# KomUniVal

## KomUniVal structure

```
struct KomUniVal {
   unsigned short uvStatus;
   TLimitStatus   uvLimitStatus;
   unsigned int   uvProcAlarmStatus;
   TValueType     uvValType;
   unsigned short uvFlags;
   D2Time         uvValTime;
   D2Time         uvProcAlarmTime;
   TBVal          uvBoval;
   int            uvIntval;
   double         uvRealval;
   TStVal         uvStationval;
   D2Time         uvTmAval;
   double         uvTmRval;
   TQVal          uvQval;
   char          *uvTxtval;
 };
```

Structure *KomUniVal* transfers a value and status of D2000 system object (I/O tag, station, line). The meaning of individual items of the structure:

### uvStatus

It can get a combination of the following values:

```
#define SB_Val_Valid     0x0001
 #define SB_Pa_Alarm      0x0002
 #define SB_Pa_NoAck      0x0004
 #define SB_Pa_Blocked    0x0008
 #define SB_Val_Weak      0x0010
 #define SB_Val_NoAck     0x0020
 #define SB_Val_Transient 0x0040
 #define SB_Val_Default   0x0080
 #define SB_Val_Manual    0x0100
 #define SB_Pa_Critical   0x0200
```

Implementation of a communication protocol can change the following values for objects of I/O tag type:

- SB_Val_Valid – validity of I/O tag value
- SB_Val_Weak – I/O tag value is suspicious (weak)

Other values are changed internally by the communication process.

### uvLimitStatus

```
typedef enum {LS_InLimit, LS_VL_Limit, LS_L_Limit, LS_H_Limit, LS_VH_Limit, LS_LimitsProblem} TLimitStatus;
```

The parameter marks limit states of I/O tag value. Implementation of a protocol sets up this parameter only in case of the value ST_SOURCE_LIMITS of the parameter *Stat* call-back of the function PointNewValue.

Possible values:

- LS_InLimit - value is in limits
- LS_VL_Limit - value is below the lowest limit (only for the I/O tag types Ai, Ao, Ci, Co)
- LS_L_Limit - value is below the low limit (only for the I/O tag types Ai, Ao, Ci, Co)
- LS_H_Limit - value is above the high limit (only for the I/O tag types Ai, Ao, Ci, Co)
- LS_VH_Limit - value is above the highest limit (only for the I/O tag types Ai, Ao, Ci, Co)
- LS_LimitsProblem - limits problem (crossing the values or a dynamic limit value is invalid)

### uvProcAlarmStatus

Process alarms attributes. Implementation of a protocol must not change this parameter.

## uvValType

```
typedef enum {VT_NAN, VT_Bo, VT_Int, VT_Re, VT_Di, VT_Do, VT_De, VT_Ai, VT_Ao, VT_Ae, VT_Ci, VT_Co, VT_Ce,
VT_St, VT_Li, VT_Al, VT_Pr, VT_TmA, VT_TmR, VT_TiA, VT_ToA, VT_TiR, VT_ToR, VT_Txt, VT_Arr, VT_Qi, VT_Unused1,
VT_TxtI, VT_TxtO} TValueType;
```

Object value type. The important types for communication:

- VT_Ai - real input
- VT_Ao - real output
- VT_Ci - integer input
- VT_Co - integer output
- VT_Di - digital input
- VT_Do - digital output
- VT_TiA - absolute time input
- VT_ToA - absolute time output
- VT_TiR - relative time input
- VT_ToR - relative time output
- VT_Qi - quaternary input
- VT_TxtI - text input
- VT_TxtO - text output

This value must not be changed!

## uvFlags

```
#define VF_A 0x0001
 #define VF_B 0x0002
 #define VF_C 0x0004
 #define VF_D 0x0008
 #define VF_E 0x0010
 #define VF_F 0x0020
 #define VF_G 0x0040
 #define VF_H 0x0080
 #define VF_I 0x0100
 #define VF_J 0x0200
 #define VF_K 0x0400
 #define VF_L 0x0800
 #define VF_M 0x1000
 #define VF_N 0x2000
 #define VF_O 0x4000
 #define VF_P 0x8000
```

Implementation can optionally sets up combinations of 16 flags ABCDEFGHIJKLMNOP of I/O value. Values of the flags are from VF_A up to VF_P.

## uvValTime

Current time of current value.

## uvProcAlarmlTime

Current time of the last change of process alarms flag. Do not change!

## uvBoval

Current value of digital input or output Di and Do.

Values:

- D_False - FALSE
- D_True - TRUE
- D_Oscillate - oscillating value – do not set, it is analyzed and set by the process D2000 KOM!

## uvIntval

Current value of integer input or output Ci and Co.

## uvRealval

Current value of real input or output Ai and Ao.

## uvStationval

Station status – not used, do not change!

```
typedef enum {ST_ON, ST_OFF, ST_COMERR, ST_HARDERR, ST_SIMUL, ST_WAIT} TStVal;
```

## uvTmAval

Current value of absolute time input and output TiA and ToA.

## uvTmRval

Current value of relative time input an output TiR and ToR.

## uvQval

Current value of quadrat input Qi.

```
typedef enum {Q_Trans, Q_Off, Q_On, Q_Err, Q_Oscillate} TQVal;
```

Possible values are Q_Trans, Q_Off, Q_On, Q_Err or Q_Oscillate. Do not use the value Q_Oscillate, it is set by the process D2000 KOM in case of evaluation of oscillation.

## uvTxtval

Pointer to a string of current value of text types TxtI a TxtO.

**Note:** The communication process allocates its own copies of text variables' values after calling the function PointNewValue.

**Related pages:**

D2000 KomAPI - interface structures