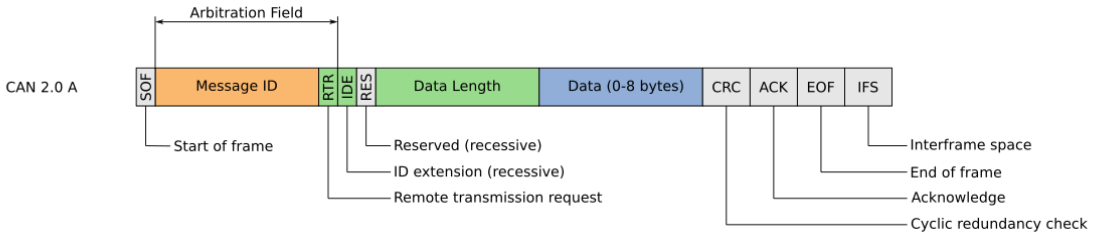
Organizace dat přijímaných/vysílaných z/do CANu

Úvodem považujeme za vhodné uživateli knihovny CANUSB32.DLL přiblížit, jak jsou v rámci CANu a standardu aplikační vrstvy CANOpen organizována data, ke kterým přes naše API prostřednictvím virtuálních proměnných či kanálů přistupuje.

Na úrovni síťové vrstvy můžeme CAN považovat za síť s multicastovou komunikací. Data jsou organizována do zpráv, které z uživatelského hlediska dle standardu CAN 2.0 A obsahují dvě základní části: 11bitové ID zprávy a 8bytové datové pole. Všechna zařízení mohou (ale nemusí) přijímat všechny zprávy. Čím nižší ID, tím má zpráva vyšší prioritu. Komunikace je bezeztrátová.

To, jak se s 11bitovým ID a 8 bajty dat nakládá, upravují protokoly aplikační vrstvy. My máme v našich zařízeních implementovány základní funkce protokolu CANOpen.



Obr. 1: Struktura rámce dle standardu CAN 2.0 A

Každé zařízení v síti (používá se i termín sběrnice) CAN má dle standardu aplikační vrstvy CANOpen svoje jedinečné ID v intervalu 1 – 127. Každé takové zařízení může po CANu komunikovat dvěma základními způsoby, a to pomocí objektů PDO (process data object) nebo SDO (service data object). PDO se používají zpravidla pro periodické odesílání dat z/do zařízení. SDO se používají pro nastavování/vyčítání různých provozních parametrů (jedná se o peer-to-peer komunikaci s potvrzováním). Každé zařízení může přijímat i vysílat až 4 různé PDO a jeden SDO.

Rozlišení toho, zda se jedná o SDO, PDO či jinou formu komunikace popsanou standardem CANOpen se provádí pomocí ID zprávy. ID zprávy se tvoří následovně:

ID zprávy = CANOpenID zařízení (1..127) + kód funkce (viz tab. 1)

Tab. 1: Kódy vybraných funkcí ve standardu CANOpen

	Směr do zařízení	Směr ze zařízení
PDO1	512	384
PDO2	768	640
PDO3	1024	896
PDO4	1280	1152
SDO	1536	1408

V případě SDO se první 4 B datového pole používají pro indexaci uvnitř tzv. adresáře objektů a rozlišení typu komunikace, další 4 B pak obsahují hodnotu zapisovaného/čteného parametru. Detailní popis SDO není cílem tohoto textu.

V případě PDO je všech 8 B dat ve zprávě ponecháno v gesci programátora. Knihovna CANUSB32.DLL v rámci těchto 8 B zavádí model očíslovaných proměnných, které sw aplikace může číst nebo do nich zapisovat. Tomuto číslu (indexu) říkáme číslo kanálu/sub kanálu. Použijeme-li při volání některé z dále

popisovaných funkcí jako tento index ID zprávy dle standardu CANOpen, hovoříme pak o **čísle kanálu**, pracuje tato funkce s celým 8B datovým polem zprávy. Můžeme ho číst z příchozí fronty, zapsat do odchozího bufferu nebo odeslat.

