

ISOBUS Virtual Terminal: User Interface and Objects (2)

Franz Höpfinger

Contents

	myst: enable_extensions: ["colon_fence"] html_meta: "description lang=en": "metadata description" "description lang=de": "metadata description" "description lang=fr": "description des métadonnées" "keywords": "Sphinx, MyST" "property=og:locale": "de"	11
1	Wiki 2: ISOBUS Virtual Terminal (VT & HMI user interfaces) .	13
1.1	Master Schools at Munich's East Station	13
1.1.1	 This is in the curriculum (Framework Curriculum 2025)	13
1.2	Contents	13
2	AuxAssignOfApp	15
3	Insert Bitmaps	17
3.1	Find PNG	17
3.2	ImageMagick®	17
3.2.1	1. All files under img_original will be cropped by 2 pixels on all sides	17
3.2.2	2. All files under img_cropped are scaled to 64x64	17
3.2.3	3. Conversion to 2 colors (monochrome)	18
3.2.4	4. Conversion to BMP format	18
3.3	Paint	18
3.3.1	Convert PNG to BMP	18
3.3.2	Scaling BMP	19
3.3.3	BMP Scaling	19
3.3.4	Reduce BMP Color Depth	19
3.4	Insert BMP into ISO Designer	19
3.4.1	 Related topic subpages on ms-muc-docs.de	20
4	Entries from students	21
5	Welcome to the ISOBUS-VT-Objects wiki	23
5.1	 Podcast	23
6	ISOBUS Object IDs	25
7	ISOBUS Object Versions	31
8	ISOBUS Objects	33
8.1	Programming	33
8.2	VT Levels	34
9	ISOBUS Wiki Links	35
10	Open Jetter ISO Designer Workspace	37

11	Jetter ISO-Designer	41
12	Empty Objects (Minimum Child Objects)	45
12.1	Objects that MUST NOT be empty (Minimum 1 Child)	45
12.2	Objects that may be empty (Minimum 0 children)	45
12.2.1	Special Feature: Input List (ID 10)	46
13	Object Hierarchy (Parent-Child Relationships)	47
13.1	Parent: Working Set object (ID 0)	47
13.2	Parent: Data Mask object (ID 1)	47
13.3	Parent: Alarm Mask object (ID 2)	48
13.4	Parent: Container object (ID 3)	49
13.5	Parent: Window Mask object (ID 34)	50
13.6	Parent: Soft Key Mask object (ID 4)	50
13.7	Parent: Key object (ID 5)	51
13.8	Parent: Button object (ID 6)	51
13.9	Parent: Key Group object (ID 35)	52
13.10	Parent: Input List object (ID 10)	52
13.11	Parent: Output List object (ID 37)	52
13.12	Parent: Auxiliary Function Type 1 object (ID 29)	53
13.13	Parent: Auxiliary Input Type 1 object (ID 30)	54
13.14	Parent: Auxiliary Function Type 2 object (ID 31)	54
13.15	Parent: Auxiliary Input Type 2 object (ID 32)	54
13.16	Parent: Object Label graphic representation	55
13.17	Parent: Object Label Reference List object (ID 40)	55
13.18	Parent: Animation object (ID 44)	57
14	Invalid Object Pools	59
14.1	1. Structural Integrity (Parsing Errors)	59
14.2	2. Referencing Errors (Dangling Pointers)	59
14.3	3. Logical Errors & Recursion	59
14.4	4. Specific Object Rules (Hard Constraints)	59
14.5	5. Resource Limits	60
14.6	Troubleshooting tips	60
15	Creating Vector Graphics	61
15.1	CorelDRAW X5	61
15.2	Solid Edge	61
15.3	LibreCAD	61
15.4	ISO-Designer	62
15.5	Alternative with Autodesk Fusion 360 and Plugin	63
15.5.1	Use Fusion 360	63

