c't-Lab Syntax Stand 21.05.2010

c't-Lab liefert kein Echo, bei einem Terminal-Programm deshalb ggf. lokales Echo einstellen. Nur ein Befehl pro Zeile. Befehle werden erst nach dem Empfang von CR oder CR/LF verarbeitet.
Schnittstellen-Parameter: 38400 Bd, 8n1. Backspace (#8) löscht letztes Zeichen aus dem Befehlszeilenpuffer, andere Control-Zeichen werden ignoriert.
Ab Firmware-Versionen X.6 besteht die Option, dem Befehl/der Abfrage eine XOR8-Prüfsumme anzuhängen, in der Form $<HEXbyte> mit der XOR-Summe über den gesamten Befehlsstring
also z.B. 0:VAL 20=1.234!$45
Das fordert das jeweilige Modul auf, die Prüfsumme gegen die errechnete zu checken und im Fehlerfall mit dem Fehlercode 7 abzubrechen.
Lässt man bei Ausgabe-Befehlen das "!" weg, erfolgt keine Ausgabe des "#0:255=0 [OK]" Prompts (vermindert Datenmenge bei kritischen Anwendungen)

allgemein

Cmd	Argument	SubCh	Wertebereich	Beispiel-Befehle	Beispiel-Antwort	Erläuterung
IDN	--	254		IDN?, *:IDN?	#0:255=1.39 [ADA-IO by c't]	Identifizierung, '*' als Moduladresse gilt für alle Slave-Channels
VAL	0..255	0..255	div.	0:VAL 20=5.0!, VAL 10?, 0:10?, 20=5.0!	#0:20=5.0000, #0:2=0.0	Allg. Form, Messwert-Spannung in Volt, VAL kann auch weggelassen werden
STR**	--	255		*:STR?, 255?	#0:255=0 [OK]	Status-Request, Bit 7 (+128)=Busy, 6 (+64)=UserSRQ, 5 (+32)=OverLoad, 4 (+16)=!!! Bit 3..0=Fehler- oder Button-Nr. (bei UserSRQ) Buttons: 1=Down/Strom, 2=Up/Spannung, 3=Enter/Feineinstellung (Inkrementalgeber-Druckkontakt)
ERC	--	251	Integer	ERC?	#0:251=0	Fehler-Zähler auslesen oder setzen, Zähler wird bei jedem Empfangs-Übertragungsfehler um 1 erhöht
SBD	--	252	Byte	252=51!, 252?	#0:252=51	Serielle Baudraten-Einstellung, UBRR-Wert des ATmega32 mit U2X-Bit=1, siehe Seite 165 Datenblatt ATmega32 und Status_Latenzen-Arbeitsblatt, erst nach Reset gültig
WEN	--	250	0..1	WEN=1!, 1:250=1!	#0:255=16 [OK]	EEPROM-Write-Enable-Bit, vor dem Beschreiben von Werten mit Schreibsperre auf 1 setzen, wird danach wieder automatisch auf 0 gesetzt

** nur Lesen

ADA-IO Version 1.74

Cmd	Argument	SubCh	Wertebereich	Beispiele (hier für Adresse 0)	Beispiel-Antwort	Erläuterung
VAL	0..7	0..7	Float	VAL 0?, 0:7?	#0:2=0.0	ATmega-interner ADC 10 Bit, Eingangsspannung unipolar 0..10 (V)
VAL	10..17	10..17	-10 bis +10	10?	#0:10=10.002	AD16-8, 16-Bit-Wandler, 8 Kanäle, Spannungswert -10..+10 (V)
VAL	20..27	20..27	-10 bis +10	0:VAL 20=5.0!, 24=1.2345!	#0:20=5.0000	Ausgabewerte DA12-8 oder DA16-8, 8 Kanäle, Spannungswert -10..+10 (V)
OFS*	0..27	100..127	Integer	0:OFS 20=37!, OFS 10?	#0:255=0 [OK], #0:110=302	Wandler-Offset in Raw-Digits (Integer-Wert)
SCL*	0..27	200..227	Float	0:SCL 20=1.0!, SCL 10?	#0:255=0 [OK], #0:210=1.0123	Skalierungen AD- oder DA-Wert
SCL*	9	209	Float	0:SCL 9=100!, SCL 9?	#0:255=0 [OK], #0:209=100.0	Grundskalierung interner AD-Wandler des ATmega32 (nominal 100)
SCL*	19	219	Float	0:SCL 19=3185!, SCL 19?	#0:255=0 [OK], #0:219=3185	Grundskalierung AD16-8-Wandler LTC1684 (nominal 3185)
SCL*	28, 29	228..229	Float	0:SCL 28=200!, SCL 29?	#0:255=0 [OK], #0:229=3200	Grundskalierung DA12-8-Wandler, SCL 28 für 12 Bit LTC1257 (nominal 200), SCL 29 für 16 Bit LTC1655 (nom. 3200)
RAW**	0..17	50..67	Integer	RAW 17?	#0:67=31548	Rohdaten direkt vom Wandler ohne Skalierung und Offset
RAW**	20..27	70..77	Integer	RAW 27?	#0:77=32767	Rohdaten, die D/A-Wandler nach Skalierung und Offset erhält
PIO	0..7	30..37	Byte	0:PIO 0?, 30?, 0:31=255!, 0:VAL 30?	#0:30=123	IO-Port 0..7 Datenbyte
DIR	0..7	40..47	Byte	DIR 0=255!, 40=255!	#0:40=255	IO-Port 0..7 Richtung, Bit auf 1 = Ausgang. Ab Version 1.74 flüchtig, d.h. bleibt nur bis zum nächsten Reset bestehen; keine Schreibfreigabe mit WEN=1 mehr erforderlich!
DSP	0..9	80..89				Display/Panel-Parameter, 2 bis 8 derzeit unbenutzt
DSP	0	80	Byte	DSP 0=20!, DSP=20!, DSP?, 0:80?	#0:80=20	Panel-Messwertanzeige, bringt Messwert-Kanal (0..27) auf Display
DSP*	1	81	0..27 [bel.Text]	DSP 1=20 [Volt]!, 81=10 [Fuellstd]!, 81=11 []!	#0:255=0 [OK]	Text unter dem auf Display angezeigten Messwert, nichtflüchtig, Wert steht korrespondierenden VAL-SubCh, String mit max. 8 Zeichen in [Klammern], [] stellt Default wieder her (Value xx)
DSP*	9	89	Byte	DSP 9=4!, 0:89=2!	#0:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)
OPT*	6	156	0..1	OPT 6=1!	#0:255=0 [OK]	Gleiche Funktion wie REF (SubCh 246)
OPT*	7	157	0..1	OPT 7=1!	#0:255=0 [OK]	Integrier-Modus für AD16-8 (ruhigere Anzeige, etwas langsamere Reaktion, default Aus=0)
OPT*	9	159	Float	OPT 9=4!	#0:255=0 [OK]	Gleiche Funktion wie DSP 9 (SubCh 89)
OPT*	30..37	180..187	Byte	OPT 30=255!	#0:255=0 [OK]	IO8-32 Ports Einschaltwerte (Default), werden unmittelbar nach Reset auf Ports 0..7 ausgegeben
OPT*	40..47	190..197	Byte	OPT 40=127!	#0:255=0 [OK]	IO8-32 Port-Datenrichtungen Einschaltwerte (Default), werden unmittelbar nach Reset auf Ports 0..7 eingestellt
ALL**	0..4	95..99	--	ALL 4?, 0:95?	(Liste mit Werten)	0=VAL 0..7 (AD10), 1=VAL 10..17 (AD16), 3=VAL 30..37 (Ports), 4=Alle verfügbaren
TRM*	0..3	240..243	Byte	TRM 0=31!, TRM1?	#0:255=0 [OK], #0:241=7	Trigger-Masken für AD-Ports 0..7 (TRM0) und 10..17 (TRM1) sowie I/O-Ports 30..37 (TRM3)
TRT*	--	247	0, 20..31767	TRT=0, 0:247=2000	#0:255=0 [OK]	Autom. Trigger-Timing in ms, 0=Aus, Werte ab 20 ms möglich
TRL*	0	248	0..1	TRL=1!, TRL?, 248=0!, 248?	#0:255=0 [OK], #0:248=1	Trigger-Edge PB2, neg. = 0, pos. = 1, Impuls an diesem Pin liefert Messwerte wie in Trigger-Maske definiert
TRG	--	249	--	TRG!, TRG?, 249?	#0:255=0 [OK] (Liste mit Werten)	manuelle Trigger-Auslösung, liefert Messwerte wie in Trigger-Maske definiert
REF*	--	246	0..1	REF=1!, REF?, 246?	#0:255=0 [OK], #0:246=0	Umschaltung interne (1) / externe (0) Referenz, Default = 1, intern
ICB	--	230	Byte	ICB=0!, 0:230?	#0:255=0 [OK], #0:230=255	Generische I2C-Befehle, Byte Lesen/Schreiben, Bausteinadresse mit ICA eingestellt
ICW	--	231	Word/Integer		#0:255=0 [OK], #0:231=1024	Generische I2C-Befehle, Word Lesen/Schreiben, Bausteinadresse mit ICA eingestellt
ICS	--	232	Word (swapped)		#0:255=0 [OK], #0:232=32412	Generische I2C-Befehle, Word mit vertauschten Bytes Lesen/Schreiben, Bausteinadresse mit ICA eingestellt
ICT**	0..1	233, 234	Float	ICT?, ICT 0?, 0:ICT 1?, 234?	#0:255=0 [OK], #0:233=24.5	Temperatur-Sensor auslesen, "0"=LM75, "1"=DS1621/1631
ICA	--	239		ICA=72!	#0:255=0 [OK]	I2C-Hardware-Bausteinadresse einstellen, 7 Bit (0..127) ohne R/W-Bit, z.B. 72 für LM75

