DeviceLib

"Der Schlüssel zur intelligenten Gerätevernetzung."

Ekon GmbH Weidacher Hartwig

T: 0474 551820 M: 3483804799

wh@my-gekko.com

V1.0 Stand 04.03.2025

connects people, buildings and technology.

Inhalt

Ziel

Die Device Description Files (DDF) bilden die Grundlage einer strukturierten, einheitlichen Gerätedatenbank. Sie beschreiben Geräteprotokolle, Kommunikationsbefehle, Parameterstrukturen und logische Variablen für eine standardisierte Anbindung externer Geräte.

Das Konzept einer DDF-basierten Gerätedatenbank verfolgt folgende Ziele:

- **Protokollunabhängigkeit:** Ein einheitliches Modell für verschiedene Protokolle wie Modbus, REST, oder ASCII/SERIAL.
- **Wiederverwendbarkeit:** Einmal erstellte DDFs können beliebig vielen Geräten oder Projekten zugeordnet werden.
- **Erweiterbarkeit:** Durch die flexible Struktur lassen sich neue Felder, Protokolle und Funktionen leicht integrieren.
- **Automatische Interpretation:** Systeme können Geräteinformationen, Werte und Befehle direkt aus der DDF-Datei interpretieren, ohne Anpassung des Programmcodes.
- **Wartbarkeit und Transparenz:** Änderungen an der Kommunikation oder Geräteparametern erfolgen ausschließlich durch die Anpassung der DDF-Datei.

Eine vollständige DDF-Datei umfasst typischerweise:

- Allgemeine Geräteeigenschaften (*GENERAL)
- Datenabfragen und -schreibbefehle (*READ, *WRITE)
- Logische Abbildung der verfügbaren Variablen (*ITEM)
- Dynamische Parameter für URL/Telegrammsteuerung (*ARGS)
- Eingangsverarbeitung von externen REST-Events (*LISTENER)
- Optional Gruppierung von Funktionen (*GROUP)

Durch diese klare Trennung von Gerätedefinition und Kommunikationslogik wird eine hohe Flexibilität und langfristige Zukunftssicherheit gewährleistet.

Konzept der DDF Datei

DDF-Datei

- *GENERAL → Gerätedaten
- *READ / *WRITE → Kommunikationsbefehle
- *ITEM → Logische Zuordnung der Variablen
- *ARGS → Dynamische Argumente
- *LISTENER → Empfang externer Ereignisse

⇓ wird eingelesen ⇓

System

- Kommunikation (Modbus, REST, SERIAL)
- Verarbeitung der Werte nach FORMULA / RFORMULA
- Bereitstellung als Variablen im Steuerungssystem
- Bereitstellung eines Konnektor-Interfaces mittels ALIAS-Namen zur automatischen Kopplung von Funktionsbausteinen
- Event-Auswertung bei REST (Listener)

Konzept des DEVICEDB-Portals

Erstellte DDF-Dateien können zentral in die Gerätedatenbank unter https://devicedb.my-gekko.com hochgeladen werden. Dort erhalten sie eine eindeutige ID und stehen für Kunden und Techniker zum Download bereit.

Workflow zur Erstellung und Bereitstellung einer DDF-Datei

1. **Erstellen einer neuen Gerätebeschreibung inkl. Test mit realem Gerät:**
   Entwicklung und Überprüfung der DDF-Datei im Testaufbau mit dem realen Endgerät.
2. **Anlegen des Geräteprofils in der Gerätedatenbank:**
   Erfassung aller relevanten Gerätedaten (Hersteller, Modell, Protokoll etc.).
3. **Upload der DDF-Datei und eventueller Zusatzinformationen:**
   Hochladen der geprüften DDF-Datei und optionaler ergänzender Dokumente (z. B. technische Handbücher, Bilder).
4. **Freigeben des Geräteprofils:**
   Überprüfung und endgültige Freigabe des Profils für die allgemeine Nutzung.
5. **Downloaden der DDF und Upload auf das Steuerungssystem:**
   Techniker/Kunden laden die geprüfte DDF herunter und integrieren diese in ihre Systeme.

Dadurch wird sichergestellt, dass alle Systeme auf konsistente, aktuelle und geprüfte Gerätedefinitionen zugreifen können.

DDF - Device Definition File

Aufbau:

Das DeviceDefinitionFile (DDF) ermöglicht die Protokollbeschreibung des anzubindenden Gerätes.

Der strukturelle Aufbau ist für alle Protokollvarianten identisch. Die Inhalte der einzelnen Bereiche können sich jedoch unterscheiden.

Der Aufbau ist eine ; getrennte CSV-Tabelle. Da teilweise Zelleninhalte auch ; enthalten müssen solche Zellen mit „ „ eingerahmt werden. Zb. Formeln.

