

## Inhalt

1. Datentypen .....	2
1.1. Datentyp ow_parameter.....	2
1.2. iButton .....	3
1.3. Temperatursensor DS18B20 / DS18S20.....	3
1.4. ADC1 DS2438 .....	3
1.5. ADC2 DS2450 .....	4
1.6. DS2405, DS2408, DS2413.....	4
2. Funktionsbausteine.....	5
2.1. i_Button_suchen .....	5
2.2. one_wire .....	6
o Temperatur – Spannungswerte .....	7
o binäre Eingänge.....	7
o binäre Ausgänge.....	8
3. Starteinstellungen.....	8
3.1. Startprogramm für Sensoren .....	8
3.2. Startprogramm iButton.....	9
4. Vorgehensweise .....	9
4.1. iButton .....	9
4.2. Sensoren .....	10

# 1. Datentypen

## 1.1. Datentyp ow\_parameter

Diese Parameter dienen zur Steuerung und Überwachung des one-wire Bausteins. Hier können diverse Einstellungen vorgenommen werden.

Parameter		Beschreibung
com_port_nr		1= interne Schnittstelle
Busstatus	Bus_i_o	Initialisierung des DS 2480 war erfolgreich, d.h. DS2480 antwortet
	Bus_Start	Startsignal für Bus, Signal wird automatisch abgeschaltet
	ow_bus_pruefen_aktiv	Busprüfung ein oder ausschalten: Es wird geprüft wie viel Sensorstörungen aufgetreten sind, bei Überschreitung eines Maximalwertes erfolgt ein RESET des DS2480.
	ow_pruefen	zyklisches Prüfen der Störungen, wenn ow_bus_pruefen_aktiv= TRUE
	bus_init_busy	internes Signal
	Stoerung_ow_Baustein	Es ist eine Störung im Baustein aufgetreten, es erfolgt ein RESET des DS2480.
	Versuche	Warten auf Antwort vom DS2480 bei der Initialisierung. Es werden maximal 6 Versuche zugelassen. Wenn Initialisierung erfolgreich war, dann hat Bus_i_o= TRUE-Signal. Wenn im Parameter Versuche eine 6 steht, dann antwortet der DS2480 nicht.
ow_zeiten	Aufrufintervall	Intervall der Bausteinaufrufe Geschwindigkeit der Datenübertragung [default T#60ms]
	Abtastzeit	Abtastintervall der Sensoren, in welchem Abstand die Messung durchgeführt werden [default T#4m] ( <b>bezieht sich auf T, D1 und D2 Bausteine</b> )
	Ablaufüberwachungszeit	Internes Signal
sonstige_werte	one_wire_Reset	Initialisierungsanforderung , ow-Baustein, Signal wird vom ow-Baustein abgeschaltet
	Mitternachts_Reset	Impuls, welcher Störungen um Mitternacht zurücksetzt, Signal wird vom ow-Baustein abgeschaltet. Wenn dies Funktion gewünscht ist, muss <b>Mitternachts_Reset</b> im eigenem Programm eingeschaltet werden.
werte_neu	Sensordaten_T_neu	Temperatur Messwerte können gelesen werden
	Sensordaten_D_neu	ADC Messwerte können gelesen werden
	bin_Ausgaenge	Ausgangsdaten werden aktualisiert
	bin_Eingaenge	Eingangsdaten haben sich geändert
	ESR_1_w_ok	Messwerte 1-w sind eingelesen
ID suchen	neue_ID_suchen	Startsignal zur Suche, wird automatisch abgeschaltet
	was_suchen	Typen auswählen, welche gesucht werden sollen
	anzahl_gefunden	hier wird automatisch eingetragen wie viele gefunden wurden
	gefundener_typ	internes Signal wo eingetragen wird welche Bausteine präsent sind
	gefundener_typ	Hier wird eingetragen, welche Bausteine gefunden wurden.
	Aktivmeldung	Visuelles Signal um die Aktivität des Bausteins zu überprüfen.
	DI_nur_bei_aenderung	Eingangssignale werden nur bei Signaländerung übernommen
	Anzahl_Globalstoerungen	So oft wurde der Bus infolge von Störungen neu gestartet.

serieller Zugriff einer WAGO 750-841/881 auf den one-wire Bus mit einem DS2480B als Gateway