číslo kanálu = ID zprávy

Použijeme-li index popsaný níže, přistupují funkce knihovny přímo k virtuálním proměnným a programátor se nemusí starat o pořadí bytů ve zprávě (little/big endian), maskování a celkovou strukturu 8B dat, která je popsána konfiguračním souborem dodaným výrobcem zařízení. Tímto způsobem lze proměnné číst z příchozí fronty a zapisovat do odchozího bufferu. Postup tvorby **čísla sub kanálu:**

číslo sub kanálu = CANOpenID*100 + 10000 + pořadové číslo proměnné v rámci zařízení

Pořadové číslo se generuje podle pořadí v jakém jsou proměnné uvedeny v konfiguračním *.COB souboru. Pro přijímané i vysílané proměnné je jen jedno společné číslování.

Příklad:

Mějme detekční kartu interferometru „DETINF“ s CANOpenID = 42. Chceme vyčíst stav počítadla a hodnoty signálů X a Y. Víme, že karta tyto proměnné posílá uvnitř svého PDO1 a jsou to první tři proměnné popsané v jejím konfiguračním souboru *.COB. Hledáme číslo kanálu pro příjem celého PDO1 a čísla sub kanálů pro příjem jednotlivých proměnných.

číslo kanálu pro PDO1 = ID zprávy pro PDO1 = 42 + 384 = 426

*číslo sub kanálu pro 1. proměnnou = 42*100 + 10000 + 1 = 14201*

*číslo sub kanálu pro 2. proměnnou = 42*100 + 10000 + 2 = 14202*

*číslo sub kanálu pro 3. proměnnou = 42*100 + 10000 + 3 = 14203*

Ukázka číslovaní kanálů a sub kanálů pro DETINF2 kartu s CANOpenID = 42

	Name	Dir	Type	Count	Data	CAN	Ch#
682	I042_DETINF2_rx1	rx	DOMAIN			PDO	426
683	I042_DETINF2_rx1_count	rx	INTEGER32			PDO	14201
684	I042_DETINF2_rx1_x_axis	rx	INTEGER16			PDO	14202
685	I042_DETINF2_rx1_y_axis	rx	INTEGER16			PDO	14203
686	I042_DETINF2_rx2	rx	DOMAIN			PDO	682
687	I042_DETINF2_rx2_dig_inputs	rx	UNSIGNED8			PDO	14204
688	I042_DETINF2_rx1	rx	DOMAIN			PDO	938
689	I042_DETINF2_rx1_count64_H	rx	INTEGER32			PDO	14205
690	I042_DETINF2_rx1_count64_L	rx	INTEGER32			PDO	14206
691	I042_DETINF2_SDOrx	rx	DOMAIN			SDO	1450
692	I042_DETINF2_SDOrx_Command	rx	UNSIGNED8			SDO	14207
693	I042_DETINF2_SDOrx_Index	rx	UNSIGNED16			SDO	14208
694	I042_DETINF2_SDOrx_SubIndex	rx	UNSIGNED8			SDO	14209
695	I042_DETINF2_SDOrx_Data	rx	UNSIGNED32			SDO	14210
696	I042_DETINF2_SDOtx	tx	DOMAIN			SDO	1578
697	I042_DETINF2_SDOtx_Command	tx	UNSIGNED8			SDO	14211
698	I042_DETINF2_SDOtx_Index	tx	UNSIGNED16			SDO	14212
699	I042_DETINF2_SDOtx_SubIndex	tx	UNSIGNED8			SDO	14213
700	I042_DETINF2_SDOtx_Data	tx	UNSIGNED32			SDO	14214

Toto je screenshot s aplikace CANMAN, kterou najdete v několika exemplářích v dodaném balíku obslužného SW. Je to dobrý pomocník mimo jiné i při zjišťování čísel kanálů (červeně) a subkanálů (modře).