16	 Videos	67
16.1	ISO Designer: German	67
16.2	ISO designer: English	67
17	 This is on the curriculum: ISOBUS Virtual Terminal & HMI user interfaces	69
17.1	 Relevance of the Virtual Terminal (ISO 11783-6) in the 2025 Curriculum Framework	69
17.2	 Curriculum Modules & Focus Areas	69
17.2.1	1. Part I – Learning Unit 1.4: Automated Functions (80 units)	69
17.2.2	2. Part II – Learning Unit 1.1: Systems Engineering (120 hours)	69
17.2.3	3. Part II – Learning Unit 1.4: Technical Documentation & Drawings (80 hours)	69
17.3	 Curriculum topics in this wiki	69
17.3.1	 Related topic subpages on ms-muc-docs.de	70
17.4	 Main portal & direct links to the 6 wikis	70
17.4.1	 PDF Downloads for All Wikis	70
18	logiBUS®	73
18.1	 Podcast	73
18.2	 Video	73
18.2.1	 Related topic subpages on ms-muc-docs.de	73
19	Objects	75
19.1	Attributes	75
19.1.1	Recommended Reading	75
19.1.2	 Podcast	76
19.2	Issue	76
19.3	Auxiliary Functions	76
19.4	Controls	76
19.5	Pictures	76
19.6	Container	76
19.6.1	 Podcast	76
19.7	Input	77
19.8	External Objects	77
19.9	Functions	77
19.10	Masks	77
19.10.1	 Podcast	77
19.11	Specials & Miscellaneous	77
19.12	Variables	77
19.12.1	Recommended reading	77
20	Reference Tables (Annex A)	79
20.1	Command Overview (ISO 11783-6 Table A.5)	79
20.2	Event IDs (ISO 11783-6 Table A.3)	79

20.3	Standard Color Palette (ISO 11783-6 Table A.4)	80
20.3.1	Transparency	81
21	Isobus objects	83
21.1	ID 0 – Working set – ISO 11783-6 – B.1	83
21.1.1	Key properties (according to ISO 11783-6, Annex B.1)	83
21.1.2	Implementation	85
21.2	ID 1 – Data mask – ISO 11783-6 – B.2	85
21.2.1	Attributes and Record Format (Table B.4)	85
21.2.2	Events (Table B.3)	87
21.2.3	Behavior and Limitations	87
21.2.4	Implementation Importance	87
21.3	ID 2 – Alarm Mask – ISO 11783-6 – B.3	87
21.3.1	Attributes and Record Format (Table B.6)	87
21.3.2	Events (Events - Table B.5)	88
21.3.3	Behavior with Multiple Alarms	89
21.3.4	Implementation Implementation Implementation Implementation Masks should be used sparingly and only for genuine warnings, as they interrupt the operator's workflow. It is important to select the appropriate audible alert (AID 4) to guide attention without stressing the operator	89
21.4	ID 3 – Container – ISO 11783-6 – B.4	89
21.4.1	Attributes and Record Format (Table B.8)	89
21.4.2	Events (Table B.7)	90
21.4.3	Practical Use	90
21.4.4	Implementation Implications	91
21.5	ID 4 – Soft Key Mask – ISO 11783-6 – B.5	92
21.5.1	Attributes and Record Format (Table B.10)	92
21.5.2	Events (Table B.9)	92
21.5.3	Interaction with Data Masks	93
21.5.4	Importance for Implementation	93
21.5.5	 Podcast	93
21.6	ID 5 – Key (Soft Key) – ISO 11783-6 – B.6	93
21.6.1	 Podcast	94
21.6.2	Events (Events - Table B.11)	95
21.6.3	The Key Code	95
21.6.4	Implementation Implementation Implications	95
21.7	ID 6 – Button – ISO 11783-6 – B.7	95
21.7.1	Annex B.7 – Button	96
21.7.2	Overview of the Button Object	96
21.7.3	Attributes and Record Format (Table B.14)	96
21.7.4	Events (Events - Table B.13)	98
21.7.5	Role of the Button Object in the Virtual Terminal	98
21.7.6	Implementation implications	98
21.7.7	Note	98
21.8	ID 7 – Input Boolean – ISO 11783-6 – B.8.2	100
21.8.1	Attributes and Record Format (Table B.16)	100
21.8.2	Events (Table B.15)	101
21.8.3	Implementation Implementation Implementation	101
21.9	ID 8 – Input string – ISO 11783-6 – B.8.3	101
21.9.1	Attributes and Record Format (Table B.17)	101
21.9.2	Functionality and Options	103
21.9.3	Events (Table B.15)	103