## 1.2.iButton

Struct i\_Button\_key:

Name	STRING	Name des keys (optional)
ID	ARRAY [1..8] OF BYTE	8-stellige ID des Sensors
aktiv	BOOL	iButton angeschlossen (TRUE-aktiv)

## 1.3.Temperatursensor DS18B20 / DS18S20

one\_wire\_T\_Sensor\_V6:

Name	STRING	Name des Sensors (optional)
ID	ARRAY [1..8] OF BYTE	8-stellige ID des Sensors
Temperatur	REAL	Messergebnis
Stoerungen	INT	Anzahl der Sensorstörungen
Temperatur_max	REAL	maximaler Messwert
Temperatur_min	REAL	minimaler Messwert
Fehler	BOOL	es sind mehr als 5 Störungen aufgetreten
aktiv	BOOL	Sensor wird bearbeitet (TRUE-aktiv)

Temperatur max/min werden zur Auswertung der Messwerte verwendet, d.h. es werden nur Messwerte innerhalb von min bis max berücksichtigt.

Wenn min = max oder max < min, dann wird jeder Wert berücksichtigt ohne Prüfung.

## 1.4. ADC1 DS2438

one\_wire\_D1\_Sensor\_V6:

Name	STRING	Name des Sensors (optional)
ID	ARRAY [1..8] OF BYTE	8-stellige ID des Sensors
Temperatur	REAL	Messergebnis
Helligkeit	REAL	Helligkeit in % von max. Helligkeit Wenn Helligkeitssensor bei maximaler Helligkeit eine Spannung von 4,5V liefert, und bei max_Helligkeit 4.5 eingetragen ist, dann liefert Helligkeit 100% bei maximaler Beleuchtung.
Busspannung	REAL	Busspannung [normal 5V]
Stoerungen	INT	Anzahl der Sensorstörungen
max_Helligkeit	REAL	Wert für 100% s.o.
Fehler	BOOL	es sind mehr als 5 Störungen aufgetreten
aktiv	BOOL	Sensor wird bearbeitet (TRUE-aktiv)

serieller Zugriff einer WAGO 750-841/881 auf den one-wire Bus mit einem DS2480B als Gateway

## 1.5. ADC2 DS2450

one\_wire\_D2\_Sensor\_V6

Name	STRING	Name des Sensors (optional)	
ID	ARRAY [1..8] OF BYTE	8-stellige ID des Sensors	
AI1	REAL	Messergebnis ADC A	
AI2	REAL	Messergebnis ADC B	
AI3	REAL	Messergebnis ADC C	
AI4	REAL	Messergebnis ADC D	
E	Auflösung A	INT	Auflösung Kanal A
	Auflösung B	INT	Auflösung Kanal B
	Auflösung C	INT	Auflösung Kanal C
	Auflösung D	INT	Auflösung Kanal D
	u_ref_A	BOOL	0=2,56V/1=5,12V
	u_ref_B	BOOL	0=2,56V/1=5,12V
	u_ref_C	BOOL	0=2,56V/1=5,12V
	u_ref_D	BOOL	0=2,56V/1=5,12V
AI_DO	BYTE	2#Freigabe,Belegung linke Tetrade Freigabe DCBA, 1-messen/0 -nix rechte Tetrade Belegung DCBA, 1-Binärausgang/0-AI oder frei	
Anzahl_DO	INT	wird automatisch ermittelt und eingetragen	
Ausgangsdaten	DO_A	BOOL	Signalzustand Kanal A, wenn als Ausgang verwendet
	DO_B	BOOL	Signalzustand Kanal B, wenn als Ausgang verwendet
	DO_C	BOOL	Signalzustand Kanal C, wenn als Ausgang verwendet
	DO_D	BOOL	Signalzustand Kanal D, wenn als Ausgang verwendet
Status	DO_A	BOOL	Status Kanal A, wenn als Ausgang verwendet, interne Verwendung
	DO_B	BOOL	Status Kanal B, wenn als Ausgang verwendet, interne Verwendung
	DO_C	BOOL	Status Kanal C, wenn als Ausgang verwendet, interne Verwendung
	DO_D	BOOL	Status Kanal D, wenn als Ausgang verwendet, interne Verwendung
Stoerungen	INT	Anzahl der Sensorstörungen	
Fehler	BOOL	es sind mehr als 5 Störungen aufgetreten	
aktiv	BOOL	Sensor wird bearbeitet (TRUE-aktiv)	