Der Inhalt gliedert sich in:

**GENERAL,*PREPROCESS(*READ),*ONCHANGE(*WRITE),*LISTENER,*GROUP,*ITEM*

In Klammer alternative Namen.

Die Bereiche sind als Tabellen aufgebaut, wobei die Zeile mit dem Bereichsnamen in den Spalten die Tabellen-Spaltennamen enthalten.

Die Bereiche *PREPROCESS(*READ),*ONCHANGE(*WRITE),*LISTENER können auch einen Unterbereich Namens *ARGS beinhalten.

Die Bereiche haben in der 1.Spalte den Bereichsnamen mit *NAME. Bei allen anderen Einträgen muss die 1.Spalte leer. Auch bei Unterbereich *ARGS fängt dieser in der 2.Spalte an.

Der Bereich *GENERAL beinhaltet eine Liste der Variablen/Eigenschaften mit dem Namen in der 2.Spalte und den Wert in der 3.Spalte.

Beispiel.

*GENERAL		
	DEVICE	CAMERA
	MANUFACTURER	GEKKO
	TYPE	Automation Control System

Alle anderen Bereich sind wie folgt aufbebaut. Die Spaltennamen und Inhalte usw. variieren.

*WRITE	ALIAS	METHOD	URL	DATATYPE		

	WRITE	GET	api/v1/var/alarms	JSON		
	*ARGS	METHOD	ALIAS	TYPE		
	ARGS	WRITE	VALUE	arg		
	ARGS	WRITE		arg		
	ARGS	WRITE		arg		
*ITEM	ALIAS	NAME	ID	VISIBILITY	RFORMULA	POLLING
	TEST	VMD	0		"X.0:=DATA.FIND.VMD.EventNotificationAlert.eventType && (DATA.T > ($.SYS.TIME-2));"	500

Allgemein:

GENERAL

In dem Abschnitt sind die verschiedenen Parameter des Gerätes definiert:

DEVICE: Gerätename

MANUFACTURER: Hersteller

PROTOCOL: Protocoll (SERCOM, MODBUS TCP,MODBUS ASCII,MODBUS RTU,REST)

MODEL_NR: Modelnummer des Gerätes

VERSION_NR: Versionsnummer des Gerätes

Die Kombination zwischen Protokoll, Model und Version sind je Hersteller eindeutig/einmalig und ergibt die ID des Gerätes.

MIN_CONTROL_VERSION: Mindest Gekko/Controllerversion

TIMESTAMP: Erstellungsdatum

Diese Einträge werden seitens der Datenbank generiert.

Die weiteren Parameter hängen vom Protokolltyp ab.

Kommunikationsparameter:

Allgemein

CONNECTION: Verbindungstyp

SERIALPORT: Zuweisbarer ConrollerPort

SERIALPORT+: Zuweisbarer Port ink. AUX und NODE

IPADDRESS: Eingebbare IP-Adresse

DOMAIN Eingebbare Domain

DEFDOMAIN: Feste Domain

Protokoll REST:

DOMAIN: Eingebbare Domainadresse (zb. http:/das/, oder IP je nach Gerät und Verwendung

AUTHENTIFICATION: Authentifizierungstyp

NONE

PASSWORD

DEFPASSWORD: Password im DDF

PASSWORD: Password bei DEFPASSORD

USER: User bei DEFPASSORD

SLAVE: Standard-Slaveadresse für 1.Device

LISTENER_PORT TCP-Port für Socketserver

SLAVESMAX: Anzahl maximale Slaves/Devices, welche über UI angegeben werden können. Fehlt der Paramter ist nur 1 Device/Slave.

Bei **SLAVESMAX>0** wird auf dem UI das Eingabefeld Slaves angezeigt. Dort können mit Komma(,) getrennt die Slaves ID angegeben werden.

Parallel dazu können bei Domain die einzelnen Devices der Station verschiedenen Domains zugewiesen werden. Die IP-Adressen können auch mit – im letzten Bereich von bis angegeben werden. Zb. 192.168.1-12 legt 12 Geräte an.

Zb. UI-Eingabe: DOMAIN: 192.168.1,192.168.1.12-16,192.168.1.19

SLAVES: CAM1,CAM1,CAM3

Die Slaveadresse wird als Zusatzinfo im Devicenamen angezeigt.

Die maximale Anzahl zwischen Slave-Liste und Domain-Liste bestimmt die Anzahl der Geräte. Begrenzt mit Slavesmax aus dem DDF. Fehlt der Domain, dann wir der vorherige genommen.

Zb. UI-Eingabe: DOMAIN: http://192.168.1.2/

SLAVES: CAM1,CAM1,CAM3

Alle Geräte werden mit derselben Domain angelegt. Diese hat in Kombination mit LISTENER_PORT keinen Sinn. Jedoch wenn ich unterschiedliche REST-Abfragen pro Device desselben Types machen möchte, wie beim RestInterface von GEKKO schon. Zb. Licht1, Licht2,...