* nichtflüchtige EEPROM-Werte mit Schreibsperre
** nur Lesen

DDS Version 3.70

Cmd	Argument	SubCh	Wertebereich	Beispiele (hier für Adresse 1)	Beispiel-Antwort	Erläuterung
VAL	0..5	0..5	Float	1:VAL 1=775!, VAL 2?, 3?, 0=1000!	#1:255=OK, #1:2=0.0	Allg. Form, VAL kann auch weggelassen werden, Aufruf der folgenden Parameter:
FRQ	--	0	Float	FRQ=1000!, 0=440!	#1:0=1000.0	Frequenz in Hz, Auflösung 1/10 Hz
LVL	--	1	Float	LVL=775.0!, 1=1000!	#1:1=775	Pegel in mVeff, auch bei Dreieck, Rechteck True-RMS-Wert
LVP	--	2	Float	LVP=1000!, 1=5000!		Vss-Pegel in mV (Spitze-Spitze), wird intern umgerechnet
DBU	--	3	Float	DBU=0.0!, 2=-6!		Pegel in dBu, 0 dBu=774,6 mVeff
WAV	--	4	0..5	WAV=1!, 4?		Wave, 0=aus, 1=Sinus, 2=Dreieck, 3=Rechteck, 4=Logik (mit automatischem Offset je nach Pegel), 5=extern Input
BST	--	5	0..100	BST=10!		Burst-Pause in 10ms-Schritten, 0=Burst aus (kontinuierlich)
INL**	0..2	10..12	Float	INL?, INL 0?, 11?	#1:10=775, #1:12=-10	Input-Level von TRMSC-Tochterplatine, 0=Veff, 1=Vss, 2=Veff umgerechnet in dB
RNG	--	19	0..3	RNG=1!	#1:255=OK, #1:19=2	Input-Range der TRMSC-Tochterplatine, 0=100mVeff, 1=1V, 2=10V, 3=100V
DCO	--	20	-10,0 bis +10,0	DCO=0!, DCO=5.0!, 5?	#1:255=0 [OK]	DC-Offset in Volt (-10 bis +10), Auflösung 5-mV-Schritte
DSP	0	80	0..6	DSP 0=0!, DSP=1!, DSP?, 0:80?	#1:80=20	Panel-Menuselect, bringt Menü (0..6) auf Display, 0=Wave, 1=Frequ, 2=Level, 3=Peak-Level, 4=InputLevel, 5=Burst, 6=DC-Offset
DSP*	9	89	Byte	DSP 9=4!, 0:89=2!	#1:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)
ALL**	0	99	--	ALL?, 0:99?	(Liste mit Messwerten)	Nur ein Messwert: gemessener Eingangs-Pegel
SCL*	0, 1	200, 201	Float	SCL 1=1.00014!, 201=1.00014!	#1:255=0 [OK]	Skalierungen für Pegelsteller, 0=Low-Bereich 0 bis 199 mVeff, 1=High-Bereich 200 bis 8000 mVeff
SCL*	2	202	Float	SCL 2=1.000!, 202=2!	#1:255=0 [OK]	Verstärkung Ausgangsstufe, Default 2, für 50-Ohm-Ausgang "1" eintragen wenn Abschlusswiderstand vorhanden
SCL*	3	203	Float	SCL 3=40!, 203=40!	#1:255=0 [OK]	Abschwächung passiver Abschwächer, 40 = 1/40stel
SCL*	10..13	210..213	Float	SCL 10=0.9976!, 210=0.9976!	#1:255=0 [OK]	Skalierungen für True-RMS-Konverter, 10=Bereich 100mV, 11=Bereich 1V, 12=Bereich 10V, 13=Bereich 100V
OPT*	0	150	Float	OPT 0=440!, 1:150=440!, 1:150?	#1:255=0 [OK], #1:150=1000	Init-Frequenz (default 1000Hz)
OPT*	1	151	Integer	OPT 1=1000!, 1:151=775!		Init-Pegel (default 775 mV)
OPT*	2	152	Integer	OPT 3=3300!		Init-Logikpegel (default 5000 mV)
OPT*	3	153	Float			Init-dB (nicht verwendet, obsolet durch OPT 1)
OPT*	4	154	Byte	OPT 4=3!		Init-Wave (default Sinus, 1)
OPT*	5	155	Byte	OPT 5=440!		Init-Burst (default 0ms=aus)
OPT*	20	170	Float	OPT 20=2.5!		Init-Offset in V (default 0V)
CMs Laborgerät (nicht in c't):						
PWR	--	5		PWR=2!		Outputmode, 0=normal, 1=symmetrisch, 2=PowerAmp an, 3=PowerAmp+Symmetrisch; nur für CMs Laborgerät
DSP	--	(6)		DSP	#1:255=OK	Laborgerät: Frequenz/Pegel auf Anzeige aktualisieren