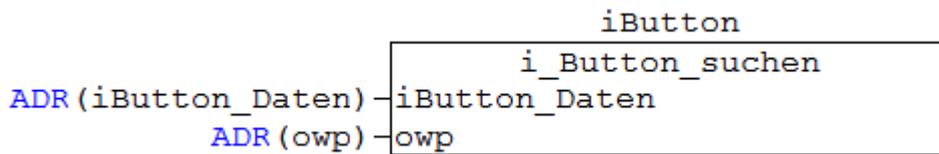
## 1.6.DS2405, DS2408, DS2413

one\_wire\_binaer\_V6

Name	STRING	Name des Sensors (optional)
ID	ARRAY [1..8] OF BYTE	8-stellige ID des Sensors
Belegung	BYTE	0- Eingang 1- Ausgang Bsp. 2#00000111 bedeutet 2 <sup>0</sup> bis 2 <sup>2</sup> sind Ausgänge und 2 <sup>3</sup> bis 2 <sup>7</sup> sind Eingänge (DS2408)
Status_Byte	BYTE	Status der Ports
Aenderung_Byte	BYTE	Port, an welchem sich der Zustand geändert hat
Ausgangs_Byte	BYTE	Wert, welcher geschrieben werden soll
Stoerungen	INT	Anzahl der Sensorstörungen
Fehler	BOOL	es sind mehr als 5 Störungen aufgetreten
aktiv	BOOL	Sensor wird bearbeitet (TRUE-aktiv)

## 2. Funktionsbausteine

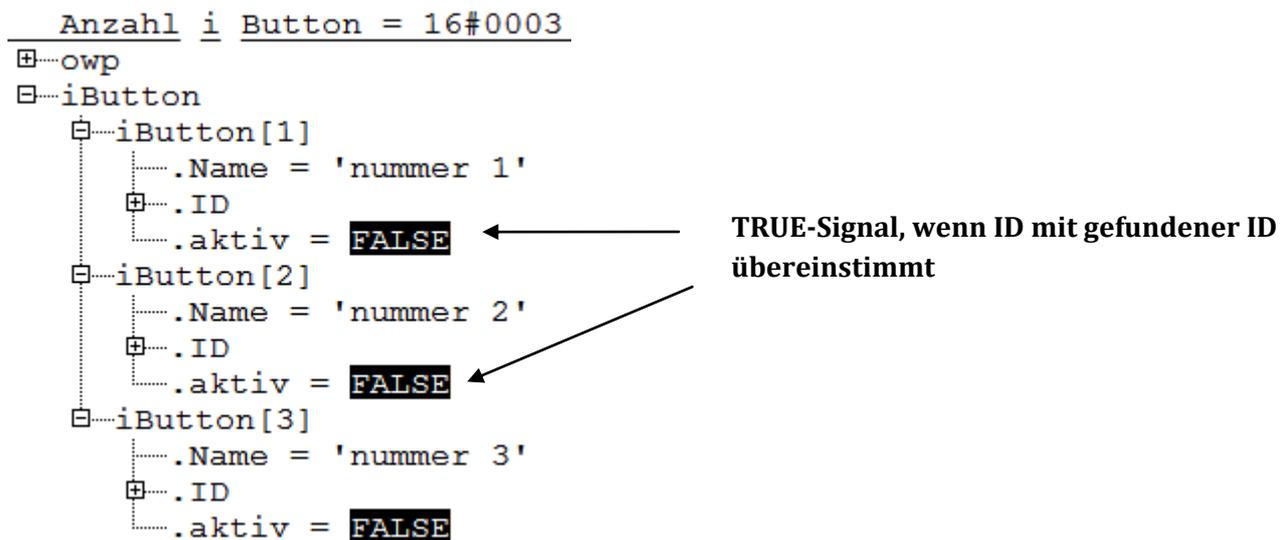
### 2.1.i\_Button\_suchen



owp	Pointer (Zeiger auf das Array ow_parameter)
iButton_Daten	Pointer (Zeiger auf das Array iButton)