Protokoll Modbus:

PORT	Modbus Port. Default = 502
SLAVE:	Standard-Slaveadresse für 1.Device
SLAVESMAX:	Anzahl maximale Slaves/Devices, welche über UI angegeben werden können.(1..)
BAUDRATE:	Baudrate
DATABITS	Databits
STOPBITS	Stopbits
PARITY:	NONE,ODD,EVEN
FLOWCONTROL:	NO,SW,HW

TIMEOUT	Timeout in ms
WAIT_AFTER_CONNECT:	Zeit in sekunden
MIN_PAUSE:	Mindestpause in ms
WORDORDER:	BIG_ENDIAN,LITTEL_ENDIAN
BYTEORDER:	BIG_ENDIAN,LITTEL_ENDIAN
ERROR_HANDLING	FLUSH_OR_RECONNECT_ON_ERROR, NOP_ON_ERROR, FLUSH_OR_CLOSE_ON_ERROR

Protokoll Serial:

SLAVESMAX:	Anzahl maximale Slaves/Devices, welche über UI angegeben werden können.
SLAVE:	Standard-Slaveadresse für 1.Device
BAUDRATE:	Baudrate
DATABITS	Databits
STOPBITS	Stopbits
PARITY:	NONE,ODD,EVEN
FLOWCONTROL:	NO,SW,HW
WAIT_AFTER_SEND:	Warte Zeit in sekunden nach senden
MIN_PAUSE:	Mindestpause in ms
WORDORDER:	BIG_ENDIAN,LITTEL_ENDIAN (nur bei Byte-Telegramm)
BYTEORDER:	BIG_ENDIAN,LITTEL_ENDIAN(nur bei Byte-Telegramm)

Beispiel:

*GENERAL		
	DEVICE	Brink Excellent HRV
	MANUFATURER	
	TYPE	Ventilation unit
	PROTOCOL	MODBUS RTU(new)
	MODEL_NR	1
	VERSION_NR	1
	ID	0x0B00004500010000
	MIN_CONTROL_VERSION	
	TIMESTAMP	
	CONNECTION	SERIALPORT+
	PORT	502
	SLAVE	1
	TIMEOUT	5000
	WAIT_AFTER_CONNECT	2
	MIN_PAUSE	50
	WORDORDER	BIG_ENDIAN
	BYTEORDER	LITTLE_ENDIAN
	ERROR_HANDLING	FLUSH_OR_RECONNECT_ON_ERROR
	DEFAULT_ALARMING	

Kommunikationsmethoden:

LISTENER

Kommunikationsmethoden welche bei Socketverbindungen ausgeführt werden.

PREPROCESS (READ)

Kommunikationsmethoden welche bei im Intervall ausgeführt werden.

ONCHANGE (WRITE)

Kommunikationsmethoden welche bei Änderung/Force ausgeführt werden. Ohne Polling intervall.

Der Aufbau ist für alle identisch je nach Protokoll. Spaltennamen unterscheiden sich teilweise.

Protokoll Sercom:

ALIAS Eindeutiger ALIAS für Zugriff auf die Variablen

SEND Sendetelegramm

SEND_DELIMITER_START Startkennung als Text oder Hex-Code (/000203)

SEND_DELIMITER_END Endkennung als Text oder Hex-Code (/000203)

RECEIVE_LEN Länge Empfangs-/oder Antworttelegramm

DELIMITER_START Startkennung als Text oder Hex-Code (/000203)

DELIMITER_END Endkennung als Text oder Hex-Code (/000203)

POLLING Abfragezyklus in ms

Protokoll Modbus:

ALIAS	Eindeutiger ALIAS für Zugriff auf die Variablen
FUNCTION	FC3,FC4,… FC1..16 (Modbusfunktionen)
SLAVE	Slaveadresse (Zahl oder Variable) (optional)
ADDRESS	Startadresse
COUNT	Anzahl Register
POLLING	Abfragezyklus in ms

Protokoll Rest:

ALIAS	Eindeutiger ALIAS für Zugriff auf die Variablen
METHOD	GET/POST/LISTEN
URL	Url
	Zusammengesetzte Url für Abfrage= Url + DOMAIN
DATATYPE	JSON/XML
POLLING	Abfragezyklus in ms