Popis základních funkcí CANUSB32.DLL

`bool  CANInit(char * IniName)`

Provede inicializaci knihovny na základě souboru popisujícího CAN sběrnici. Jméno souboru se předává parametrem IniName, jedná se o standardní nulou zakončený řetězec.

Konfigurační *.CAN soubor obsahuje:

- Nastavení typu rozhraní, kterým se aplikace připojí k CAN sběrnici. Typicky se volí mezi přímým připojením přes USB/CAN interface, TCP/IP připojení přes server NETCANS a simulátorem sítě. Parametr *COMTYPE.*
- Pokud se připojujeme přes TCP/IP, tak IP adresu serveru. Parametr *server.*
- CANOpen ID zařízení (číslo 1-127), ze kterých chceme přijímat data nebo jim něco posílat. Sekce *[CanDevice001]*, *[CanDevice002]* atd. Parametr *CanOpenID*
- Formát proměnných přijímaných/odesílaných z/do zařízení. Parametr Device v dané sekci odkazuje na *.COB soubor obsahující popis proměnných. Tento soubor dodává výrobce zařízení.
- Pozor! Pokud zařízení není uvedeno v *.CAN souboru, nemůže s ním aplikace komunikovat.

Vrací:

`true` – inicializace se zdařila

`false` – chyba při inicializaci

Příklad:

```
if (!CANInit("moje_sit.can")) {
      printf("Chyba pri inicializaci CANu!\n");
      return -1;
}
```

`bool  CANClose(void)`

Zavře knihovnu.

Vrací:

`true` – deinicializace se zdařila.

`false` – chyba při deinicializaci, např. že třeba nebylo co deinicializovat.

Příklad:

```
if (!CANClose()) {
      printf("Chyba pri deinicializaci CANu!\n");
      return -1;
}
```

FUNKCE PRO PŘÍJEM DAT ZE SÍTĚ CAN

`long CANReadChanNum(void)`

Vrací číslo přijatého kanálu, který je aktuálně k dispozici ke čtení z přijímacího bufferu. Vrací 0, pokud je přijímací buffer prázdný. Typicky se tato funkce volá v nekonečné smyčce v separátním threadu. Pokud vrátí nenulovou hodnotu, provedeme pomocí funkce `CANReadChan` načtení 8 bajtů přijatých dat.

`long CANReadSubChanNum(void)`

Vrací číslo přijatého sub kanálu (proměnné), která je aktuálně k dispozici ke čtení z přijímacího bufferu. Vrací 0, pokud je buffer prázdný. Typicky se tato funkce volá v nekonečné smyčce v separátním threadu. Pokud vrátí nenulovou hodnotu, provedeme pomocí funkce `CANReadChan` načtení proměnné.

`bool CANReadChan( long ChanNum, void *pData)`

Do bufferu pData načte z přijímacího bufferu buď celý CAN objekt (8 bajtů), nebo proměnnou. Podle toho, zda je v parametru ChanNum číslo kanálu nebo číslo sub kanálu.

Příklad:

```
// Ukazka nacitani promennych z detekcni karty interferometru
"DETINF2" // s CANOpenID = 42

while(RxThreadControl){
      long SubChanNum = CANReadSubchanNum();
      switch(SubChanNum){
            case 14001: //POCITADLO
                        CANReadChan(14001,&CNT);
                  break;

            case 14002: //OSA X
                        CANReadChan(14002,&X);

                        //cteni chybovych priznaku
                        ERR_VELOCITY    = ((X&1)!=0);
                        ERR_MAGNITUDE   = ((X&2)!=0);

                        //ocistime data ADC od chybovych priznaku
                        X = X&0xFFFC;
                  break;
            case 14003: //OSA Y
                        CANReadChan(14001,&Y);
                  break;
            default:
                  break;
      }
}
```