21.9.4 Implementation Implementation Implementation Significance	103
21.10 ID 9 – Input number – ISO 11783-6 – B.8.4	104
21.10.1 Attributes and Record Format (Table B.18)	104
21.10.2 Scaling Logic	105
21.10.3 Validation	105
21.10.4 Events (Table B.15)	106
21.10.5 Real-Time Editing (AID 15, Bit 1)	106
21.10.6 Implementation Implications	106
21.11 ID 10 – Input list – ISO 11783-6 – B.8.5	106
21.11.1 Attributes and Record Format (Table B.20)	106
21.11.2 Functionality and Display	107
21.11.3 Events (Events - Table B.19)	107
21.11.4 Implementation Implications	107
21.12 ID 11 – Output string – ISO 11783-6 – B.9.2	107
21.12.1 Attributes and Record Format (Table B.22)	107
21.12.2 Functionality and Special Features	109
21.12.3 Static Text vs. Dynamic Variable	109
21.12.4 Events (Events - Table B.21)	109
21.12.5 Implementation Implementation Implementation Implementation (IM)	109
21.13 ID 12 – Output number – ISO 11783-6 – B.9.3	109
21.13.1 Attributes and Record Format (Table B.23)	109
21.13.2 Scaling Logic	111
21.13.3 Formatting Options	111
21.13.4 Events	111
21.13.5 Implementation Implementation Implementation	111
21.14 ID 13 – Output line – ISO 11783-6 – B.10.2	111
21.14.1 Attributes and Record Format (Table B.27)	111
21.14.2 Functionality and Geometry	112
21.14.3 Events (Events - Table B.26)	112
21.14.4 Implementation Implications	112
21.15 ID 14 – Output Rectangle – ISO 11783-6 – B.10.3	113
21.15.1 Attributes and Record Format (Table B.29)	113
21.15.2 Appearance and Properties	113
21.15.3 Geometric Calculation	114
21.15.4 Events (Table B.28)	114
21.15.5 Implementation Implications	114
21.16 ID 15 – Output Ellipse – ISO 11783-6 – B.10.4	114
21.16.1 Attributes and Record Format (Table B.31)	114
21.16.2 Ellipse Types and Geometry	115
21.16.3 Angle Calculation (Important!)	115
21.16.4 Events (Events - Table B.30)	115
21.16.5 Implementation Implications	116
21.17 ID 16 – Output polygon – ISO 11783-6 – B.10.5	116
21.17.1 Attributes and Record Format (Table B.33)	116
21.17.2 Polygon Types and Fill Rules	117
21.17.3 Geometry and Points	117
21.17.4 Events (Table B.32)	117
21.17.5 Implementation Implementation Implementation	117
21.18 ID 17 – Output meter – ISO 11783-6 – B.11.2	117
21.18.1 Attributes and Record Format (Table B.35)	117
21.18.2 Functionality and Display	119
21.18.3 Deflection Direction (AID 5, Bit 3)	119
21.18.4 Events (Table B.34)	119