Zusatzargumente für REST/SERCOM

***ARGS**	Kennzeichner (ARG)
METHOD	ALIAS der zuzuweisenden Funktion , *ALL für Alle
ALIAS	ALIAS des Argumentes**: Ist der Alias leer wird der Namen verwendet.**
TYPE	data, Nur bei REST zusätzlich (url,arg)
NAME	Bei SER: Formatierung zb. %02lldTR1 Bei REST: Name des Eintrages.
FORMAT	Formatierung (STRING,DEZIMAL, DOUBLE, T_YYYYMMDDHHMM) DAY Zeitangabe als Tag. 2025-01-30 ISO8601 Zeitangabe auf Basis einer time_t Values. T_YYYYMMDDHHMM Zeitangabe auf Basis einer YYYYMMDDHHMMSS DINT WERTES. STRING in NAME %s DEZIMAL in NAME %lld DOUBLE in Name %d ENUM:val1=CODE1,val2=CODE2,valx=CODEx Wandelt Int in Text. Zb. ENUM:0=STANDARD,2=AT,3=ENGLAND,4=ANYWHERE Freie Eingabe: mit „%lld“ oder „%s“ oder „%g“. zb. „S_%lldkW”

Nur bei REST:

BLOCK_START Gliederung der Daten bei XML und JSON

BLOCK_END Gliederung der Daten bei XML und JSON

Zb.

ARGS	FULL	NAME	data	NAME	TEST		HELLO_%s
ARGS	FULL	INHABER	data	INHABER			BLOCK_START
ARGS	FULL	NAME	data	NAME	TEST2		STRING
ARGS	FULL	WERT	data	WERT	12		DEZIMAL
ARGS	FULL	INHABER	data	INHABER			BLOCK_END

Beispiel in JSON:

```
„INHABER“ : {
    „NAME“ : “TEST2“,
    „WERT“ : 12
    }
```

Beispiel in XML:

```
<INHABER>
    <NAME>TEST2</NAME>
    <WERT>12</WERT>
</INHABER>
```

VALUE Default-Wert

ITEM zugewiesenes Item ohne Formel ($..= Zugriff auf GENERALParameter)

Zb. $SLAVE

Die Reihenfolge der Args bestimmt den Telegrammaufbau (bei SERCOM und REST)

zB.

*PREPROCESS	ALIAS	METHOD	URL	DATATYPE	POLLING			
	FULL	GET	http://	JSON	1000			
	*ARGS	METHOD	ALIAS	TYPE	NAME	VALUE	ITEM	FORMAT
	ARGS	FULL	URL	url	/api/v1/trend/meteo/trend%lld/status	0		DEZIMAL
	ARGS	FULL	TSTART	arg	tstart=%s	2017-02-19T15:00:00+01:00	X.10	T_YYYYMMDDHHMM
	ARGS	FULL	TEND	arg	tend=%s	2017-03-19T15:00:00+01:00	X.11	T_YYYYMMDDHHMM
	ARGS	FULL	DATACOUNT	arg	datacount=%lld	96	X.12	DEZIMAL

Groups:

Liste der Sub-Devices eines Gerätes. Zb. hat ein Umrichter meherere Subdevices.

Battery,Netz, PV, House zb. Für den Energiemanager. Aber auch noch Netz-Import, Netz-Export als Zählerdevice oder Batter-Ladung, Entladung.

Die ID wird dann bei den Items diesen als Liste zugewiesen. Dh. ein Item kann auch zu mehreren Gruppen zugewiesen werden. Auch kann der Wert für unterschiedliche Gruppen auch andere Werte haben. Zb. ALIAS=POWER ist Battery_Ladung anders als Netz-Import zsw. Oder Battery-Entladung. Der ALIAS ist dann indentisch. Der Name und ID des Items unterschiedlich.

Die Reihenfolge der Funktionen bestimmt die Abarbeitungsfolge.

Beispiel:

*GROUP	ID	ALIAS	NAME
	0	GRID	GRID
	1	BATTERY	BATTERY
	2	PV	PV
	3	HOUSE	HOUSE
	4	GRID IMPORT	GRID IMPORT
	5	GRID EXPORT	GRID EXPORT
	6	BATTERY CHARGE	BATTERY CHARGE
	7	BATTERY DISCHARGE	BATTERY DISCHARGE

Items:

Liste der Items

ALIAS	Definierter ALIAS für GERÄTE-ZUWEISUNG
NAME	Name welcher im Steuerungssystem erscheint
ID	Eindeutige ID 0..254
VISBILITY	HIDDEN,VISIBLE(Standard=VISIBLE)
UNIT	Einheit
GROUP	siehe Gruppen. List der Gruppen mit , getrennt. zB. 0,1,2,3
TYPE	Leer = STANDARD Wert aus Formula ARRAY =Nur bei REST JSON anwendbar. RFORMULA beinhaltet den Namen der direkten Array -Variable. Bei Abfrage über das Control wird der Wert des übergebenen Indexes zurückgegeben. STRING = Nur bei REST anwendbar. RFORMULA beinhaltet den Namen der direkten Variable. RFORMULA: DATA.arrayvar getSetDeviceItem(…,DATA.arrayvar,10,…) liefert DATA.arrayvar[10] zurück.
DEFAULT	Defaultwert
WFORMULA	Berechungsformel welche beim Schreiben/Ändern der Variable aus dem Steuerungssystem ausgeführt wird
RFORMULA	Berechungsformel welche zyklisch ausgeführt wird
POLLING:	Abfrage/Berechnungsintervall in ms