FUNKCE PRO ODESÍLÁNÍ DAT NA SÍŤ CAN

`bool  CANWriteChan( long ChanNum, void *pData)`

Zapíše data odkazovaná pointerem pData do odesílacího bufferu pro kanál nebo subkanál udávaný parametrem ChanNum. V případě použití čísla kanálu si funkce z dané adresy přečte celých 8 bajtů. V případě použití čísla subkanálu si funkce načte tolik dat, kolik je dle definice proměnné v souboru *.COB potřeba, tj. např. pro INTEGER32 to budou 4 bajty, pro UNSIGNED16 2 bajty atd.

`bool  CANWriteChanNum(long ChanNum)`

Na síť CAN fyzicky odešle data zapsaná v předchozím kroku do odesílacího bufferu funkcí CANWriteChan. Použijeme-li jako parametr číslo kanálu, odešlou se naráz všechny jeho subkanály. Použijeme-li číslo subkanálu, odešlou se kromě něj i všechny ostatní subkanály patřící pod stejný kanál. To je dáno tím, že 8B zpráva na CAN musí vždy odejít celá.

Přidání CANUSB32.DLL do projektu C/C++

Knihovna byla přeložena v prostředí MS Visual Studio 6.0. Pomocí utilit vámi používaného IDE je možné vytvořit *.LIB soubor a ten společně s hlavičkovým souborem CANUSB32.H přidat do projektu. To máme vyzkoušeno jak na starších verzích MSVC tak Borland C++ Builderu. V novějších verzích začíná být trochu zmatek v tom, jaký který linker požaduje formát *.LIB souboru. Jako nejjistější a nejfunkčnější cesta se v aktuálních prostředích jeví zavedení knihovny pomocí funkce LoadLibrary z Windows API, tak je to provedeno v ukázkovém souboru, který přikládám.

FORMÁT DAT ODESÍLANÝCH DETEKČNÍ KARTOU KARTOU DETINF2

Kanál PDO1

Číslo kanálu = CANOpenID + 384

Subkanál COUNT

Číslo subkanálu = CANOpenID *100 + 10000 + 1

Datový typ: int32

Význam: počítadlo interferenční fáze, 1024 LSB odpovídá jednomu interferenčnímu proužku

Subkanál X_AXIS

Číslo subkanálu = CANOpenID *100 + 10000 + 2

Datový typ: int16

Význam: signál osy X z detekční jednotky interferometru + chybové stavy

Bit	15-2	1	0
Význam	Hodnota z A/D převodníku	Podkročení minimální amplitudy	Překročení max. rychlosti

Subkanál Y_AXIS

Číslo subkanálu = CANOpenID *100 + 10000 + 3

Datový typ: int16

Význam: signál osy Y z detekční jednotky interferometru

Bit	15-2	1	0
Význam	Hodnota z A/D převodníku	-	-

KOMUNIKACE S KARTOU DETINF2 POMOCÍ PROTOKOLU SDO

Karta pro nastavování různých provozních parametrů používá zjednodušenou verzi protokolu CANOpen SDO.

Karta má v paměti RAM a FLASH strukturu nazývanou adresář objektů. Každý objekt je jednoznačně identifikován 16bitovým indexem a 8bitovým sub indexem. Objekty mohou být typu char, u/int16, u/int32, float32.

Jedná se o komunikaci, která má vždy 2 kroky – příkaz a odpověď.

Postup zápisu:

1. Aplikace odešle příkaz k zápisu hodnoty, index, sub index a hodnotu.
2. Zařízení odpoví potvrzením zápisu, indexem, sub indexem a zapsanou hodnotou. V případě chyby odpoví chybovým kódem.

Postup čtení:

1. Aplikace odešle příkaz ke čtení hodnoty, index, sub index. Pole s hodnotou je ve zprávě ignorováno.
2. Zařízení odpoví potvrzením čtení, indexem, sub indexem a čtenou hodnotou. V případě chyby odpoví chybovým kódem.