- Der Funktionsbaustein 'i\_Button\_suchen' wird im zyklischen Programm aufgerufen (mit 10 – 50ms Zykluszeit getestet).
- Der Parameter **'owp.ow\_Zeiten.Aufrufintervall'** bestimmt die Geschwindigkeit der Datenübertragung auf dem one-wire Bus, mit einem Wert von T#40-100ms sollte auch ein größeres Netzwerk stabil laufen.
- **Aufrufintervall muss größer als die Zykluszeit sein!**
- Wenn viele Sensorstörungen auftreten, muss der Wert vergrößert werden (in 10ms Schritten).
- Jede gefundene iButton-ID wird mit den hinterlegten ID's verglichen und bei Übereinstimmung wird *aktiv* = TRUE.

#### Globale Variable: one\_wire\_Daten



- Bei Programmstart müssen die beiden Parameter *one\_wire\_Reset* und *Bus\_Start* auf TRUE gesetzt werden.

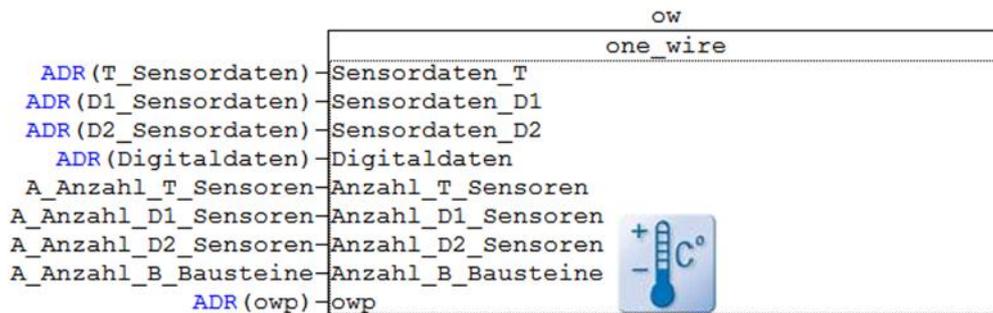
```

owp.sonstige_werte.one_wire_Reset:= TRUE;
owp.Busstatus.Bus_Start:= TRUE;

```

## 2.2.one\_wire

Baustein für Temperatur, -Spannungsmessung und für Binärbausteine.



Sensordaten_T	POINTER TO ARRAY	Zeiger auf das Array der Temperatursensordaten
Sensordaten_D1	POINTER TO ARRAY	Zeiger auf das Array der D1-Sensoren
Sensordaten_D2	POINTER TO ARRAY	Zeiger auf das Array der D2-Sensoren
Digitaldaten	POINTER TO ARRAY	Zeiger auf das Array der Binärsensoren
Anzahl_T_Sensoren	INT	Anzahl Temperatursensoren
Anzahl_D1_Sensoren	INT	Anzahl D1 Sensoren
Anzahl_D2_Sensoren	INT	Anzahl D2 Sensoren
Anzahl_B_Bausteine	INT	Anzahl Binärsensoren
owp	POINTER TO ow_parameter	Zeiger auf das Array ow_parameter

Für den Baustein muss die Globale Variable **'one-wire\_Daten'** angelegt werden.