Beispiel:

*ITEM	ALIAS	NAME	VISIBILITY	ID	UNIT	TYPE	DEFAULT	WFORMULA	RFORMULA	POLLING
	DEVICE	Device		0					"X.0:=D02.0.WORD;"	3000
	DEVICE_VER_(B/P)	Device ver. (B/P)		1			12		"X.1:=D04.0.WORD;"	3000
	PRESSURE_EXHAUST_CHNL	Exhaust chnl pressure		5	Pa				"X.5:=D12.0.WORD;"	3000
	FLOW_SETPOINT_[M3/H]	Setpoint flow		6	m^3/h				"X.6:=D13.0.WORD;"	3000
	PRESSURE_IMB_OK	Pressure imbalance ok		7					"X.7:=D16.0.WORD;"	3000
	PRESSURE_FIXED_IMB	Fixed press imbalance		8	m^3/h				"X.8:=D22.0.INT;"	3000
	FLOW_SUPPLY_CUR	Current supply flow		9	m^3/h				"X.9:=D28.0.WORD;"	3000
	FLOW_EXHAUST_CUR	Current exhaust flow		10	m^3/h				"X.10:=D29.0.WORD;"	3000
	BYPASS_FLAP_POS	Bypass flap pos		11					"X.11:=D30.0.WORD;"	3000
	BYPASS_FLAP_FUNC	Bypass flap func		12					"X.12:=D31.0.WORD;"	3000
	PREHEAT_REG_STATUS	Preheat reg status		13					"X.13:=D37.0.WORD;"	3000
	POWER_PREHEAT_REG	Preheat reg power		14	%				"X.14:=D38.0.WORD;"	3000
	ERROR_CODE_CUR	Current error code		15					"X.15:=D39.0.WORD;"	3000
	FILTER_DISPLAY	Filter display		16					"X.16:=D40.0.WORD;"	3000
	GROUND_HEAT_MODE	Ground heat mode		17					"X.17:=D41.0.WORD;"	3000
	TEMP_GROUND_MIN	Min ground heat temp		18	°C/10				"X.18:=D47.0.INT;"	3000
	TEMP_GROUND_MAX	Max ground heat temp		19	°C/10				"X.19:=D53.0.INT;"	3000
	CO2_SENSOR_NUM	CO2 sensor # (max4)		20					"X.20:=D59.0.WORD;"	3000
	CO2_SENSOR_VALUE	CO2 sensor value		21	ppm				"X.21:=D60.0.WORD;"	3000
	MODBUS_SLAVE_ADDR	Modbus slave addr		30				"W00.0.WORD:=X.30.WORD; W00.F:=1;"		3000

Formel:

Wichtig: Bei Verwendung von ; in den Formeln muss die Formel mit „“ eigefasst werden. Zudem muss das letzte Zeichen innerhalb der „ ein ; sein.

Zugriff auf Items:

X.0 Zugriff auf die Itemvariablen.

0= ID des Items

Zugriff auf System-Variablen:

$.SYS

Aktuell:

$.SYS.TIME Systemzeit in Sekunden

Zugriff auf Temporäre Variablen:

Wird der Name der Funktion nicht gefunden wird dies als TEMP Variable verwendet. Variablen gelten nur innerhalb des Devices(Slave)

Zugriff auf Kommunikationsmethoden:

NNN.xxx NNN= ALIAS der Methode.

Xxx= Abhängig vom Protokoll

Generelle Variablen:

F=Forcen/Changed (GET/SET)

L=Datenlänge

Q=Qualität

EC=Errorcounter (GET/SET)

EN=Error-Nummer

T=Timesstamp (GET/SET))

P=Pausieren für x Zyklen (GET/SET)

Modbus:

xxx.FORMAT	xxx=Start-Register
FORMAT	Konvertierung
	WORD/UINT/DWORD/DUINT/LWORD/LUINT/INT/DINT/LINT/FLOAT

Sercom(Ascii)

GET/LESEN

xxx.yyy.FORMAT	xxx=Start-Byte, yyy=Länge,FORMAT=Format
FORMAT	INT,FLOAT,BYTESUM(Summe aller Bytes)
xxx.zzz	xxx=Byte
	zzz=String-Enumliste
	zb. NNN.8.NO,YES,MAYBEE
	Ist der String ab dem 8.Byte gleich mit „NO“ , dann ist Wert 0
	YES=1, MAYBEE=2, Anders =0