21.18.5 Implementation Implications	119
21.19 ID 18 – Output Linear Bar Graph – ISO 11783-6 – B.11.3	119
21.19.1 Attributes and Record Format (Table B.37)	119
21.19.2 Functionality and Options	122
21.19.3 Scaling	122
21.19.4 Events (Events - Table B.36)	122
21.19.5 Implementation Implications	122
21.19.6  Podcast	122
21.20 ID 19 – Output Arched Bar Graph – ISO 11783-6 – B.11.4	123
21.20.1 Attributes and Record Format (Table B.39)	123
21.20.2 Functionality and Geometry	124
21.20.3 Events (Table B.38)	124
21.20.4 Implementation Implications	124
21.21 ID 20 – Picture graphic – ISO 11783-6 – B.12.2	125
21.21.1 Attributes and Record Format (Table B.41)	125
21.21.2 Functionality and Display	126
21.21.3 Events (Events - Table B.40)	126
21.21.4 Implementation Implementation Importance	126
21.22 ID 21 – Number variable – ISO 11783-6 – B.13.2	126
21.22.1 Attributes and Record Format (Table B.43)	126
21.22.2 Functionality and Referencing	127
21.22.3 Events (Table B.42)	127
21.22.4 Implementation Implications	127
21.23 ID 22 – String variable – ISO 11783-6 – B.13.3	127
21.23.1 Attributes and Record Format (Table B.44)	127
21.23.2 Functionality and Special Features	128
21.23.3 Referencing and Updating	128
21.23.4 Events (Table B.42)	128
21.23.5 Implementation Significance	128
21.24 ID 23 – Font attributes – ISO 11783-6 – B.14.2	128
21.24.1 Attribute and Record Format (Table B.46)	128
21.24.2 Font Sizes and Render Modes	129
21.24.3 Events (Table B.45)	129
21.24.4 Implementation Implications	129
21.25 ID 24 – Line attributes – ISO 11783-6 – B.14.3	130
21.25.1 Attribute and Record Format (Table B.48)	130
21.25.2 Line Thickness and Appearance	130
21.25.3 Line Style (Line Art - AID 3)	130
21.25.4 Events (Table B.47)	131
21.25.5 Implementation Implementation Implementation	131
21.26 ID 25 – Fill attributes – ISO 11783-6 – B.14.4	131
21.26.1 Attribute and Record Format (Table B.50)	131
21.26.2 Fill Types and Logic	132
21.26.3 Using Fill Patterns (Important!)	132
21.26.4 Events (Table B.49)	132
21.26.5 Implementation Implementation Importance	132
21.27 ID 26 – Input attributes – ISO 11783-6 – B.14.5	132
21.27.1 Attribute and Record Format (Table B.52)	132
21.27.2 Functionality and Validation	133
21.27.3 Events (Table B.51)	133
21.27.4 Implementation Implications	133
21.28 ID 27 – Object pointer – ISO 11783-6 – B.15	133