### Beispiel:

```

VAR_GLOBAL CONSTANT
    Anzahl_D1_Sensoren :INT:=3;      (*Anzahl ADC1 Sensoren*)
    Anzahl_D2_Sensoren :INT:=3;      (*Anzahl ADC2 Sensoren*)
    Anzahl_B_Bausteine :INT:=3;      (*Anzahl an binären 1-wire Bausteinen *)
    Anzahl_T_Sensoren  :INT:=2;      (*Anzahl Temperatursensoren*)
END_VAR
VAR_GLOBAL
    owp: ow_parameter; (*Parameter für one_wire Baustein*)
    (*Messwerte*)
    Temperatur_Sensordaten: ARRAY[1..Anzahl_T_Sensoren] OF one_wire_T_Sensor_V4:=
    ( Name:='Sensor 1:TEST1', ID:=16#28,16#30,16#9D,16#CE,16#01,16#00,16#00,16#2F),
    ( Name:='Sensor 2:TEST2', ID:=16#28,16#FE,16#AD,16#CE,16#01,16#00,16#00,16#27);

    D_Sensordaten: ARRAY[1..Anzahl_D_Sensoren] OF one_wire_D_Sensor_V4:=
    ( Name:='Sensor 1:TEST1', ID:=16#26,16#BD,16#BE,16#CF,16#00,16#00,16#00,16#62),
    ( Name:='Sensor 2:TEST2', ID:=16#26,16#63,16#BF,16#CF,16#00,16#00,16#00,16#78),
    ( Name:='Sensor 3:TEST3', ID:=16#26,16#9B,16#BF,16#CF,16#00,16#00,16#00,16#AB);
    (*-----*)
    Digitaldaten: ARRAY[1..Anzahl_B_Bausteine] OF one_wire_binaer_V4:=
    (Name:='DS2405',
    ID:=16#05,16#27,16#CF,16#20,16#00,16#00,16#00,16#C7,Belegung:=2#00000001),
    (Name:='DS2413',
    ID:=16#3A,16#A8,16#7E,16#03,16#00,16#00,16#00,16#25,Belegung:=2#00000001),
    (Name:='DS2408',
    ID:=16#29,16#05,16#95,16#09,16#00,16#00,16#00,16#3E,Belegung:=2#00000011);
    
```

- Temperatur – Spannungswerte: wenn der Messvorgang beendet ist, wird für Temperatursensoren `owp.werte_neu.Sensordaten_T_neu` und für ADC-Sensoren `owp.werte_neu.Sensordaten_D_neu` auf **TRUE** gesetzt. Damit kann eine ereignisgesteuerte Task gestartet werden um mit einem Programm die Messwerte aus den Arrays zu lesen. In diesem Programm muss `owp.werte_neu.Sensordaten_T_neu` bzw. `owp.werte_neu.Sensordaten_D_neu` auf **FALSE** gesetzt werden.

Wenn die Messwerte direkt aus dem Array gelesen werden, kann `owp.werte_neu.Sensordaten_T_neu` und `owp.werte_neu.Sensordaten_D_neu` vernachlässigt werden.

<b>Programm zum Lesen der Messdaten:</b>
Temperaturwert_1:=Temperatur_Sensordaten[1].Temperatur;
Temperaturwert_1:=Temperatur_Sensordaten[2].Temperatur;

- binäre Eingänge: `owp.werte_neu.bin_Eingaenge` wird **TRUE**, wenn sich ein Eingangssignal geändert hat sofern `owp.DI_nur_bei_aenderung` **TRUE**-Signal hat. Bei `owp.DI_nur_bei_aenderung` = **FALSE**, wird o.g. Wert in jedem Zyklus gestartet.

Mit diesem Wert kann eine ereignisgesteuerte Task gestartet werden in welcher ein Programm zum Lesen der Eingangsdaten aufgerufen wird. Wenn die Eingangssignale zyklisch gelesen werden, kann dieser Parameter vernachlässigt werden.

Programm zum Lesen der Eingangsdaten:	
0001	<div style="display: flex; align-items: center;"> <div style="border: 1px dashed gray; padding: 2px; margin-right: 10px;">Eingang_1</div> <div style="flex-grow: 1; border-bottom: 1px solid gray; margin-bottom: 5px;"></div> <div style="border: 1px dashed gray; padding: 2px; margin-left: 10px;">Digitaldaten[3].Status_Byte.3</div> </div> <p style="text-align: right;">Ein Bit wird gelesen und abgelegt</p>
0002	<div style="display: flex; align-items: center;"> <div style="border: 1px dashed gray; padding: 2px; margin-right: 10px;">Eingang_2</div> <div style="flex-grow: 1; border-bottom: 1px solid gray; margin-bottom: 5px;"></div> <div style="border: 1px dashed gray; padding: 2px; margin-left: 10px;">Digitaldaten[3].Status_Byte.4</div> </div>
0003	<div style="display: flex; align-items: center;"> <div style="border: 1px dashed gray; padding: 2px; margin-right: 10px; color: magenta;">FALSE</div> <div style="flex-grow: 1; border-bottom: 1px solid gray; margin-bottom: 5px;"></div> <div style="border: 1px dashed gray; padding: 2px; margin-left: 10px;">owp.werte_neu.bin_Eingaenge</div> </div> <p style="text-align: right;">owp.werte_neu.bin_Eingaenge wird abgeschaltet</p>

○ binäre Ausgänge:

**owp.werte\_neu.bin\_Ausgaenge** wird in jedem Zyklus auf TRUE gesetzt.

Mit diesem Wert kann eine ereignisgesteuerte Task gestartet werden in welcher ein Programm zum Schreiben der Ausgangsdaten aufgerufen wird.

