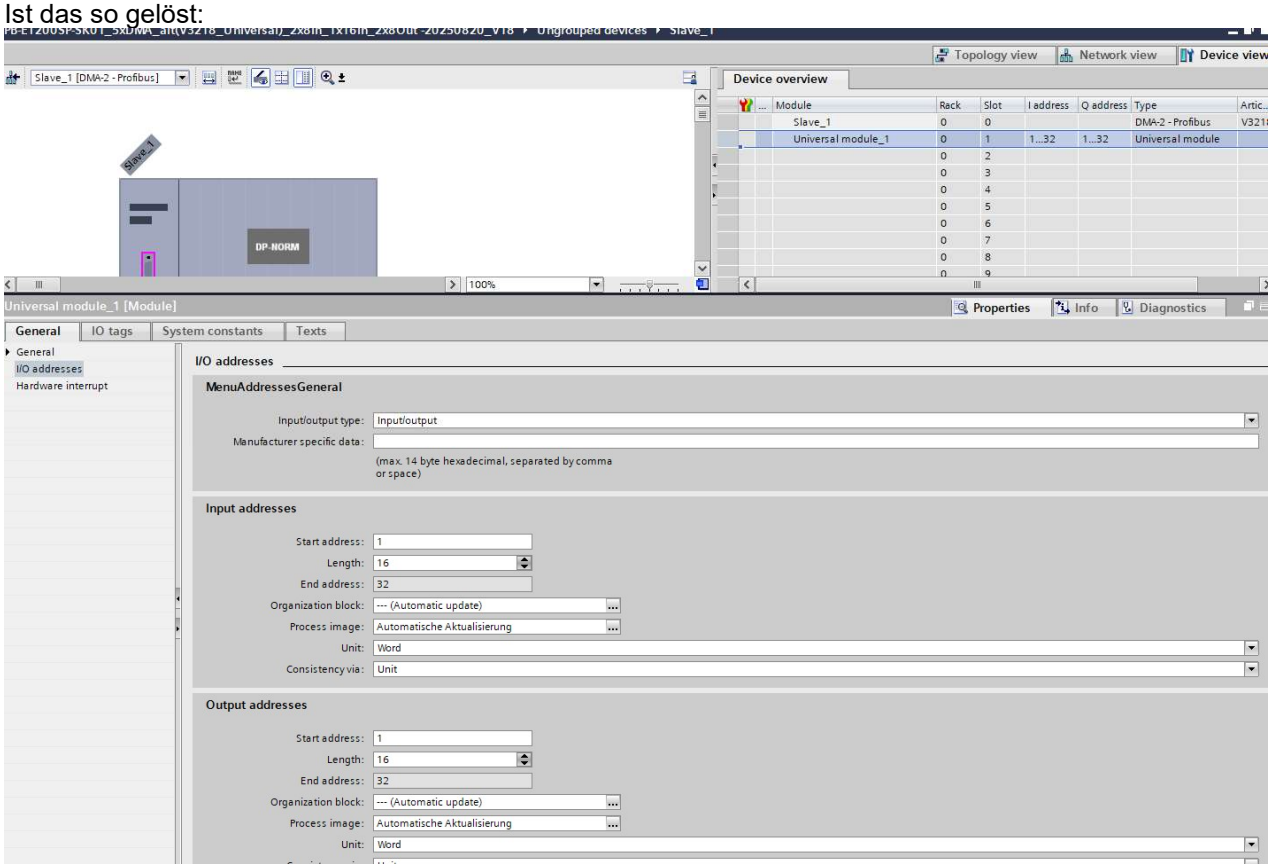
Neuffener Strasse 29
D-72636 Frickenhausen
Tel: (+49) 7025 - 911 007
Fax: (+49) 7025 - 911 008
Email: info@h-c-s-gmbh.de
www.h-c-s-gmbh.de

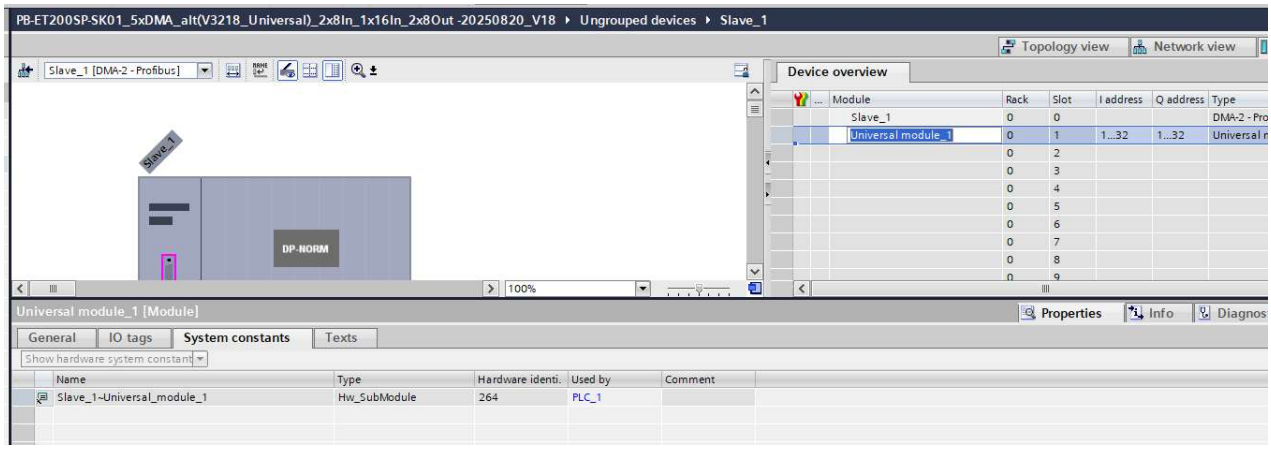
Zusätzlich mus beim Universalmodul der I/O Adressbreich angegeben werden.
Wenn man das Universalmodul reinzieht sind die I,Q Adressbereiche leer !