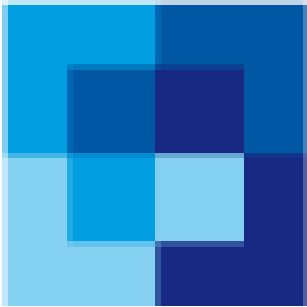
21.28.1	Attributes and Record Format (Table B.55)	133
21.28.2	Functionality and Application	134
21.28.3	Events (Table B.54)	134
21.28.4	Implementation Implementation Implementation Implementation Implementation Implementation Implementation (IMA) significantly reduces the complexity of the screen control. Instead of manually managing many objects using Hide/Show, the ECU only needs to change a single ID in the pointer. This saves CAN bus bandwidth and simplifies the program logic on the machine control	134
21.29	ID 28 – Macro – ISO 11783-6 – B.16	134
21.29.1	Attributes and Record Format (Table B.56)	134
21.29.2	Functionality and Structure	135
21.29.3	Available Macro Commands (Excerpt)	135
21.29.4	Events	135
21.29.5	Implementation Implementation Significance	135
21.30	ID 29 – Auxiliary Function Type 1 – ISO 11783-6 – J.4.2	135
21.30.1	Attributes and Record Format (Table J.1)	136
21.30.2	Functionality	136
21.31	ID 30 – Auxiliary Input Type 1 – ISO 11783-6 – J.4.4	136
21.31.1	Attributes and Record Format (Table J.3)	136
21.31.2	Functionality	137
21.32	ID 31 – Auxiliary Function Type 2 – ISO 11783-6 – J.4.3	137
21.32.1	Attributes and Record Format (Table J.2)	137
21.32.2	Special Features	138
21.33	ID 32 – Auxiliary Input Type 2 – ISO 11783-6 – J.4.5	139
21.33.1	Attributes and Record Format (Table J.4)	139
21.33.2	Functionality	140
21.34	ID 33 – Auxiliary Control Designator Type 2 Object Pointer – ISO 11783-6 – J.4.7	140
21.34.1	Attributes and Record Format (Table J.6)	140
21.34.2	Events (Events - Table J.7)	140
21.34.3	Benefits for the Developer	141
21.35	ID 34 – Window Mask – ISO 11783-6 – B.19	141
21.35.1	Attributes and Record Format (Table B.61)	141
21.35.2	Events (Events - Table B.60)	143
21.36	ID 35 – Key Group – ISO 11783-6 – B.20	143
21.36.1	Attributes and Record Format (Table B.63)	143
21.36.2	Events (Table B.62)	144
21.36.3	Meaning and Functionality	144
21.37	ID 36 – Graphics Context Object – ISO 11783-6 – B.18	144
21.37.1	Attributes and Record Format (Table B.59)	144
21.37.2	Functionality and Structure	146
21.37.3	Graphics Context Commands	146
21.37.4	Events (Table B.58)	147
21.37.5	Application Example	147
21.38	ID 37 – Output List – ISO 11783-6 – B.9.4	147
21.38.1	Attributes and Record Format (Table B.25)	147
21.38.2	Functionality	148
21.38.3	Events (Table B.24)	148
21.39	ID 38 – Extended Input Attributes – ISO 11783-6 – B.14.6	148
21.39.1	Attribute and Record Format (Table B.53)	148
21.39.2	Functionality	149
21.40	ID 39 – Colour Map – ISO 11783-6 – B.17	149

21.40.1	Attributes and Record Format (Table B.57)	149
21.40.2	Functionality	150
21.40.3	Note on VT Version 6	150
21.41	ID 40 – Object Label Reference List – ISO 11783-6 – B.21	150
21.41.1	Attributes and Record Format (Table B.64)	150
21.41.2	Meaning and Functionality	151
21.42	ID 41 – External Object Definition – ISO 11783-6 – B.22	151
21.42.1	Attributes and Record Format (Table B.66)	151
21.42.2	Functionality	152
21.42.3	Events (Table B.65)	152
21.43	ID 42 – External Reference NAME – ISO 11783-6 – B.23	152
21.43.1	Attributes and Record Format (Table B.68)	152
21.43.2	Functionality	152
21.43.3	Events (Table B.67)	152
21.44	ID 43 – External Object Pointer – ISO 11783-6 – B.24	153
21.44.1	Attributes and Record Format (Table B.70)	153
21.44.2	Functionality and Rules	153
21.44.3	Events (Events - Table B.69)	153
21.44.4	Application Example	154
21.45	ID 44 – Animation – ISO 11783-6 – B.25	154
21.45.1	Attributes and Record Format (Table B.72)	154
21.45.2	Functionality	155
21.45.3	Events (Events - Table B.71)	155
21.45.4	Recommendation	155
21.46	ID 45 – Colour Palette – ISO 11783-6 – B.26	156
21.46.1	Attributes and Record Format (Table B.73)	156
21.46.2	Functionality	156
21.46.3	Difference to the Colour Map (ID 39)	156
21.46.4	Events	156
21.47	ID 46 – Graphic Data (PNG) – ISO 11783-6 – B.27	156
21.47.1	Attributes and Record Format (Table B.74)	157
21.47.2	Special Features	157
21.47.3	Usage	157
21.48	ID 47 – Working Set Special Controls – ISO 11783-6 – B.29	157
21.48.1	Attributes and Record Format (Table B.78)	157
21.48.2	Meaning and Functionality	158
21.48.3	Events (Table B.77)	158
21.49	ID 48 – Scaled Graphic – ISO 11783-6 – B.28	158
21.49.1	Attributes and Record Format (Table B.76)	158
21.49.2	Events (Events - Table B.75)	159
21.50	ISOBUS VT Objects	160
21.50.1	List of objects	160
22	Lehrplan praxis	161
22.1	 Input Attributes, Key Events & Interaction Diagnostics	161
22.1.1	1. Interactive Objects & Input Fields	161
22.1.2	2. Error Diagnostics on VT Interfaces	161
22.2	 ISOBUS VT Object Pools & HMI (ISO 11783-6)	161
22.2.1	1. Fundamentals of the Virtual Terminal (VT / UT)	161
22.2.2	2. Core Objects in the Object Pool	162
22.3	 Mask Layout, Softkeys & Ergonomics	162