* nichtflüchtige EEPROM-Werte mit Schreibsperre
** nur Lesen

DCG 2.9

(Erweiterungen von mcb in rot)

Cmd	Argument	SubCh		Beispiele (hier für Adresse 4)	Beispiel-Antwort	Erläuterung	Default-Werte	
VAL	0..15	0..15	Float	4:VAL 0=10.0!, 0?	#1:255=0 [OK], #1:2=0.1	Allg. Form für SubCh		
TMP**	0	233	Float	TMP?, 233?	#1:233=46	Temperatur des Kühlkörpers in °C		
DCV	--	0	Float	4:DCV=10.0!, 0?	#1:255=0 [OK], #1:2=0.0	Spannung U SOLL in Volt		
DCA	--	1	Float	DCA=100!, 1=20!		Strom I SOLL in A		
DCA	1	2	Float	DCA 1=100!, 2=20!		Strom I SOLL in mA		
DCA	2	3	Float	DCA 2=100!, 3=20!		Strom I SOLL in µA		
MAH	--	7	Float	MAH?, 7?, MAH=0!		Messwert kumulierter Strom in Ah (wird durch MAH=0! , MWH=0! auf 0 gesetzt)		
MWH	--	8	Float	MWH?, 8? MWH=0!		Messwert kumulierte Leistung in Wh (wird durch MAH=0! , MWH=0! auf 0 gesetzt)		
MSV**	--	10	Float	MSV?, 2?		Messwert U IST in Volt		
MSA**	--	11	Float	MSA?, 4:3?		Messwert I IST in A		
MSA**	1	12	Float	MSA 1?, 4:12?		Messwert I IST in mA		
MSA**	2	13	Float	MSA 2?, 4:13?		Messwert I IST in µA		
MSA**	4	15	Float	MSA 4?, 4:15?		Messwert ungeregelte Eingangsspannung U in Volt		
MSW**	--	18	Float	MSW?, 18?		tatsächlich abgegebene Leistung in Watt		
PCV	--	20	Float	4:PCV=100!, 20?	#1:255=0 [OK], #1:20=100	Prozentwert für Ausgangsspannung, 0=Aus, 100=mit DCV eing. Wert		
PCA	--	21	Float	PCA=100!, 21=33.333!		Prozentwert für Ausgangsstrom, 0=Aus, 100=mit DCA eing. Wert		
RON	--	27	Integer	RON=10!, 27=4!		Ripple (Brumm) On-Zeit in ms, nur gerade Werte, Uaus=DCV		
ROF	--	28	Integer	ROF=6!		Ripple (Brumm) Off-Zeit in ms, nur gerade Werte, Uaus=DCV* (100-RIP)/100		
RIP	--	29	Integer	RIP=25!, 29=25!		Ripple (Brumm) in Prozent von der Sollspannung		
DSP	0	80	Byte	DSP=4!, 0:80=0!, 80?	#1:255=0 [OK], #1:80=3	Angezeigtes Menü auf PM8-Panel 0..5, 0=U, 1=I, 2=Ripple%, 3=T on, 4=T off, 5=TrackCh		
DSP*	9	89	Byte	DSP 9=4!, 0:89=2!	#1:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)		
OFS*	0, 1	100, 101	Integer	OFS 0=5!, 101=4!, 101?		Offset Spannungs-DAC (Sollwert), 0=Low-Bereich, 1=High-Bereich		
OFS*	2..5	102..105	Integer	OFS 2=5!, 102=4!		Offset Strom-DAC (Sollwert), 2=2mA, 3=20mA, 4=200mA, 5=2A-Bereich		
OFS*	10, 11	110, 115	Integer	OFS 10=0!		Offset Spannungs-ADC (Istwert), 10=Low-Bereich, 11=High-Bereich		
OFS*	12..15	112..115	Integer	OFS 12=0!		Offset Strom-ADC (Istwert), 12=2mA, 13=20mA, 14=200mA, 15=2A-Bereich		
SCL*	0, 1	200, 201	Float	SCL 0=1.006!, 200=1.006!		Skalierung Spannungs-DAC (Sollwert), 0=Low-Bereich, 1=High-Bereich		
SCL*	2..5	202..205	Float	SCL 2=0.976!, 202=0.976!		Skalierung Strom-DAC (Sollwert), 2=2mA, 3=20mA, 4=200mA, 5=2A-Bereich		
SCL*	10, 11	210, 215	Float	SCL 0=1.006!, 200=1.006!		Skalierung Spannungs-ADC (Istwert), 10=Low-Bereich, 11=High-Bereich		
SCL*	12..15	212..215	Float	SCL 2=0.976!, 202=0.976!		Skalierung Strom-ADC (Istwert), 12=2mA, 13=20mA, 14=200mA, 15=2A-Bereich		
ALL**	0	99		ALL?, 4:99?	(Liste mit Messwerten)	ALL 0 enthält VAL 10, 11 und 15		
RAW**	0..1	50..51	Integer	RAW 0?	#1:50=31548	Rohdaten direkt vom 16-Bit-A/D-Wandler (so vorhanden) ohne Skalierung und Offset	**12 Bit**	**16 Bit**
RAW**	3..5	52..54	Integer	RAW 3?	#1:53=675	Rohdaten direkt vom internen A/D-Wandler (3 Kanäle Istspannung, Iststrom, Eingangsspannung), ohne Skalierung und Offset	**12 Bit**	**16 Bit**
RAW**	20..21	70..71	Integer	RAW 21?	#1:71=32767	Rohdaten, die D/A-Wandler nach Skalierung und Offset erhält	**12 Bit**	**16 Bit**
OPT*	0	150	Float	OPT 0=3.3!	#1:255=0 [OK]	Default-Ausgangsspannung	5.0	5.0
OPT*	1	151	Float	OPT 1=0.1!		Default-Ausgangsstrom	0.02	0.02
OPT*	2	152	Float			InitGainPre, Gain OpAmp U6 nach ADC	5.0	3.0
OPT*	3	153	Float			InitGainOut, Gain Sense-Differenzverstärker, (R21/R20) mit R23=R20 und R22=R21	3.0	3.0
OPT*	4	154	Float			InitGainI, Stromeinstellungs-Spannungsteiler R33/(R33+R34)	0.5	0.25
OPT*	5	155	Float			Uref LT1019 =2,5000V oder LTC1257 = 2.048V;	2.048	2.500
OPT*	6	156	Float			Umax maximale Ausgangsspannung für CheckLimits	20	20
OPT*	7	157	Float			R sense 1. Bereich nom. 2mA, 470 Ohm	470	470
OPT*	8	158	Float			R sense 2. Bereich nom. 20mA, 47 Ohm	47	47
OPT*	9	159	Float			R sense 3. Bereich nom. 200mA, 4.7 Ohm	4.7	4.7
OPT*	10	160	Float	OPT 10=0.452!		R sense 4. Bereich nom. 2A, 0.47 Ohm	0.47	0.47
OPT*	11	161	Float			Imax A 1. Bereich	0.002	0.002
OPT*	12	162	Float			Imax A 2. Bereich	0.020	0.020
OPT*	13	163	Float			Imax A 3. Bereich	0.200	0.200
OPT*	14	164	Float	OPT 14=1!		Imax A 4. Bereich, für CheckLimits	1000	2000
OPT*	15	165	Float			ADCUlofac, Skalierung Uist durch Spannungsteiler an Q1, R12/(R12+R24)	1	2
OPT*	16	166	Float			ADCUhifac, Skalierung Uist durch Spannungsteiler an Q1 R12/(R24+R12||R16)	5	6
OPT*	17	167	Float			InitOptions, Bit 0=1: LTC1655 statt 1257, Bit 1=1: LTC1864 vorhanden, Bit 2=1:DCP vorhanden	0	7
OPT*	18	168	Float			SwitchU, DAC-Umschaltpunkt Low/High-Bereich in Volt (Uoutmax ADC = 2.048V*InitGainOut)	6	12.1
OPT*	19	169	Float			untere Relais-Umschaltspannung (wieder auf 15V AC)	21	12.1
OPT*	20	170	Float			obere Relais-Umschaltspannung (wieder auf 30V AC)	22	12.5
OPT*	21	171	Float	OPT 21=60!		Lüfter-Einschalt-Temperatur	50	50
OPT*	22	172	Float	OPT 22=10!		Einschaltwert RON Ripple On-Time in ms, nur gerade Werte	4	4
OPT*	23	173	Float	OPT 23=10!		Einschaltwert ROF Ripple Off-Time in ms, nur gerade Werte	6	6
OPT*	24	174	Float	OPT 24=25!		Einschaltwert RIP Ripple Prozent (Brumm), 0..100	0	0