SET/SCHREIBEN

ARG.xxx	xxx = ALIAS des Argumentes der Funktion

Rest(JSON)

GET/LESEN

xxx xxx= HTTP_CODE

xxx= VALUE Wert des Objektes

VAL.NNN.NNN.NNN

NNN= Name des Objektes

xxx=ARRAY Zugriff auf eine Arrary mit den Folgefunktionen

ARRAY.MEDIA.NNN.NNN.NNN =Mittelwert

ARRAY.MAX.NNN.NNN.NNN =Maxwert

ARRAY.MIN.NNN.NNN.NNN =Minwert

ARRAY.LEN.NNN.NNN.NNN =Länge des Arrays

NNN= Name des Objektes

xxx=FIND Vergleichen des Wertes mit dem folgenden String

FIND.STRING.NNN.NNN.NNN Sucht den STRING im Objektwert

NNN= Name des Objektes

SET/SCHREIBEN

ARG.xxx xxx = ALIAS des Argumentes der Funktion

Aufbau der Formel:

Die Formeln müssen in „ eingeschlossen sein, da sie ; als Befehlszeichen verwenden. Auch newLine ist anwendbar.

Aufbau:

Beinhaltet alle Möglichkeiten der aktuellen Formelberechungen. Für die richtige Ausführung ist die konsequente Anwendung von Klammern () wichtig.

Operationen:

*	MUL
/	DIV
+	ADD
-	SUB
^	POT
>	GR
>=	GGL
<	KL
<=	KGL
!=	UGL
==	GL
>>	SHR
<<	SHL
&&	AND
||	OR
&	BAND
|	BOR
:=	RESULT

Strukturfunktionen: IF THEN ELSE ENDIF;

IF THEN Anweisungen dürfen nicht verschachtelt werden. Jedoch können mehrere IF THEN ELSE IF THEN ELSE IF THEN ELSE ENDIF; angelegt werden.

Beispiel:

```
"IF X.0 >100 THEN
 X.1:=X.1+ 1;
ELSE IF X.0 > 10 THEN
X.1:=X.1+10;
TESTVAR:=(X.1+X.2)*(10-9);
DEBUG(TESTVAR,X.2,X.0);
ELSE
X.1:=X.1+1000;
ENDIF;
IF X.1 > 1000 THEN
X.2:=10;
ELSE IF X.1 >200 THEN X.2:=100;  ELSE  X.2:=1000; ENDIF;
X.2:=X.2+10;"
```

Variablenzugriff:

Siehe oben.

Funktionsaufrufe in Formeln:

In Formeln ist es möglich Funktionen zu verwenden. Die Funktionen können mehre Parameter haben. Diese müssen Variablennamen sein und nicht direkt numerische Werte. Die gesamte Funktion darf kein Leerzeichen enthalten und auch keine Operationszeichen wie +-/….

Zb. ~~DEBUG(TEST, TEST);~~ DEBUG(TEST,TEST);

Aufbau: NNN(XXX,XXX,XXX,XXXX)

NNN= Name der Funktion

XXX= Parameter der Funktion. Diese müssen Variablennamen sein und nicht direkt numerische Werte. In dem Fall müssen die Werte zuerst in eine Variable geschrieben werden und dann dieses al Parameter übergeben.

Wichtig: Die Funktion darf kein Leerzeichen enthalten und auch keine Operationszeichen wie +-/….

Beispiel:

X.1:=DATE_DAY($.SYS.TIME);

Schreibt in X.1 den Monatstag der aktuellen Uhrzeit rein.

UHRZEIT:=$:SYS.TIME-86000; X.1:=DATE_DAY(UHRZEIT);

Schreibt in X.1 den Monatstag des Vortages.

Aktuelle Funktionen:

Bei den Funktionen müssen die Paramter als Variablen übergeben werden. Direkt Werte gehen nicht. (Grund: bei einer Zahl mit punkt kommt das Aufsplitten durcheinander.)

Beispiel: PARAM_TIME:=10000; X.1:= DATE_MONTH(PARAM_TIME);

Datumsfunktionen:

DATE_MONTH(PARAM_TIME),

DATE_YEAR(PARAM_TIME),

DATE_YDAY(PARAM_TIME),

DATE_HOUR(PARAM_TIME),

DATE_MIN(PARAM_TIME),

DATE_SEC(PARAM_TIME)

PARAM_TIME= VARIABLE mit TIME WERT (Zeit in Sekunden)

Zeitfunktionen:

TIME_FROM_YDAY(PARAM_YEAR, PARAM_YEARDAY)

TIME_FROM_DATE (YEAR,MON, DAY)

Math-Funktionen

LOG(VALUE,BASE)

Help-Funktionen:

DEBUG(VAR1,VAR2,VAR2,..);

Die Funktion DEBUG gibt im Log des Device die Namen der Variablen mit dem aktuellen Wert aus.

