> Hex <input checked="" type="radio"/> ASCII
Interpreted value:	<input type="text"/>
Has static length:	<input type="checkbox"/>
Length:	<input type="text"/> 0
Length determined by:	<input type="text"/>
Endianness:	Little endian

Message part details	
Set message part name, type and length	
Name:	a
Description:	<input type="text"/>
Data type:	String
Is payload:	<input type="checkbox"/>
Value:	<input type="text"/>
Interpret as:	<input type="radio"/> Hex <input checked="" type="radio"/> ASCII
Interpreted value:	Error interpreting value. Check type and length.
Has static length:	<input checked="" type="checkbox"/>
Length:	<input type="text"/> 0
Endianness:	Little endian

- Pokud vytvářím command není možné použít dynamic length. Pokud je to tak dobré, pak bych rovnou zakázal možnost vybrat dynamic length při editaci. Chtělo by to rozhodnout, zda command může mít dynamickou délku.



- Když přijmu unsigned long long int jako parametr délky message part PT, tak se číslo špatně vypíše v přehledu komunikace. Každopádně číslo se interpretuje správně.

Received:

receive PT: (length: 216172782113783808, PT: 0x544595)

má být 3

- Když přijmu byte array proměnlivé délky, tak se vypíše v opačném pořadí. V souboru je pořadí správné, stejně tak i přijmutí pomocí raw data.

Sent:

send u long long; (cislo; 10)

Sent:

0x01020304050607080910

(seek to 0)

Received:

receive PT: (length: 720575940379279360, PT: 0x10090807060504030201)

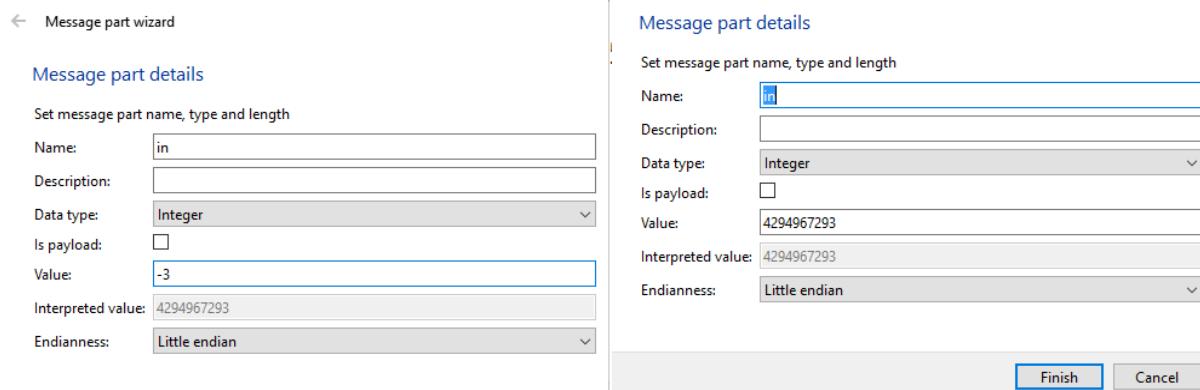
(seek to 0)

Received:

0x0a000000000000001020304050607080910

- Pokud protokol nemá command, v IODevice mám možnost vybrat zprávu z prázdné nabídky. Nešlo by v takovém případě vypsat „Protokol neobsahuje žádné příkazy“? At uživatel hned ví, co se děje.
 - Pokud mám proměnlivou délku zprávy, tak ukazatel s parametrem na délku je udělán pomocí indexu do message parts. Pokud změním pořadí částí zprávy, tak se délka odkazuje pořád na původní index -- je potřeba zvážit, zda má smysl tohle opravovat. Chybová hlášení fungují dobře. Minimálně bych aktualizoval chybová hlášení při změně pořadí.
 - Zadávání konstantní hodnoty v message part details. U jednobytových struktur při hex reprezentaci nelze zadat jednociferné číslo. Pro byte a character nejde zadat „0“, ale jen „00“. U více bytových hodnot tohle jde v pořádku. Problém nastal, když jsem chtěl zadat character 0 (jako „\n“) a dostával jsem hlášku „error interpreting value“ a chvíli trvalo, než jsem přišel na to, že musím zadat „00“.

- Při zadávání konstantní zprávy se znaménkovým číslem se záporná čísla vypisují jako neznaménková čísla. Na obrázku je vlevo vytváření části zprávy a vpravo později při editaci. U short a long long int se číslo vypisuje správně.



- Při odeslání/přijmutí int se špatně vypíše záporné číslo:

Sent:
 sent signed: (short positive: 10, short negative: -10,
 int positive: 20, int negative: 4294967276, ll positive:
 30, ll negative: -30)
Received:
 receive signed: (short positive: 10, short negative:
 -10, int positive: 20, int negative: 4294967276, ll
 positive: 30, ll negative: -30)

Send data

Protocol	signed things
Message	sent signed
short positive	10
short negative	-10
int positive	20
int negative	-20
ll positive	30
ll negative	-30
Send	

- U string, byte array, character, unsigned char a byte by nemělo záležet na endianitě. Jsou buď jednobyтовé nebo sekvence bytů. V těchto případech bych nedával endianitu na výběr (message part details).

- Při odesílání big-endian hodnoty program spadne (testováno s `short` a `int`). Vytvořil jsem protokol, co obsahoval `command` s jedním shortem s big-endian (nezáleží, zda je to konstanta nebo payload). Při odeslání do souboru program okamžitě spadne. U `response` endianita funguje dobře.
- Pokud vyberu `unsigned char` jako datový typ zprávy, pak v přehledu je ukázán jako `byte`. Pokud je to to samé, pak není třeba mít oboje (`byte` a `unsigned char`) v nabídce.

[Message parts](#)

[Define message parts](#)

Status	Type	Name	Description	Length
✓	Character	char		1 bytes
✓	Byte	uchar		1 bytes
✓	Byte	byte		1 bytes

[Message part details](#)

Set message part name, type and length

uchar

uchar

Unsigned char

01

Hex **ASCII**

01

Little endian

- Autoreceive neinterpretuje příchozí zprávy, pokud jsme přijmul víc dat. Například pošlu na device dva příkazy, který kdyby přišly zvlášť, tak je správně dekóduje. Pokud přijmou najednou, tak ne. (nemyslím si, že je to nutně špatně, ale uvádím to pro úplnost)