* nichtflüchtige EEPROM-Werte mit Schreibsperre

** nur Lesen

ACV 1.05

Cmd	Argument	SubCh		Beispiele (hier für Adresse 5)	Beispiel-Antwort	Erläuterung
VAL	8	8	Byte	VAL 8=2!, 5:8=2!		SPDIF Sampling-Format 0=Consumer 48kHz, 1=Consumer 96kHz, 2=Consumer 192kHz, 3=Professional 48kHz, (4=Professional 96kHz, 5=Professional 192kHz)
SMP	--	8	Byte	SMP=0!, 5:8=3!		SPDIF Sampling-Format 0=Consumer 48kHz, 1=Consumer 96kHz, 2=Consumer 192kHz, 3=Professional 48kHz, (4=Professional 96kHz, 5=Professional 192kHz)
RNG	--	19	0..8	RNG=1!, 5:19=1!	#5:255=OK, #5:19=2	Messbereichsumschaltung, RNG 0=-20dB ... RNG 8=+50dB
DSP*	9	89	Byte	DSP 9=4!, 5:89=2!	#5:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)
INL**	0..1	10..11	Float	INL?, INL 0?, 11?	#5:10=775	Level von True-RMS-Messgleichrichter in mVeff, VAL 10=Links, VAL 11=Rechts, skaliert mit Gain/Range
DSP*	0	80	Byte	DSP=0!, 0:80=2!	#5:255=0 [OK]	Panel-LCD angezeigte Menü-Auswahl, 0=AuxCmdsel, 1=RateSel, 2=GainSel, 3=LevelBarDispl, 4=mVDispl
DSP*	9	89	Byte	DSP 9=4!, 5:89=2!	#5:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)
OPT*	0..1	150..151	0..8	OPT 1=1!	#5:255=0 [OK]	OPT 0 bzw. VAL150=Default RNG, 1 bzw. 151=Default SPDIF-Rate
SCL*	0..7	200	Integer	SCL 0=2098!, 5:201=633!	#5:255=0 [OK]	Skalierungen für Pegel-Anzeige, Integer-Multiplikator, ca. 2100 bei geraden, 663 bei ungeraden Gains
RAW**	0, 1	50, 51	Integer	RAW 0?	#5:50=978	AD-Raw-Wert, RAW 0 bzw. VAL 50=links, RAW 1 bzw. VAL 51=rechts
ALL	0	99		ALL?, 5:99?	(Liste mit Messwerten)	Beide Input-Level von True-RMS-Messgleichrichter in mVeff, 10=Links, 11=Rechts

* nichtflüchtige EEPROM-Werte mit Schreibsperre
** nur Lesen

DIV 3.04

Cmd	Argument	SubCh		Beispiele (hier für Adresse 3)	Beispiel-Antwort	Erläuterung
VAL	0	0		3:VAL 0?, 0?	#3:0=1.564, #3:0=-99999	ADC24, Auslesen des gewandelten, skalierten Wertes, je nach Bereich DC oder AC TrueRMS. Overload: -99999
VAL	1	1		VAL 1?, 1?	#3:1=1.564	Intergrierter Messwert
VAL	2	2		VAL 2?, 2?	#3:2=1.564	Langsam integrierter Messwert
VAL	10	10		VAL 10?, 10?	#3:10=1.560	ADC10, schnellerer Interner 10-Bit-Wandler, je nach Bereich AC TrueRMS oder DC
VAL	11	11		VAL 11?, 11?	#3:11=1.560	ADC10, schnellerer Interner 10-Bit-Wandler, je nach Bereich AC Peak oder DC
RNG	--	19	0..15	RNG=1!	#3:255=OK, #3:19=2	Messbereichsumschaltung DC250mV..DC250V (0..3), AC250mV..AC250V (4..7), DC25mA..DC10A (8..11), AC25mA..AC10A (12..15)
RAW**	0, 10..12	50, 60..62		RAW 0?, 3:50?	#3:50=61245	DA-Wandler Rohdaten, 50=ADC24, 60=ADC10 AC TrueRMS, 61=ADC10 AC Peak, 62=DC
DSP*	8	88	Byte	DSP 8=0!, 0:88=2!	#3:255=0 [OK]	LCD-Anzeige Messwert-Integrationsmodus 0=Aus/1=schnell/2=langsam
DSP*	9	89	Byte	DSP 9=4!, 0:89=2!	#3:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)
OFS*	0..15	100..104		OFS 0=9277, 100=9277, 100?	#3:255=OK, #3:100=12345	Offsets für 16 Ranges DC250mV..AC10A von Wandler ADC24
OFS*	20.35	120..135		OFS 20=-7, 120=-7	#3:255=OK	Offsets für 16 Ranges DC250mV..AC10A von Wandler ADC10 (Atmega-Intern)
OPT*	0	150	0..15			Default-Range TO BE IMPLEMENTED
SCL*	0..15	200..215		SCL 0=1.00032!, 200=1.00032!, 200?		Skalierungen für 16 Ranges DC250mV..AC10A von Wandler ADC24
SCL*	20..35	220..235		SCL 20=1.00032!, 220=1.00032!, 220?		Skalierungen für 16 Ranges DC250mV..AC10A von Wandler ADC10 (Atmega-Intern)
TRM*	0	240	Byte	TRM 0=7!, TRM?	#3:255=0 [OK], #3:240=7	Trigger-Maske, Bit 0=SubCh 0, Bit 1=SubCh10, Bit2=SubCh11
TRT*	--	247	0, 20..31767	TRT=0, 0:247=2000	#3:255=0 [OK]	Autom. Trigger-Timing in ms, 0=Aus, Werte ab 20 ms möglich
TRL*	0	248	0..1	TRL=1!, TRL?, 248=0!, 248?	#3:255=0 [OK], #3:248=1	Trigger-Edge PB2, neg. = 0, pos. = 1, Impuls an diesem Pin liefert Messwerte wie in Trigger-Maske definiert. Erfordert Reset
TRG	--	249	--	TRG!, TRG?, 249?	#3:255=0 [OK] (Liste mit Werten)	manuelle Trigger-Auslösung, liefert Messwerte wie in Trigger-Maske definiert
ALL	0	95..99		ALL?, 99?	(Liste mit Messwerten)	TO BE IMPLEMENTED

