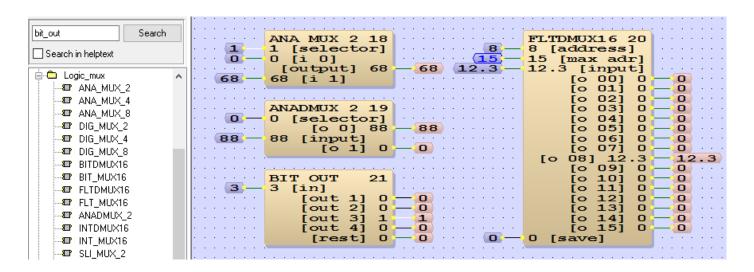


## TT200206 - FUP - Multiplexer and Demultiplexer Functions

| Note              | This Support Knowledge Base article KB is the result of a support request.                 |  |  |  |  |
|-------------------|--|--|--|--|--|
|                   | It is not part of the official documentation of DEOS AG and does not claim to be complete. |  |  |  |  |
|                   | The article is intended to support the solution of a similar problem.                      |  |  |  |  |
|                   | If you have any questions, comments or additions, please contact DEOS AG Support.          |  |  |  |  |
| Title             | Multiplexer and Demultiplexer Functions (TT200206)   |  |  |  |  |
| Object            | FUP  |  |  |  |  |
| Reference version | 2  |  |  |  |  |
| Date              | 02.2020  |  |  |  |  |
| Author            | EK   |  |  |  |  |
| Goal              | To explain the usage of the Multiplexer and Demultiplexer Function Blocks                  |  |  |  |  |

## **Content:**



## TT200206 - FUP - Multiplexer and Demultiplexer Functions

1. Multiplexer and demultiplexer modules are used frequently in our program. We've many modules for different point types and they're under the "Logic mux" group

```
ANA MUX 2 18
                                                                          FT.TDMUX16
                                                                                        20
                                    selector
                                                                          *address
Search in helptext
                          150
                                   *i
                                       o
                                                                          *max adr
                                                         D26
                                                                          *input
                                           *output
🖮 🗁 Logic_mux
                         I51
    ANA MUX 2
                                                                                         01
                                                                                        02
                                                                                     *0
    -- ■P ANA_MUX_4
                                   XUMCIANA
                                               2
                                                  19
    ANA_MUX_8
                         148
                                                                                     *0
                                                                                        04
05
                                   *selector
    DIG_MUX_2
                                                   o
                                                         D27
                                                                                    *0
                                                                                         06
07
    - DIG_MUX_4
                          T52
                                   *input
                                                         D28
    ■ DIG_MUX_8
                                                                                    *0
                                                                                         08
09
    ■ BITDMUX16
    -₽ BIT_MUX16
                                   BIT
                                         OUT
                                                  21
                                                                                    *0
*0
                                                                                        10
                         I54
    ₽ FLTDMUX16
                                                                                        12
13
                                            *out
                                                         D29
D30
D31
                                                                                     *0
    FLT MUX16
    ■ ANADMUX_2
                                                   3
                                                                                     *0
                                             *out
    -₽ INTDMUX16
                                                         D33
    INT MUX16
    -- SLI_MUX_2
```

2. First, we look at the "ANA\_MUX\_2" module. This module can be used to select 2 floating points value based on a digital input and send it to the output. For example, we can use it to send the PID output (i\_1) to the cooling valve (output) when the AHU (selector) is ON, and close the valve (i\_0 = 0) when AHU is OFF

```
ANA MUX 2 18
1 [selector]
0 0 [i 0]
0 [output] 62
62 62 [i 1]

ANA MUX 2 18
0 [selector]
0 0 [i 0]
0 [output] 0
62 62 [i 1]
```

3. The second module is "ANADMUX\_2" which is the opposite to "ANA\_MUX\_2". This module sends the input to 2 output respectively, based on the digital input. For example, we can send the PID output (input) to the cooling valve (o\_0) or heating valve (o\_1), based on the operating mode of the AHU (selector)

4. For the "BIT\_OUT" module (under "Logic\_dig"), one of the outputs (out\_1 to out\_4) will turn on based on the input. For example, when in=2, out\_2 will become 1 and the other outputs are 0. If in=0 or in>4, all outputs are 0. This module can be used for mode operation (e.g. auto/on/off), to perform different actions based on the mode input

```
BIT OUT
                                BIT
   [in]
     out
                                    [out
                                           1]
2]
                                               0
                                                     0
                                               ō
                                                     ō
                                     ľout
                                           3]
               0
                     o
                                               o
                                                     0
      out
     out
                                     out
                                                     0
```

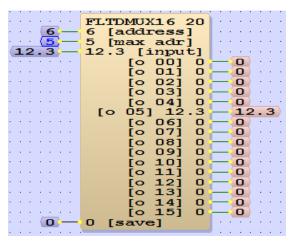
5. This module can be used in cascaded mode, to handle more than 4 outputs, like this

```
BIT
     OUT
                               BIT
                                   [in]
   [in]
                      0
                                                      0
                                     [out
     out
                ō
           3j
4]
                                           3]
4]
                                                01
                                                      0
                0
                      0
      out
                                      out
```

6. The last module to show you is "FLTDMUX16". The module sends the floating point input to one of the outputs (o\_00 to o\_15) based on the address input

|              | <del> </del> |             |         |              |                     |
|--------------|--------------|-------------|---------|--------------|---------------------|
|              | FLTDMUX16 20 |             |         | FLTDMUX16 20 |                     |
| 8            |              |             | (6)—(   | 6 [address]  |                     |
|              |              |             |         |              |                     |
| 15 [max adr] |              |             | 15      |              |                     |
| (12.3)—      | 12.3 [input] |             | (12.3)— | 12.3 [input] | · · · · · · ·       |
|              | [0 001 0     | O · · ·     |         | [0 00] 0     | <b>─(0)</b> · · · · |
|              | lo 011 0     |             |         | [o 01] 0     | <b>—(0)</b>         |
|              | io 02i 0     | O · · · ·   |         | [o 02] 0     | <b>— 0</b> · · · ·  |
|              | io 03i o     |             |         | [o 03] 0     |                     |
|              | [0 04] 0     |             |         | [o 04] 0     | <b>—</b> • • • • •  |
|              | io 051 0     |             |         | [o 05] 0     |                     |
|              | io 06i 0     |             |         | [0 06] 12.3  | 12.3                |
|              | io 071 0     |             |         | [o 071 0     |                     |
|              | [0 08] 12.3  | <b>12.3</b> |         | 0 [80 0]     | <b>—</b> • • • • •  |
|              | 0 160 01     |             |         | 0 [60 oj     |                     |
|              | [0 10] 0     |             |         | [o 10] 0     | <u> </u>            |
|              | io 111 0     |             |         | [o 11] 0     | O                   |
|              | [0 12] 0     | O           |         | [o 12] 0     | <b>—</b> • • • •    |
|              | [o 13] 0     |             |         | [o 13] 0     | O                   |
|              | [0 14] 0     |             |         | [o 14] 0     | <b>—</b> • • • • •  |
|              | [o 15] 0     |             |         | [o 15] 0     | O                   |
|              | 0 [save]     |             | 🕡 —     | 0 [save]     |                     |
|              |              |             |         |              |                     |

7. If you don't have 16 outputs, you can limit the number of outputs using the "max\_adr" input. So, when the "address" is greater than the "max\_adr", the "max\_adr" output will be set, like below



8. If the input "save" is set to 1, then you can send different values to different outputs, by changing the "address" and the "input" at the same time, like below. To reset the "old" outputs, set "save" to 0