22.3.1	1. Data Mask Design	162
22.3.2	2. Softkey Structures	162

0.1 **myst: enable_extensions: ["colon_fence"] html_meta: "description lang=en": "metadata description" "description lang=de": "metadata description" "description lang=fr": "description des métadonnées" "keywords": "Sphinx, MyST" "property=og:locale": "de"**

1. Wiki 2: ISOBUS Virtual Terminal (VT & HMI user interfaces)



Meisterschulen am Ostbahnhof

1.1 Master Schools at Munich's East Station

1.1.1 This is in the curriculum (Framework Curriculum 2025)

The ISOBUS Virtual Terminal (VT / UT according to ISO 11783-6) is the central human-machine interface (HMI) for operating, configuring, and visualizing all networked attachments (Part I LE 1.4 and Part II LE 1.1 & LE 1.4).

-  This is in the curriculum – Virtual Terminal & HMI in the curriculum
 -  ISOBUS VT Object Pools & HMI (ISO 11783-6, LE 1.4 & LE 1.1)
 -  Screen Layout, Softkeys & Ergonomics (LE 1.4 & LE 1.4)
 -  Input Attributes, Key Events & Interaction Diagnostics (LE 2.2 & LE 1.3)

Welcome to the ISOBUS Virtual Terminal documentation.

This documentation is part of the knowledge database of the master craftsman schools at Munich East Station.

Useful Links:

-  Main Menu
-  Super Search (All Wikis)
-  Download PDF Handbook

!!! note This project is under active development.

1.2 Contents

 Main Menu |  Super Search (All Wikis) |  Quick Search (IEC 61499)

2. AuxAssignOfApp

AuxAssignOfApp.txt It can be found on the logiBUS® web server:

The screenshot shows a web browser at 192.168.4.1 displaying the 'ESP32 File Server' interface. It includes upload controls, a file list table, and status information.

ESP32 File Server

Upload a file: Keine ausgewählt

Set path on server:

Set new directory:

Current Directory: /data/

Name	Type	Size	Version	Chip	Update	Delete
DefaultPool.iop	file	22 kB		N / A		Delete
test_FORTE_PC.fboot	file	353 kB		N / A		Delete
AuxAssignOfApp.txt	file	0 kB		N / A		Delete

384.0 kB of 8088.0 kB used (95 % free), 7704.0 kB remaining

Show [Version](#) of running application.