* nichtflüchtige EEPROM-Werte mit Schreibsperre
** nur Lesen

EDL 1.78

(Erweiterungen von mcb in rot)

Cmd	Argument	SubCh		Beispiele (hier für Adresse 6)	Beispiel-Antwort	Erläuterung	
VAL	0..255	0..255	Float	6:VAL 0=10.0!, 0?, 6:1=0!	#6:255=0 [OK], #6:2=10.1	Allg. Form für SubCh	
TMP**	0	233	Float	6:TMP?, 233?	#6:233=46	Kühlkörper-Temperatur in °C, bei externer Leistungsstufe: abhängig vom gewählten A-Bereich	
ENA	--	0	Boolean	6:ENA=1!, 0?	#6:255=0 [OK], #1:2=0.0		
DCA	--	1	Float	DCA=100!, 1=20!		Strom I SOLL in A	
DCA	1	2	Float	DCA 1=100!, 2=20!		Strom I SOLL in mA	
DCP	--	3	Float	DCP=12!		Leistung P SOLL in Watt	
DCV	--	4	Float	DCV=10.5!		Untere Abschaltspannung U in Volt	
DCR	--	5	Float	DCR 1=123!, 5=123!		Sollwiderstand im R-Betrieb	
MAH	--	7	Float	MAH?, 7?, MAH=0!		Messwert kumulierter Strom in Ah (wird durch MAH=0! , MWH=0! auf 0 gesetzt)	
MWH	--	8	Float	MWH?, 8? MWH=0!		Messwert kumulierte Leistung in Wh (wird durch MAH=0! , MWH=0! auf 0 gesetzt)	
--	--	9	Byte	6:9=4!, 9?		Bereichs-Direktwahl 0 (kleinster Bereich) bis 3 (größter Bereich), 4 = Auto-Select je nach eingestelltem DCA (default)	
MSV**	0	10	Float	MSV?, 6:10?		Messwert U IST in V während On-Zeit	
MSA**	2	16	Float	MSA?, 6:11?		Messwert I IST in A während On-Zeit	
MSA**	1	12	Float	MSA 1?, 6:12?		Messwert I IST in mA während On-Zeit	
MSV**	5	15	Float	MSV 5?, 6:15?		Messwert U IST in Volt während Off-Zeit	
MSA**	5	16	Float	MSA 5?, 6:16?		Messwert I IST in A während Off-Zeit	
MSA**	6	17	Float	MSA 6?, 6:17?		Messwert I IST in mA während Off-Zeit	
RNG	--	19	Byte	RNG=1!, 19?		Eingangsspannungs-Messbereich und Modus, 0=OutputOff, 1=IconstHiVolt, 2=IconstLoVolt, 3=RmodeHiVolt, 4=RmodeLoVolt, 5=PmodeHiVolt, 6=PmodeLoVolt	
MSW**	--	18	Float	MSW?, 18?		tatsächlich aufgenommene Leistung in Watt	
PCA	--	21	Float	PCA=100!, 21=33.333!		Prozentwert für Ausgangsstrom, 0=Aus, 100=mit DCA eing. Wert	
RON	--	27	Integer	RON=10!, 27=4!		Ripple On-Zeit in ms, Iaus=DCA	
ROF	--	28	Integer	ROF=6!		Ripple Off-Zeit in ms, Iaus=DCA* (RIP)/100	
RIP	--	29	Integer	RIP=25!, 29=50!		Strom während Off-Zeit in Prozent von DCA-Wert (einschl. PCA)	
DSP	0	80	Byte	DSP=4!, 6:80=0!, 80?	#6:255=0 [OK], #6:80=3	Angezeigtes Menü auf PM8-Panel 0..5, 0 oder 1=I/U, 2=Mode, 3=T on, 4=T off, 5=I off%, 6=TrackCh	
DSP*	9	89	Byte	DSP 9=4!, 6:89=2!	#6:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)	
OFS*	2..5	102..105	Integer	OFS 2=5!, 102=4!		Offset Strom-DAC (Sollwert), 2=2mA, 3=20mA, 4=200mA, 5=2A-Bereich	
OFS*	10, 11	110, 115	Integer	OFS 10=0!		Offset Spannungs-ADC (Istwert), 10=Low-Bereich, 11=High-Bereich	
OFS*	12..15	112..115	Integer	OFS 12=0!		Offset Strom-ADC (Istwert), 12=2mA, 13=20mA, 14=200mA, 15=2A-Bereich	
SCL*	2..5	202..205	Float	SCL 2=0.976!, 202=0.976!		Skalierung Strom-DAC (Sollwert), 2=2mA, 3=20mA, 4=200mA, 5=2A-Bereich	
SCL*	10, 11	210, 215	Float	SCL 0=1.006!, 200=1.006!		Skalierung Spannungs-ADC (Istwert), 10=Low-Bereich, 11=High-Bereich	
SCL*	12..15	212..215	Float	SCL 2=0.976!, 202=0.976!		Skalierung Strom-ADC (Istwert), 12=2mA, 13=20mA, 14=200mA, 15=2A-Bereich	
ALL**	0	99		ALL?, 6:99?	(Liste mit Messwerten)	einschließlich Off-Werte	
RAW**	0..1	50..51	Integer	RAW 0?	#6:50=31548	Rohdaten direkt vom 16-Bit-A/D-Wandler ohne Skalierung und Offset	
RAW**	3..4	52..53	Integer	RAW 3?	#6:53=675	Rohdaten direkt vom internen A/D-Wandler (2 Kanäle Istspannung, Iststrom integriert)	**Default-Werte**
RAW**	20..21	70..71	Integer	RAW 21?	#6:71=32767	Rohdaten, die D/A-Wandler nach Skalierung und Offset erhält, in 70 für On-Zeit, in 71 für Off-Zeit	**12 Bit**
TRM*	0	240	Byte	TRM 0=1!, TRM=1,TRM?	#6:255=0 [OK], #6:240=2	Trigger-Maske, Bit 0=1 -> Freigabe Trigger Input PL6, Bit 1=1 -> Periodische Ausgabe On- und Off-Werte	00000001
OPT*	1	151	Float	OPT 1=0.1!	#6:255=0 [OK]	Default-Ausgangsstrom	0.02
OPT*	2	152	Float			InitLowDividerU, U-Gain U12 einschl. Istspannungs-Vorteiler wenn PreAmp ein (PC5/Q1=high)	2.5
OPT*	3	153	Float			InitHiDividerU, Istspannungs-Vorteilfaktor (R53+R58)/R52+1	10
OPT*	4	154	Float			InitGainI, Gain Sense-OpAmp U13	0.5
OPT*	5	155	Float			Uref LT1019 =2,5000V	2.5
OPT*	6	156	Float			maximal aufzunehmende Leistung (U*I oder U*U/R), darüber Abschaltung und Fault-Meldung	25
OPT*	7	157	Float			R sense 1. Bereich nom. 2mA, 100 Ohm oder 10mA, 20 Ohm	100
OPT*	8	158	Float			R sense 2. Bereich nom. 20mA, 10 Ohm oder 100mA, 2 Ohm	10
OPT*	9	159	Float			R sense 3. Bereich nom. 200mA, 1 Ohm oder 1A, 0.22 Ohm	1
OPT*	10	160	Float	OPT 10=0.2197!		R sense 4. Bereich nom. 2A, 0.1 Ohm oder 10A, 0.02 Ohm	0.1
OPT*	11	161	Float			Imax A 1. Bereich	0.002
OPT*	12	162	Float			Imax A 2. Bereich	0.020
OPT*	13	163	Float			Imax A 3. Bereich	0.200
OPT*	14	164	Float	OPT 14=5.0!		Imax A 4. Bereich, für CheckLimits	2
OPT*	15	165	Float			UmaxHi maximale Klemmenspannung im HiVolt-Bereich	25
OPT*	16	166	Float			UmaxLo maximale Klemmenspannung im LoVolt-Bereich	6.1
OPT*	17	167	Float			Bit 0, 1 = DACtype, 0=MAX543/LTC8043, 1=AD5452, 3=DAC8811, Bit 2=LM75 vorhanden, Bit 3 = LM75 extern I2C-Adr. $48	4
OPT*	18	168	Float			Init Ipercent, Einschaltwert für Off-Strom in Prozent vom Sollstrom	0
OPT*	19	169	Float			Init PWonTime	10
OPT*	20	170	Float			Init PWoffTime	0
OPT*	21	171	Float	OPT 21=60!		Lüfter-Einschalt-Temperatur	50