Unter I/O addresses muss man diesen Bereich definieren, dann kann das Programm auch den Datenaustauch vornehmen:

Beim meinem „Word Beispiel“:
“PB-ET200SP-SK01_5xDMA_alt(V3218_Universal16Word)_2x8In_1x16In_2x8Out -20250820_V18”
Ist das so gelöst:



Die HW-IO ist in diesem Fall 264



Neuffener Strasse 29
D-72636 Frickenhausen
Tel: (+49) 7025 - 911 007
Fax: (+49) 7025 - 911 008
Email: info@h-c-s-gmbh.de
www.h-c-s-gmbh.de

Im Programm dann die Wandlung Word→ Byte

```
#RET_VAL_TMP := DPRD_DAT(LADDR := #DEV_ID, RECORD => #DATA_IN_WORD);

#TMP_WORD := #DATA_IN_WORD[0];
#DATA_IN[0] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[1] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[1];
#DATA_IN[2] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[3] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[2];
#DATA_IN[4] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[5] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[3];
#DATA_IN[6] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[7] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[4];
#DATA_IN[8] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[9] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[5];
#DATA_IN[10] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[11] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[6];
#DATA_IN[12] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[13] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[7];
#DATA_IN[14] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[15] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[8];
#DATA_IN[16] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[17] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[9];
#DATA_IN[18] := #TMP_WORD_AT[0];     // High Byte
#DATA_IN[19] := #TMP_WORD_AT[1];     // Low Byte
#TMP_WORD := #DATA_IN_WORD[10];
#DATA_IN[20] := #TMP_WORD_AT[0];     // High Byte
```

Und am Ende wieder Byte → Word

```
#TMP_WORD_AT[1] := #DATA_OUT[15];     // Low Byte
#DATA_OUT_WORD[7] := #TMP_WORD;
#TMP_WORD_AT[0] := #DATA_OUT[16];     // High Byte
#TMP_WORD_AT[1] := #DATA_OUT[17];     // Low Byte
#DATA_OUT_WORD[8] := #TMP_WORD;
#TMP_WORD_AT[0] := #DATA_OUT[18];     // High Byte
#TMP_WORD_AT[1] := #DATA_OUT[19];     // Low Byte
#DATA_OUT_WORD[9] := #TMP_WORD;
#TMP_WORD_AT[0] := #DATA_OUT[20];     // High Byte
#TMP_WORD_AT[1] := #DATA_OUT[21];     // Low Byte
#DATA_OUT_WORD[10] := #TMP_WORD;
#TMP_WORD_AT[0] := #DATA_OUT[22];     // High Byte
#TMP_WORD_AT[1] := #DATA_OUT[23];     // Low Byte
#DATA_OUT_WORD[11] := #TMP_WORD;
#TMP_WORD_AT[0] := #DATA_OUT[24];     // High Byte
#TMP_WORD_AT[1] := #DATA_OUT[25];     // Low Byte
#DATA_OUT_WORD[12] := #TMP_WORD;
#TMP_WORD_AT[0] := #DATA_OUT[26];     // High Byte
#TMP_WORD_AT[1] := #DATA_OUT[27];     // Low Byte
#DATA_OUT_WORD[13] := #TMP_WORD;
#TMP_WORD_AT[0] := #DATA_OUT[28];     // High Byte
#TMP_WORD_AT[1] := #DATA_OUT[29];     // Low Byte
#DATA_OUT_WORD[14] := #TMP_WORD;


#RET_VAL_TMP := DPWR_DAT(LADDR := #DEV_ID, RECORD := #DATA_OUT_WORD);
IF (#RET_VAL_TMP <> 0) THEN    // Diskrepanz!
    #COM_ERR_TMP := 1;
```

Würde man eine Byte-Struktur definieren, dann wäre die Wandlung nicht notwendig.
Gleiches gilt auch wenn aus der GSD ein 1x16Word verwendet wid siehe hier Beispiel:

“PB-ET200SP-SK01_5xDMA_alt(V3218_1x16Word)_2x8In_1x16In_2x8Out -20250820_V18”