Programm zum Schreiben der Ausgangsdaten:		
0001	TEST_DO1——Digitaldaten[1].Ausgangs_Byte.0	Ein Bit wird geschrieben
0002	TEST_DO2——Digitaldaten[3].Ausgangs_Byte.0	
0003	FALSE——owp.werte_neu.bin_Ausgaenge□	owp.werte_neu.bin_Ausgaenge wird abgeschaltet

### 3. Starteinstellungen

#### 3.1. Startprogramm für Sensoren

Es werden nur diejenigen one-wire Sensoren gemessen in welchen aktiv=TRUE ist.

Im Startprogramm muss bei allen Sensoren, welche abgearbeitet werden sollen, aktiv auf TRUE geschaltet werden.

Startprogramm: für DS18B20+DS2438+DS2405+DS2408+DS2413		
owp.sonstige_werte.one_wire_Reset	:= TRUE;	(*RESET Signal*)
owp.DI_nur_bei_aenderung	:= TRUE;	(*Eingangssignale nur bei Signaländerung lesen*)
owp.Busstatus.ow_bus_pruefen_aktiv	:=TRUE;	
owp.Busstatus.Bus_Start	:= TRUE;	
owp.ow_zeiten.Abstastzeit	:= T#1m;	(*alle Sensoren werden im Minutentakt gemessen*)
owp.Busstatus.Stoerung_ow_Baustein	:=FALSE;	
FOR x:=1 TO Anzahl_T_Sensoren DO		
Temperatur_Sensordaten[x].aktiv:= TRUE;		(*alle Temperatursensoren sind aktiv*)
END_FOR;		
FOR x:=1 TO Anzahl_B_Bausteine DO		
Digitaldaten[x].aktiv:= TRUE;		(*alle Binärbausteine sind aktiv*)
END_FOR;		
FOR x:=1 TO Anzahl_D1_Sensoren DO		
D1_Sensordaten[x].aktiv:= TRUE;		(*alle ADC Sensoren sind aktiv*)
END_FOR;		

### 3.2. Startprogramm iButton

<b>Startprogramm: für iButton</b>	
owp.sonstige_werte.one_wire_Reset:= TRUE;	(*RESET Signal*)
owp.Busstatus.Bus_Start:= TRUE;	

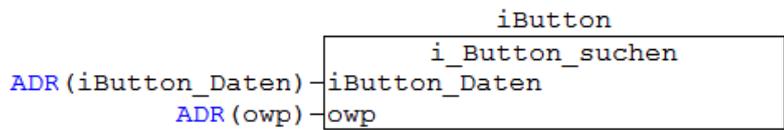
## 4. Vorgehensweise

### 4.1. iButton

- Globale Variable → ‚one\_wire\_Daten‘ anlegen
  - Anzahl\_i\_Button als VAR\_GLOBAL CONSTANT einfügen
  - one-wire Parameter einfügen
  - Array mit ID's der iButton einfügen
  - ID's müssen manuell eingetragen

<pre> Ergebnis: VAR_GLOBAL CONSTANT     Anzahl_i_Button: INT := 3; END_VAR VAR_GLOBAL     owp: ow_parameter; (*Parameter für one_wire Baustein*)     iButton: ARRAY [1..Anzahl_i_Button] OF i_Button_key:=         (Name:='nummer 1',          ID:=16#01,16#F2,16#DB,16#58,16#13,16#00,16#00,16#65),         (Name:='nummer 2',          ID:=16#01,16#0D,16#FF,16#5B,16#13,16#00,16#00,16#5C),         (Name:='nummer 3',          ID:=16#00,16#00,16#88,16#15,16#13,16#00,16#00,16#7A); END_VAR                 </pre>
---

- Startprogramm anlegen **s.o.**
- Programm für iButton anlegen ‚**ow\_iButton**‘
  - Funktionsbaustein ‚**i\_Button\_suchen**‘ aufrufen
  - owp und iButton zuweisen