FPGA 2.1

Cmd	Argument	SubCh		Beispiele (hier für Adresse 7)	Beispiel-Antwort	Erläuterung
VAL	0..79	0..79	LongInt	7:VAL 0=512!, 0?	#7:255=0 [OK], #7:2=0.1	Allg. Form für SubCh, 80 Schreib-/Lese-Register im FPGA über SPI, die ersten 4 werden ständig aktualisiert für PM8
DSP	0	80	Byte	DSP=4!, 7:80=0!, 80?	#7:255=0 [OK], #7:80=3	Angezeigtes Menü auf PM8-Panel TBD
HEX		88	Byte	HEX=1, 88=0!		Hex-Mode bei OptoBus-Zahlenausgeben, zusätzlich in der Form [$12345678]
DSP*	9	89	Byte	DSP 9=4!, 0:89=2!	#7:255=0 [OK]	Inkrementalgeber Impulse pro Rastpunkt (12er/24er haben meist 4, 30er/32er haben meist 2)
OPT*	0	150	Byte	OPT 0=1!, OPT=1	#7:255=0 [OK]	Default-BIN/BIT-Dateinummer (FPGA-Konfigurationsdatei), Default=0
OPT*	1	151	Byte	OPT 1=-1!		Default-INI-Dateinummer (Script-Datei)
OPT*	3	153	Byte	OPT 3=128!		Default-Auto-Increment-Register/SubCh AIR für MEM-und DAT-Files (siehe dort)
OPT*	4	154	Byte	OPT 4=0!		MCH-Default (siehe LabScript)
OPT*	5	155	Byte	OPT 5=10!		SCH-Default (siehe LabScript)
OPT*	7	157	Byte	OPT 7=255!		Nachkomma-Anzahl in der Antwort, Default 255=Auto
OPT*	9	159	Integer	OPT 9=500!		GET-/PUT-Timeout, Default 500 ms
OPT*	10..13	160..163	LongInt	OPT 10=32768!		Init-Werte für 4 angezeigte Param-Register 0..3
CLK		90..99				90..99 RTC-Register und Befehle (Echtzeituhr), Stunden/Minuten/Sekunden/Tag/Monat/Jahr
CFG	-	240	Byte	CFG=3!, CFG?	#7:255=0 [OK]	Auswahl und Laden der Datei Nummer <byte> von SD-Karte. Bei BIN- und BIT-Files wird die Datei gelesen und ins FPGA programmiert, INI-Dateien werden als Script ausgeführt. Achtung: OptoBus ist bei Konfigurationsdaten etwa 2 Sekunden blockiert!
LST**, DIR**	-	241	-	LST?, DIR?, 241?	#7:241=0 [<FileName1>], #7:241=1 [<FileName2>] usw.	Directory-Liste mit Dateinummern und zugehörigen Dateinamen
FNM**		242	-	FNM?, 7:242?		FileNum, Anzahl der Dateien auf der SD-Karte
						FPGA 1.0 behandelt Scripts (von SD-Karte gelesene INI-Dateien) im Prinzip so, als hätte es diese Befehle/Abfragen über den OptoBus erhalten.

Datei-Befehle neu ab Version 2.1

Alphanumerische Befehlsargumente für Dateinamen zulässig, die nicht mit Ziffern beginnen dürfen.

CFG	240	CFG=9, CFG=DDS.BIN		File Load, BIT/BIN-Konfigurationen, aber auch INI-, DAT-, MEM- und BMP-Dateien (werden je nach Suffix anders behandelt)
		CFG=BINDATEI.DAT		BIN- und BIT-Dateien werden als Konfiguration ins FPGA geladen, INI-Dateien werden als LabScript ausgeführt, MEM- und DAT-Dateien auf eine SPI-AutoIncrement-Adresse ausgegeben (z.B. für BlockRAM-Inhalte, DAT immer binär, byteweise!), BMP-Dateien als Background ins QVGA-Video-RAM geladen, sofern ein solches im FPGA implementiert ist.
		CFG=PICOBLAZ1.MEM		MEM-Dateien sind BlockRAM-Inhalte für den PicoBlaze-Core. Diese müssen dem von KCPSM3-Assembler ausgegebenen Format entsprechen (Textdatei mit einem Hex-Wert pro Zeile, max. 1024 Werte).
AIR	248	AIR=<n>, 248?, AIR=128	#7:248=128	Auto-Increment-Registernummer (für FPGA-SPI) für MEM- und DAT-Files (z.B. BlockRAM-Inhalte), Default: Register/SubCh 128 bzw. Inhalt OPT 3. Dieses Register ist das Daten-Empfangregister,**nach** jedem Zugriff muss sich ein Adresszähler im FPGA erhöhen. Ein Reset des Zählers erfolgt**vor und nach** jeder Datei-Übertragung durch automatischen Zugriff auf SPI-Register <n>+1 (Default: 129). Dieser Zugriff kann daher auch zum Reset des (PicoBlaze-)Cores dienen.