První krok komunikace, tedy příkaz, se provádí prostřednictvím kanálu s názvem SDOtx a jeho subkanálů.

Kanál SDOtx

Číslo kanálu = CANOpenID + 1536

Subkanál SDOtx_Command

Číslo subkanálu = CANOpenID*100 + 10000 + 11
Datový typ: uint8
Význam: Identifikuje, zda se bude jednat o zápis do adresáře objektů, nebo žádost o vyčtení dat.

Příkaz	**Hodnota SDOtx_Command**
Zápis do adresáře objektů	43
Žádost o vyčtení dat	64

Subkanál SDOtx_Index

Číslo subkanálu = CANOpenID*100 + 10000 + 12
Datový typ: uint16
Význam: Index zapisované nebo čtené hodnoty v adresáři objektů.

Subkanál SDOtx_SubIndex

Číslo subkanálu = CANOpenID*100 + 10000 + 13
Datový typ: uint8
Význam: Sub index zapisované nebo čtené hodnoty v adresáři objektů.

Subkanál SDOtx_Data

Číslo subkanálu = CANOpenID*100 + 10000 + 14

Datový typ: uint32

Význam: 4 bajty obsahující zapisovanou hodnotu. Při žádosti o vyčtení dat je hodnota ignorována.

Druhým krokem komunikace je odpověď od zařízení na kanálu SDOrx.

Kanál SDOrx

Číslo kanálu = CANOpenID + 1408

Subkanál SDOrx_Command

Číslo subkanálu = CANOpenID*100 + 10000 + 7

Datový typ: uint8

Význam: Sděluje, zda se jedná o potvrzení zápisu, vyčtená data, nebo chybový stav.

Příkaz	**Hodnota SDOtx_Command**
Potvrzení zápisu	96
Vyčtená data	66
Chybový stav	128

Subkanál SDOrx_Index

Číslo subkanálu = CANOpenID*100 + 10000 + 8

Datový typ: uint16

Význam: Index zapisované nebo čtené hodnoty v adresáři objektů.

Subkanál SDOrx_SubIndex

Číslo subkanálu = CANOpenID*100 + 10000 + 9

Datový typ: uint8

Význam: Sub index zapisované nebo čtené hodnoty v adresáři objektů.