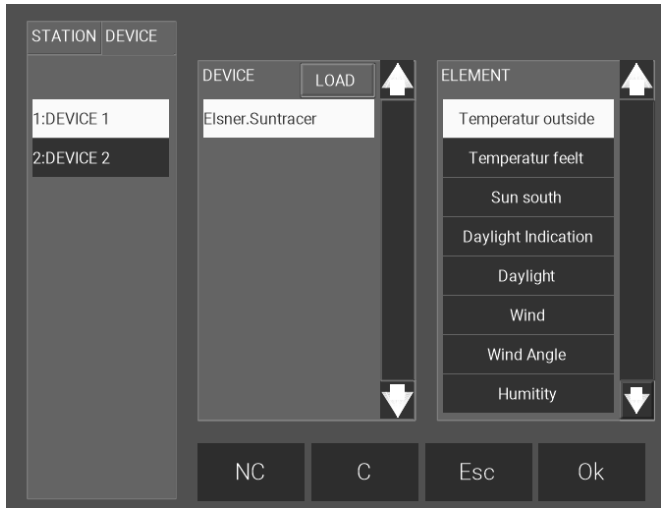
Zb. IF .. THEN DEBUG(VAR1,VAR2,X.10); …

Erscheint im Log: 10:12:30:VAR1=6,VAR2=3,X.10=10.2);

Device im OS:

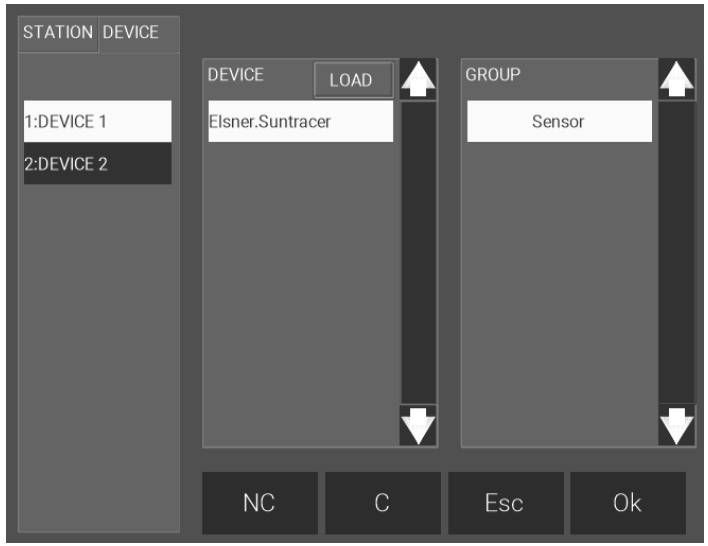
Freie Zuordnung

Die Items können wie jede Ios in allen Systemen/Elementen verwendet werden.



Plug&Play as Device

In verschiedene Elementen können die Devices ausgewählt werden um ein komplettes Gerät oder Subgerät(Gruppe) dem Element zuzuordnen.



Es werden ALIAS verwendet, damit das OS in einem Device nach dem passenden Werten verlinkt.

Abhängig vom Funktionsbaustein sucht das OS im dem Element/Funktion zugeordnetem Device nach den passenden Werten.

z.B. In der Wetterstation sucht das OS automatisch nach TEMPERATUR, SUN_EAST,SUN_WEST,…..

Findet er den Wert wird dieser zugeordnet.

Im Energiezähler sucht das OS den Wert ENERGY und POWER. Die Werte müssen in den passenden Einheiten im Device vorhanden sein.

Die Kombi aus ALIAS und eventuell Gruppe ergibt die Zuordnung im OS bei den Elementen/Funktionen.

Anhang:

Modus-Fehlercode

Fehlercode (Hex)	Fehlercode (Dez)	Bezeichnung
-0x01	-1	ILLEGAL_FUNCTION
-0x02	-2	ILLEGAL_DATA_ADDRESS
-0x03	-3	ILLEGAL_DATA_VALUE
-0x04	-4	SLAVE_DEVICE_FAILURE
-0x05	-5	ACKNOWLEDGE
-0x06	-6	SLAVE_DEVICE_BUSY
-0x07	-7	NEGATIVE_ACKNOWLEDGE
-0x08	-8	MEMORY_PARITY_ERROR
-0x0A	-10	GATEWAY_PROBLEM_PATH
-0x0B	-11	GATEWAY_PROBLEM_TARGET
-0x0C	-12	COMM_TIME_OUT
-0x0D	-13	PORT_SOCKET_FAILURE
-0x0E	-14	SELECT_FAILURE
-0x0F	-15	TOO_MANY_DATA
-0x10	-16	INVALID_ID
-0x11	-17	INVALID_EXCPETION_CODE
-0x12	-18	CONNECTION_CLOSED
-0x13	-19	TOO_BIG_PACKET
Nicht angegeben	Nicht angegeben	NOT_CONNECTED
Default		OK

DEVICEDB Portal

Das Portal listet alle im OS integrierten öffentlich verfügbaren Geräte. Jedes Gerät hat eine eindeutige UID welche sich aus dem Hersteller, Model(Gerät), Protokoll und Version der Integration zusammensetzt.