Befehle neu/geändert ab FPGA Version 2.5

Alphanumerische Befehlsargumente können nun auch in "Häkchen" gesetzt werden, somit sind z.B. auch Dateinamen zulässig, die mit Ziffern beginnen.
Eine batteriegepufferte externe Echtzeituhr DS1302 an PortB/Debug wird unterstützt, Pinbelegung siehe CORERAM-Karte

CLK	0..5	90..95		CLK 0=17, CLK 1?	#7:91=45	RTC-Register (Echtzeituhr), 90=Stunden, 91=Minuten, 92=Sekunden, 93=Tag, 94=Monat, 95=Jahr zweistellig. Schreiben setzt auch externe RTC, falls vorhanden
CLK	7	97			#7:97=0 [17:45:00 02.05.09]	AVR-Uhr mit externer RTC DS1302 synchronisieren (auslesen, wird beim Einschalten automatisch ausgeführt), Anzeige Uhrzeit/Datum Die interne Uhr des AVR-Controllers läuft auch ohne externe RTC, sollte dann aber nach dem Einschalten auf die/das richtige Uhrzeit/Datum gesetzt werden, damit angelegte Dateien das korrekte Erstellungsdatum erhalten.
OPT*	0	150	Byte	OPT 0=1!, OPT=1	#7:255=0 [OK]	obsolet wg. OPT 30, ohne Funktion
OPT*	1	151	Byte	OPT 1=-1!		obsolet wg. OPT 31
OPT*	10	160	Byte			Einschalt-Defaultwert für Core**Rx**SubCh Kommunikation FPGA-CPU-Core an AVR intern
OPT*	11	161	Byte			Einschalt-Defaultwert für Core**Tx**SubCh, Kommunikation AVR an FPGA-CPU-Core intern
OPT*	20..23	170..173	LongInt	OPT 20=32768!		Init-Werte für 4 angezeigte Param-Register 0..3, **Achtung: Neue Belegung!**
OPT*	30	180		OPT 30="PACMAN.INI", OPT 30="AMDDS.BIN"		Default-FPGA-Konfigurationsdatei oder INI-Script, statt OPT 0 und OPT 1 ausgeführt, falls nicht leer
OPT*	31	181		OPT 31="MYDATA.TXT"		Default-FileName für mit FWR und FWV angelegte/erweiterte Messdaten-Textdateien, default "DATAFILE.XLS"
FNA		243		FNA="<filename>"		FileName für mit FWR und FWV angelegte/erweiterte Messdaten-Textdateien temporär neu setzen, default der mit OPT 31 angelegte Dateiname.
FDL		244		FDL="<filename>"		FileDelete, beliebige Datei löschen
FQU		249		FQU="<filename>"	#7:255=31 [NOTFOUND] [NOCARD]	File Query, liefert 0 [OK] wenn File existiert, sonst Fehlermeldung

FWR		220	FWR = <register>, FWR=3		FileWriteRegister, LabScript-Registerinhalt 0..9 in (ggf. mit FNA bestimmte) Textdatei schreiben, wird automatisch erstellt (wenn nötig), Header angelegt, geöffnet und geschlossen, Registernummer und Uhrzeit werden ebenfalls geschrieben (TAB-getrennt).
FWV		230	FWV <index>=<wert>, FWV=1.23456, FWV 3=775.0		FileWriteValue, angegebenen Wert in (ggf. mit FNA bestimmte) Textdatei schreiben, wird automatisch erstellt (wenn nötig) geöffnet und geschlossen, Header angelegt. Index und Uhrzeit werden ebenfalls geschrieben (TAB-getrennt).
TTF		800	TTF 3="LISTING.TXT"		TeleType File an SPI, TTF <SPI-Register>="<filename>", für CPU-Cores, Wartezeit 50 ms nach jedem <CR/LF>
TTY		870	TTY ="<filename>" oder TTY=<filenumber>	Inhalt der Datei als Klartext	TeleType File an serielle Schnittstelle, TTY ="<filename>". Funktioniert i.d.R. nur, wenn FPGA-Modul das letzte (oder einzige) in der OptoBus-Kette ist, ansonsten werden die Zeilen von folgenden Moduln möglicherweise als Befehle interpretiert. Wartezeit 10 ms nach jedem <CR/LF>
TSF		880	TSF="<filename>"		Text Save (append) to File, TSF="<filename>" <CR>, <bieliebiger Text> , Ende mit ETX ($03), für CPU-Cores
XMR		890	XMR?		XModem-Receive AutoInc-Block vom Rechner an mit AIR bestimmtes SPI-Register, vorher müssen AIS, AIW und ggf AIR entsprechend gesetzt sein. Nur XMODEM 128/Checksum unterstützt, nicht CRC oder 1K. Funktioniert nur, wenn FPGA-Modul das **einzige** in der OptoBus-Kette ist; andere Module leiten Binärdateien nicht durch!
TSR		900	TSR <SPI-Register>="<string>", TSR 3="RUN 50000"		Type String an SPI, für CPU-Cores mit eigener Intelligenz (z.B. ct-BASIC)
TSS		970	TSS ="<string>"	<string>	Type String an serielle Schnittstelle, für Anwendung innerhalb von INI-Scripten und Debug-Zwecke
COM	0	990	COM 0=1		Einstellen des Core**Rx**SubCh-SPI-Register, F_INT-IRQ holt Zeichen aus diesem Register ab und interpretiert Zeichenkette als c't-Lab-Befeh
COM	1	991	COM 1=2		Einstellen des Core**Tx**SubCh-SPI-Register, für Zeichenausgabe an CPU-Cores mit eigener Intelligenz (z.B. ct-BASIC). Anfragen/Befehle, die über F_INT/CoreRxSubCh hereinkommen, werden hier beantwortet.
COM	2	992	COM 2=0		Default-Kommunikationskanal für INI-Files, 0=OptoBus (default), 1=intern COM an FPGA
AIR		280	AIR=<n>, 280?, AIR=128, 280=128	#7:280=128	Auto-Increment-Registernummer (FPGA-SPI-SubCh) für MEM- und DAT-Files (z.B. BlockRAM-Inhalte) und BLD/BSV-Befehle, Default: Register/SubCh 128 bzw. Inhalt OPT 3. Dieses Register ist das Daten-Empfangregister, **nach** jedem Zugriff muss sich ein Adresszähler im FPGA erhöhen. Ein Reset des Zählers erfolgt **vor und nach** jeder Datei-Übertragung durch automatischen Zugriff auf SPI-Register <n>+1 (Default: 129). Dieser Zugriff kann daher auch zum Reset des (PicoBlaze-)Cores dienen. **Achtung: Neue SubCh-Belegung** Die dem AIR-Wert folgenden 3 SPI-Register sind für AutoIncrement-Übertragungen reserviert und werden bei Blockübertragungen (mit BSV/BLD) automatisch beschrieben. Wenn AIR=128 (default), gil folgende Belegung: 129 = CoreSelect-Register, selektiert bestimmten Core, falls mehrere im FPGA vorhanden 130 = Startadresse in 32-Bit-Breite, Beschreiben dieses Registers muss im FPGA **Schreib**zugriff auf RAM-Inhalt freischalten 131 = Startadresse in 32-Bit-Breite, Beschreiben dieses Registers muss im FPGA den **Lese**zugriff auf RAM-Inhalt freischalten
AIS		281	AIS=0		Auto-Increment-Select, bestimmten Core auswählen für DAT, MEM, HEX-Files, wird auf AIR-Wert plus 1 ausgegeben (default 129).
AIW		282	AIW=1		Auto-Increment-Wortbreite 1, 2 oder 4 Bytes
BLD		283	BLD="<filename>", BLD=<filenummer>		AutoInc-Block von SD-Card lesen und auf AIR ausgeben, ähnlich CFG=<datfile>.DAT, nur mit dem Unterschied, dass **Datei-Extension nicht interpretiert** wird
BSV		284	BSV="<filename>", BSV=<filenummer>		AutoInc-Block von AIR lesen und auf SD-Card speichen
AIM		285	AIM=<adresse>, AIM=49152, AIM=0		AutoInc-Blockstart in LongWords, für BSV und BLD, wird automatisch vor Übertragung auf SPI-Register 130 oder 131 ausgegeben
AIE		286	AIE=<adresse>, AIE=65535		AutoInc-Blockende in LongWords, für BSV, Anzahl der gelesenen Bytes = AIE-Wert minus AIS-Wert. Wird nicht auf SPI ausgegeben, sondern dient als Endpunkt für BSV-Befehl beim Hochzählen der Adresse.