- Zyklische Task T#40ms einfügen und Programm ‚**ow\_iButton**‘ aufrufen

## Zur Steuerung übertragen und Testen.

serieller Zugriff einer WAGO 750-841/881 auf den one-wire Bus mit einem DS2480B als Gateway

## 4.2. Sensoren

- Globale Variable → ‚one\_wire\_Daten‘ anlegen
- Anzahl\_T\_Sensoren, Anzahl\_D1\_Sensoren, Anzahl\_D2\_Sensoren, Anzahl\_B\_Bausteine als VAR\_GLOBAL CONSTANT einfügen
- one-wire Parameter einfügen
- Array mit ID's der Sensoren einfügen
- ID's müssen manuell eingetragen werden, sonst ist die Zuordnung schwer

Ergebnis:

```
VAR_GLOBAL CONSTANT
  Anzahl_T_Sensoren: INT := 2;      (*Anzahl Temperatursensoren*)
  Anzahl_D1_Sensoren: INT := 3;    (*Anzahl ADC D1 Sensoren*)
  Anzahl_D2_Sensoren: INT := 0;    (*Anzahl ADC D2 Sensoren*)
  Anzahl_B_Bausteine: INT :=3;     (*Anzahl binärer 1-wire Bausteine*)
END_VAR
VAR_GLOBAL
  owp: ow_parameter; (*Parameter für one_wire Baustein*)
  (*Messwerte*)
  Temperatur_Sensordaten: ARRAY[1..Anzahl_T_Sensoren] OF one_wire_T_Sensor_V6:=
  ( Name:='Sensor 1:TEST1', ID:=16#28,16#30,16#9D,16#CE,16#01,16#00,16#00,16#2F),
  ( Name:='Sensor 2:TEST2', ID:=16#28,16#FE,16#AD,16#CE,16#01,16#00,16#00,16#27);
  D1_Sensordaten: ARRAY[1..Anzahl_D1_Sensoren] OF one_wire_D1_Sensor_V6:=
  ( Name:='Sensor 1:TEST1', ID:=16#26,16#BD,16#BE,16#CF,16#00,16#00,16#00,16#62),
  ( Name:='Sensor 2:TEST2', ID:=16#26,16#63,16#BF,16#CF,16#00,16#00,16#00,16#78),
  ( Name:='Sensor 3:TEST3', ID:=16#26,16#9B,16#BF,16#CF,16#00,16#00,16#00,16#AB);
  Digitaldaten: ARRAY[1..Anzahl_B_Bausteine] OF one_wire_binaer_V6:=
  (Name:='DS2405',
  ID:=16#05,16#27,16#CF,16#20,16#00,16#00,16#00,16#C7,Belegung:=2#00000001),
  (Name:='DS2413',
  ID:=16#3A,16#A8,16#7E,16#03,16#00,16#00,16#00,16#25,Belegung:=2#00000001),
  (Name:='DS2408',
  ID:=16#29,16#05,16#95,16#09,16#00,16#00,16#00,16#3E,Belegung:=2#00000011);
```

- Startprogramm anlegen **s.o.**
- Programm anlegen , **ow\_kommunikation** '
  - Funktionsbaustein , **one\_wire**' aufrufen
  - owp und Arrays zuweisen
- Zyklische Task T#40ms einfügen und Programm , **ow\_kommunikation** ' aufrufen
- Programm für Messwertauswertung (wenn benötigt) anlegen s.o.
  - Ereignisgesteuerte Task einfügen →Ereignis: owp.werte\_neu.Sensordaten\_T\_neu
- Mit dieser Task das Programm für Messwertauswertung aufrufen

### Wenn mit DS2405/DS2408/DS2413 gearbeitet wird, dann:

- Programm für Eingangsdatenauswertung anlegen s.o.
  - Ereignisgesteuerte Task einfügen →Ereignis: owp.werte\_neu.bin\_Eingaenge
  - Mit dieser Task das Programm für Eingangsdatenauswertung aufrufen
- Programm für Ausgangsdaten anlegen s.o.
  - Ereignisgesteuerte Task einfügen →Ereignis: owp.werte\_neu.bin\_Ausgaenge
  - Mit dieser Task das Programm für Eingangsdatenauswertung aufrufen

**Zur Steuerung übertragen und Testen.**