DEVICE DB Login

MANUFACTURER	NAME	TYPE	PROTOCOL	
				Filtern

MANUFACTURER	NAME	TYPE	PROTOCOL	STATE
BRINK	Brink Excellent HRV	Ventilation unit	MODBUS RTU(new)	no integration
Elsner	P04/3-RS485 basic	Weather station	SERCOM(new)	profile implemented
Elsner	P04/3-RS485-CET	Weather station	SERCOM(new)	no integration
Elsner	P04/3-RS485-GPS	Weather station	SERCOM(new)	no integration
Elsner	Suntracer	Weather station	SERCOM(new)	no integration
Frauenhofer ISE	EnergyChartsPrice	Portal	REST(new)	profile checked
go-e	Charger Gemini	Wallbox	MODBUS TCP(new)	no integration
Hoval	Hoval HomeVent	Ventilation unit	MODBUS TCP(new)	no integration
Huawei	SUN2000	PV-Inverter	MODBUS TCP(new)	pre-check done
Meltem	M-WRG-II xx	Ventilation unit	MODBUS RTU(new)	profile documented
Mennekes	Amtron Charge Control	Wallbox	MODBUS TCP(new)	profile implemented
myGEKKO	API TREND	Automation Control System	REST(new)	no integration
myGEKKO	GENERIC DDF	Multifunctional	SERCOM(new)	no integration
myGEKKO	Mifare Reader	Access	SERCOM(new)	no integration
Seneca	Seneca(S504C B19200)	Current meter	MODBUS RTU(new)	pre-check done
SunGrow	SGxx series	PV-Inverter	MODBUS TCP(new)	profile implemented
Thies	WSC11 4.90056.10.000	Weather station	SERCOM(new)	profile implemented
Thies	WSC11 4.90056.10.001	Weather station	MODBUS RTU(new)	pre-check done

In der Detailansicht der Geräte finden sich die Informationen zum Gerät, eventuelle Herstellerunterlagen und je nach Integrationstyp die Detailinfos. Ziel ist es, dass der Kunde einen Überblock über die Integrationstiefe des Gerätes und der damit verbunden Möglichkeiten zur Verwendung im OS erhält.

DEVICE DB | **Login**

General parameter

DEVICENAME:	SUN2000
MANUFACTURER:	Huawei
TYPE:	PV-Inverter
PROTOCOLL:	MODBUS TCP(new)
MODEL_NR:	1
VERSION_NR:	1
ID	0x0A00000900010100
MIN_MYGEKKO_VERSION:	
FUNCTION_TYPE:	Interface
PROGRESS_STATE:	pre-check done

Note

from

Documents

Solar Inverter Modbus Interface Definitions.pdf

Device Description File

SUN2000[Huawei]PV-Inverter MODBUS TCP(new) (0x0A00000900010100).ddf

INDEX	ALIAS	NAME	UNIT
0	POWER_IMPORT	Grid Power import	kW
1	POWER_EXPORT	Grid Power export	kW
2	ENERGY_IMPORT	Grid Energy import	Wh
3	ENERGY_EXPORT	Grid Energy export	Wh
4	POWERMAX	Grid Power max	kW
5	POWER_CHARGE	Battery Power charge	kW
6	POWER_DISCHARGE	Battery Power discharge	kW
7	ENERGY_CHARGE	Battery Energy charge	Wh
8	ENERGY_DISCHARGE	Battery Energy discharge	Wh
9	SOC	Battery SoC	%
10	POWERMAX	Battery Power max	kW
11	POWER	PV Power	kW
12	ENERGY	PV Energy	Wh
13	VOLTAGE_L1	PV Voltage L1	V

Für Geräte die über DDF integriert sind kann der Nutzer die DDF downloaden,um sie dann im OS auf das entsprechende Devices upzuloaden und zu verwenden.

Wird das Gerät eingebunden, passt die die Eingabemaske den Möglichkeiten des Gerätes an und kann darüber parametriert werden.

Das Verhalten ist ähnlich den IO-Stationen. Je Gerät/Device können bis zu 250 Unter/SubDevices angelegt werden. Es können aktuell max.30 Devices angelegt werden.