* nichtflüchtige EEPROM-Werte mit Schreibsperre
** nur Lesen

LabScript 1.0

Script-Implementation für FPGA ab 1.2

INI-Files sind auf der SD-Karte gespeicherte Textdateien, die beliebige c't-Lab-Befehle enthalten dürfen. Diese werden abgearbeitet, als wären sie vom PC über die OptoBus-Schnittstelle an das Modul gelangt. Zu erstellen mit Wordpad o.dgl.

Für Scripte, die als INI-Datei auf SD-Karte gespeichert sind

Achtung: Script-Befehle sind nicht rekursiv, deshalb sind CFG, LST und FNM in einem Script nicht erlaubt (es sei denn, sie sprechen ein anderes Modul an). Branches "vorwärts" verlangsamen den ersten Durchlauf, und zwar um so mehr, je weiter "unten" sie im Script stehen.

Erweiterung auf Integer-SubCh 0..32767
Für die Script-Verarbeitung stehen 10 Fließkomma-Register zur Verfügung, von denen das erste (0) als Fließkomma-Akkumulator dient.
Load- und Store-Befehle funktionieren mit jedem Register (0..9 einschl. Akkus)

Cmd	Argument	SubCh <opcode>	<format>	Syntax/Beispiele	Beispiel-Antwort	Erläuterung
//	-	-	-	// Kommentarzeile		Ignorierte Kommentarzeile
REG		300..309	Float	REG <n>=<wert>, REG <n>?		Load Register 0..9 immediately mit Wert, auch Abfrage
ACC		300		ACC=<wert>, ACC?, 300?		Load Akku A (Register 0) immediately mit Wert, auch Abfrage
MOV		310..319		MOV <zielregister>=<quellregister>		Move Registerinhalt <zielregister> <= <quellregister>
DEC		320..329		DEC <register>		320 Decrement Register (0..9), Ergebnis für Branches merken
INC		330..339		INC <register>		330 Increment Register (0..9), Ergebnis für Branches merken
CPZ		340..349		CPZ <register>		340 Compare Register (0..9) mit "0", Ergebnis für Branches merken
XCH		350..359		XCH <register>=<register>, XCH = 4, XCH 2=6		350 Exchange 0..9 <=> 0..9, ohne Argument: Akku A mit Register (0..9)
MUL		600..609		MUL <register>		600 Multiplikation Akku A = Akku A * Register (0..9)
DIV		610..619		DIV <register>		610 Division Akku A = Akku A / Register (0..9)
ADD		620..629		ADD <register>		620 Addition Akku A = Akku A + Register (0..9)
SUB		630..639		SUB <register>		630 Subtraktion Akku A = Akku A - Register (0..9)
SQR		640..649		SQR <register>		640 Quadratwurzel Register (0..9) <= sqr(Register (0..9))
SQU		650..659		SQU <register>		650 Quadrat Register (0..9) <= Register (0..9) * Register (0..9)
NEG		660..669		NEG <register>		660 Negiere Register (*-1)
GET		400..409		GET <zielregister>?		Warte auf Ergebnis, speichere in Registerinhalt 0..9. Funktioniert wegen nötigem OptoBus-"Kreisverkehr" nur unzureichend oder gar nicht. (#X:XXX gleich Kanal-Match)
PUT		500..509		PUT <register>		Sende Wert in Register auf MCM/SCM
FWR		900..999		FWR <dateinummer> = <register>, FWR 75=2!, FWR = 0!	angehängt an Datei DATA75.XLS, DATA0.XLS	FileWrite, Register 0..9 in Datei DATA0..DATA99.XLS schreiben (reine Textdatei, kann mit Excel geöffnet werden)
LBL		1000..1099		LBL <labelnr>		Label 0..99 setzen (hier begrenzt auf 0..31)
GTO		1100..1199		GTO <labelnr>		Goto <labelnr>
BRA		1100..1199		BRA <labelnr>		Branch always to <labelnr>, wie GTO
BRG		1200..1299		BRG <labelnr>		Branch if greater 0 to Label <labelnr>, wenn Akku A größer als B
BGE		1300..1399		BGE <labelnr>		Branch if greater or equal 0 to Label <labelnr>, wenn Akku A größer oder gleich B
BEQ		1400..1499		BEQ <labelnr>		Branch if equal 0 to Label <labelnr>, wenn Akku A gleich B
BLE		1500..1599		BLE <labelnr>		Branch if lower or equal 0 to Label <labelnr>, wenn Akku A kleiner oder gleich B
BRL		1600..1699		BRL <labelnr>		Branch if lower 0 to Label <labelnr>, wenn Akku A kleiner als B
INP		2000..2255		INP <SubCh>?, INP 248? INP 0?		Akku (Reg. 0) Input, mit Ergebnis von moduleigenem SubCh 0..255 füllen
OUT		2000..2255		OUT <SubCh>=<register>, OUT 128=3		Register Output, Ausgabe <register> auf moduleigenen SubCh 0..255
WTH		290		WTH		Wait Tick Hour
WTM		291		WTM		Wait Tick Minute
WTS		292		WTS		Wait Tick Second, warte auf nächsten Sekundenwechsel
DLY		299		DLY=<ms_zeit>, 299=1500!		Delay/Pause, Ausführung wird <ms_zeit> Millisekunden angehalten. Nur in Scripts sinnvoll, da OptoBus während dieser Zeit blockiert ist. Liefert beim Lesen letzten Pausenwert.
MCH		270		MCM=<Moduladresse>		Main-Channel für GET/PUT, Script-gesendete/empfangene Daten, Moduladresse
SCH		271		SCM=<SubCh>		SubChannel für GET/PUT, Script-gesendete/empfangene Daten

Display-Befehle QVGA-Treiber FPGA (nur bis Firmware #2.33)				**obsolet ab FW 2.4**
DPV	700		DPV=1.2345	Display Value Immediate, Wert DPV=1.2345 @ DPX, DPY
DPR	710..719			Display Inhalt Register 0..9 DPR=<register> @ DPX, DPY
DPS	720		DPS=HALLO_FPGA	Display String DPS=HALLO_FPGA @ DPX, DPY
DPC	730			Display RTC, DPC=0 Zeit, DPC=1 Datum @ DPX, DPY
DRW	740..742			740 DrawRect 741 FillRect 742 Drawline mit XY-Param @ DPX, DPY
DRI	750		DRI=1	Draw Instrument, Bargraph usw.
DBG	770		DBG=0, DBG=demo.bmp	Display-Background, DBG=0 keiner, DBG=<alphastr>=BMP-Name
DPX	780		DPX=80	Display-Koordinaten, EndX, vorheriger Wert wird StartX für Linien/Rechtecke
DPY	781		DPY=200	Display-Koordinaten, EndY, vorheriger Wert wird StartY
DFX	782		DFX=1	Font-Skalierung X-Richtung (vertikal), 1=klein, 2=mittel, 3=groß
DFY	783			Font-Skalierung Y (horizontal)
DCO	790		DCO=42	Block-Farbe 6-Bit-Darstellung für nächste Anzeige-Operation, Bitfolge RRGGBB

Wünsche und Anregungen bitte an cm@ctmagazin.de