Subkanál SDOtx_Data

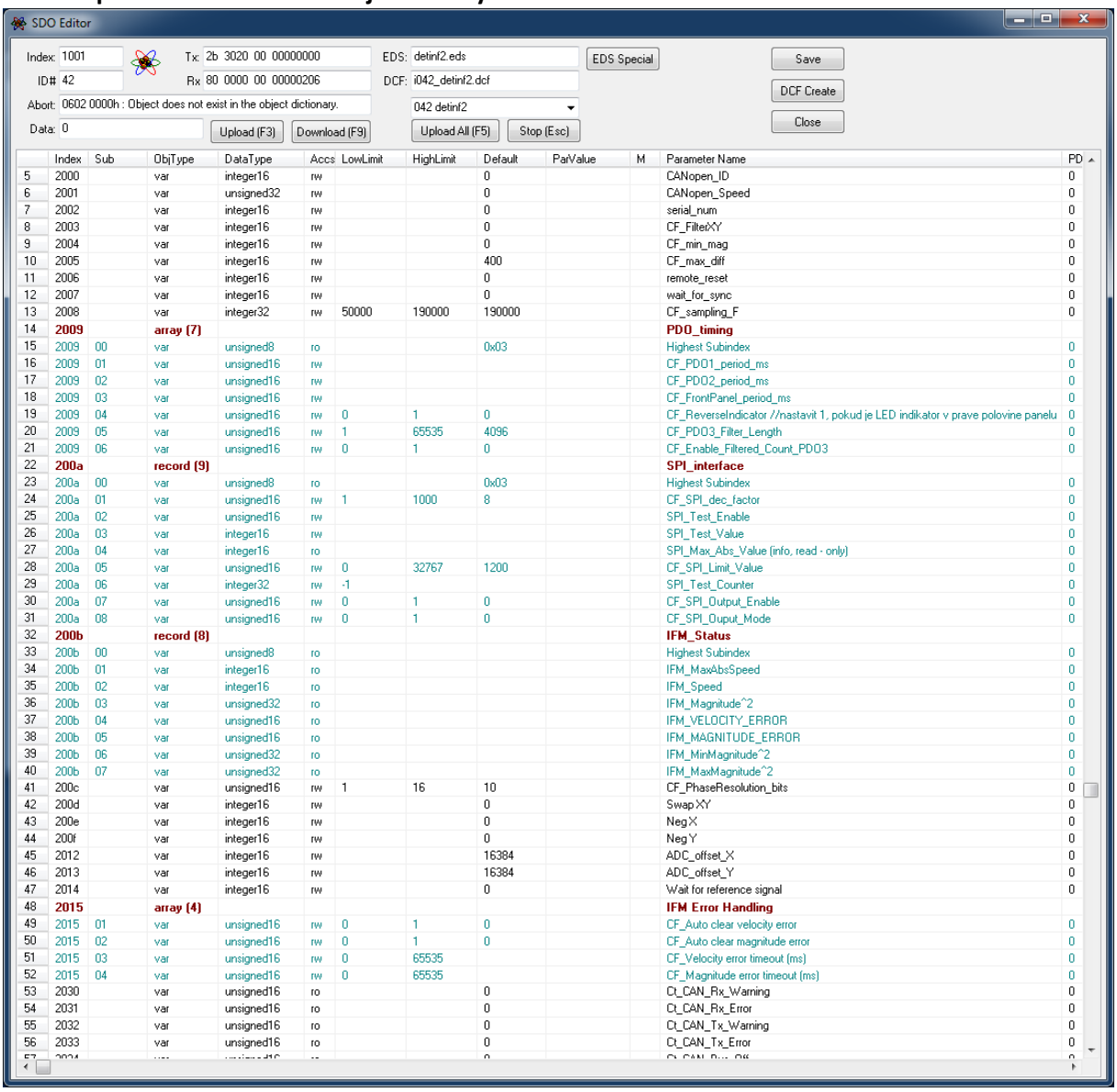
Číslo subkanálu = CANOpenID*100 + 10000 + 10

Datový typ: uint32

Význam: 4 bajty obsahující zapisovanou nebo čtenou hodnotu. V případě chybového stavu obsahuje kód chyby.

Typ chyby	**Kód chyby**
Neexistující objekt (nesprávný index nebo subindex)	0x05040001
Zapisovaná hodnota je vyšší než maximální možná	0x06090031
Zapisovaná hodnota je nižší než minimální možná	0x06090032

Přehled parametrů v adresáři objektů karty DETINF2:



	Index	Sub	ObjType	DataType	Accs	LowLimit	HighLimit	Default	ParValue	M	Parameter Name	PD
5	2000		var	integer16	rw			0			CANopen_ID	0
6	2001		var	unsigned32	rw			0			CANopen_Speed	0
7	2002		var	integer16	rw			0			serial_num	0
8	2003		var	integer16	rw			0			CF_FilterXY	0
9	2004		var	integer16	rw			0			CF_min_mag	0
10	2005		var	integer16	rw			400			CF_max_diff	0
11	2006		var	integer16	rw			0			remote_reset	0
12	2007		var	integer16	rw			0			wait_for_sync	0
13	2008		var	integer32	rw	50000	190000	190000			CF_sampling_F	0
14	**2009**		**array (7)**								**PDO_timing**	
15	2009	00	var	unsigned8	ro			0x03			Highest Subindex	0
16	2009	01	var	unsigned16	rw						CF_PDO1_period_ms	0
17	2009	02	var	unsigned16	rw						CF_PDO2_period_ms	0
18	2009	03	var	unsigned16	rw						CF_FrontPanel_period_ms	0
19	2009	04	var	unsigned16	rw	0	1	0			CF_ReverseIndicator //nastavit 1, pokud je LED indikator v prave polovine panelu	0
20	2009	05	var	unsigned16	rw	1	65535	4096			CF_PDO3_Filter_Length	0
21	2009	06	var	unsigned16	rw	0	1	0			CF_Enable_Filtered_Count_PDO3	0
22	**200a**		**record (9)**								**SPI_interface**	
23	200a	00	var	unsigned8	ro			0x03			Highest Subindex	0
24	200a	01	var	unsigned16	rw	1	1000	8			CF_SPI_dec_factor	0
25	200a	02	var	unsigned16	rw						SPI_Test_Enable	0
26	200a	03	var	integer16	rw						SPI_Test_Value	0
27	200a	04	var	integer16	ro						SPI_Max_Abs_Value (info, read - only)	0
28	200a	05	var	unsigned16	rw	0	32767	1200			CF_SPI_Limit_Value	0
29	200a	06	var	integer32	rw	-1					SPI_Test_Counter	0
30	200a	07	var	unsigned16	rw	0	1	0			CF_SPI_Output_Enable	0
31	200a	08	var	unsigned16	rw	0	1	0			CF_SPI_Ouput_Mode	0
32	**200b**		**record (8)**								**IFM_Status**	
33	200b	00	var	unsigned8	ro						Highest Subindex	0
34	200b	01	var	integer16	ro						IFM_MaxAbsSpeed	0
35	200b	02	var	integer16	ro						IFM_Speed	0
36	200b	03	var	unsigned32	ro						IFM_Magnitude^2	0
37	200b	04	var	unsigned16	ro						IFM_VELOCITY_ERROR	0
38	200b	05	var	unsigned16	ro						IFM_MAGNITUDE_ERROR	0
39	200b	06	var	unsigned32	ro						IFM_MinMagnitude^2	0
40	200b	07	var	unsigned32	ro						IFM_MaxMagnitude^2	0
41	200c		var	unsigned16	rw	1	16	10			CF_PhaseResolution_bits	0
42	200d		var	integer16	rw			0			Swap XY	0
43	200e		var	integer16	rw			0			Neg X	0
44	200f		var	integer16	rw			0			Neg Y	0
45	2012		var	integer16	rw			16384			ADC_offset_X	0
46	2013		var	integer16	rw			16384			ADC_offset_Y	0
47	2014		var	integer16	rw			0			Wait for reference signal	0
48	**2015**		**array (4)**								**IFM Error Handling**	
49	2015	01	var	unsigned16	rw	0	1	0			CF_Auto clear velocity error	0
50	2015	02	var	unsigned16	rw	0	1	0			CF_Auto clear magnitude error	0
51	2015	03	var	unsigned16	rw	0	65535				CF_Velocity error timeout (ms)	0
52	2015	04	var	unsigned16	rw	0	65535				CF_Magnitude error timeout (ms)	0
53	2030		var	unsigned16	ro			0			Ct_CAN_Rx_Warning	0
54	2031		var	unsigned16	ro			0			Ct_CAN_Rx_Error	0
55	2032		var	unsigned16	ro			0			Ct_CAN_Tx_Warning	0
56	2033		var	unsigned16	ro			0			Ct_CAN_Tx_Error	0

Pozn. indexy a subindexy jsou v hexadecimálním formátu.

Nejpoužívanější SDO karty DETINF2

Vysvětlivky: rw – lze číst i zapisovat; wo – pouze k zápisu; ro – pouze ke čtení; flash – proměnná uložena ve flash, nedoporučuje se ji často přepisovat

CF_min_mag (rw, flash)
Index: 0x2004
Sub Index: 0x00
Význam: Nastavení prahu pro chybu příliš nízké intenzity. Podkročení nastaví chybový příznak ERR_MAGNITUDE na 1. Smaže se jedině resetem. U novější verze karty funguje automatické mazání chyb po určitém čase (tu zatím nemáte).

CF_max_diff (rw, flash)
Index: 0x2005
Sub Index: 0x00
Význam: Nastavení prahu pro chybu příliš rychlého pojezdu. Rychlost je vyjádřena jako změna hodnoty počítadla proužků mezi 2 po sobě jdoucími vzorky A/D převodníku. Doporučujeme neměnit přednastavenou hodnotu. Překročení nastaví chybový příznak ERR_SPEED na 1. Smaže se jedině resetem. U novější verze karty funguje automatické mazání chyb po určitém čase (tu zatím nemáte).

Remote_reset (wo)
Index: 0x2006
Sub Index: 0x00
Význam: Zápis hodnoty 1 provede reset počítadla a chybových příznaků.

CF_PDO1_period_ms (rw, flash)
Index: 0x2009
Sub Index: 0x01
Význam: Perioda odesílání vzorků signálu X, Y a počítadla na CAN (PDO1). Doporučujeme nenastavovat kratší než 4 ms.

IFM_MaxAbsSpeed (ro)
Index: 0x200B
Sub Index: 0x01
Význam: Lze vyčíst maximální pozorovanou změnu údaje počítadla mezi 2 po sobě jdoucími vzorky A/D převodníku. Resetem se statistika nuluje.

IFM_Speed (ro)
Index: 0x200B
Sub Index: 0x02
Význam: Lze vyčíst aktuální změnu údaje počítadla mezi 2 po sobě jdoucími vzorky A/D převodníku.

IFM_Magnitude^2 (ro)
Index: 0x200B
Sub Index: 0x03
Význam: Lze vyčíst druhou mocninu aktuální amplitudy interferenčního proužku ($I^2 = X^2 + Y^2$).

IFM_VELOCITY_ERROR (ro)

Index: 0x200B
Sub Index: 0x04
Význam: Lze vyčíst stav chybového příznaku VELOCITY_ERROR.

IFM_MAGNITUDE_ERROR (ro)
Index: 0x200B
Sub Index: 0x05
Význam: Lze vyčíst stav chybového příznaku MAGNITUDE_ERROR.

IFM_MinMagnitude^2 (ro)
Index: 0x200B
Sub Index: 0x06
Význam: Druhá mocnina minimální pozorované amplitudy interferenčního proužku (od posledního resetu).

IFM_MaxMagnitude^2 (ro)
Index: 0x200B
Sub Index: 0x07
Význam: Druhá mocnina maximální pozorované amplitudy interferenčního proužku (od posledního resetu).

ADC_offset_X (rw, flash)
Index: 0x2012
Sub Index: 0x00
Význam: Offset A/D převodníku pro osu X. Lze doladit pozici kružnice/elipsy opisované signály X a Y tak, aby měla střed o souřadnicích (0, 0). Základní hodnota pro dokonalý signál bez offsetu je 16 384.

ADC_offset_Y (rw, flash)
Index: 0x2013
Sub Index: 0x00
Význam: Offset A/D převodníku pro osu Y. Lze doladit pozici kružnice/elipsy opisované signály X a Y tak, aby měla střed o souřadnicích (0, 0). Základní hodnota pro dokonalý signál bez offsetu je 16 384.

Pozn.: U obou offsetů se jedná o číslo, které se od hodnoty vzorku z A/D převodníku automaticky hardwarově odčítá. Tzn. menší hodnota parametru způsobí posun signálu do více kladných hodnot. Doporučujeme nejprve vyladit interferometr na max. kontrast a intenzitu. Modifikace offsetů je až poslední možnost.