File format:

The screenshot shows the content of the file AuxAssignOfApp.txt, which contains two lines of hexadecimal data.

```
339 2 2 31002 28001 cfb6d6d2a008100a0
339 2 2 31003 28003 cfb6d6d2a008100a0
```

339 2 2 31002 28001 cfb6d6d2a008100a0 339 2 2 31003 28003 cfb6d6d2a008100a0

This means

		Description
339		Manufacturer Code
2		Model Identification Code
2		Aux Type (in this case, Boolean non-latching)
31002		Aux Function Object ID on Implement
28001		Aux Input Object ID on Joystick
cfb6d6d2a008100a0		Joystick ISO Name

3. Insert Bitmaps

!!! Note ISO 11783-6 does not support PNG graphics for VT3/UT2. See also [ISOBUS-Objects-Versions](#)]

3.1 Find PNG

<https://www.iso.org/obp/>

Download PNG

Right-click on the icon and save it, ideally in the following folder: ... C:\git\ms\4diac_training1\Valve Control\ISO-DesignerProjects\Workspace\DefaultPool\img

Other folders are possible, but this one is recommended for the sake of organization.

3.2 ImageMagick®

<https://imagemagick.org/> can simplify the conversion process; it can convert hundreds of images with a single mouse click, if needed. ... <https://imagemagick.org/script/download.php#windows> Download for Windows

at:

https://github.com/Meisterschulen-am-Ostbahnhof-Munchen/4diac_training1/tree/main/Ventilsteuerung/ISO-DesignerProjects/Workspace_TECU/DefaultPool/img/script

You will find a script that (assuming ImageMagick is installed) executes the following processes when double-clicked:

Original: ...



by https://www.iso.org/obp/graphics/grs/343178e8-8b69-4dd7-83c6-cee7c66b28a4_200.png

3.2.1 1. All files under `img_original` will be cropped by 2 pixels on all sides

To the folder: `img_original`

Before		After
	->	
200x200		196x196

This removes the 2px crop marks from <https://www.iso.org/obp/>. These are generally undesirable for aesthetic reasons.

... The cropped images are then located under “`img_cropped`”

3.2.2 2. All files under `img_cropped` are scaled to 64x64

to the folder: `img_cropped`

before		after
--------	--	-------

	-->	
196x196		64x64

The cropped images are then located under “img_resized”

3.2.3 3. Conversion to 2 colors (monochrome)

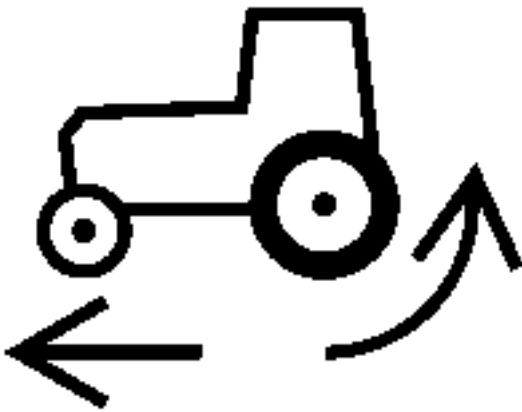
to the folder: [img_resized](#)

before		after
	-->	
64x64		64x64

The images are then located under “img_monochrome”

3.2.4 4. Conversion to BMP format

to the folder: [img_monochrome](#)

before		after
	-->	
64x64		64x64

The images are then located under “img”
to the folder: [img](#)

3.3 Paint

Converting PNG to BMP is very easy with Microsoft Paint, which is included in every Windows installation. However, if you want to convert many files, the effort becomes considerable.

3.3.1 Convert PNG to BMP

Name the bitmap appropriately; ISO filenames are often not very descriptive.

(no spaces, no umlauts) See: <https://meisterschulen-am-ostbahnhof-munchen-docs.readthedocs.io/projects/visual-programming-languages-docs/de/latest/Allgemeines.html#namen>

3.3.2 Scaling BMP

3.3.3 BMP Scaling

Size matters:

where	size			
Softkey	72x72 pixels			
Datamask	free			
CCI A3	64x64 pixels			

Thoughts:

80x80 → as a button, resulting in an inner area of 72x72

72x72 → as a button, creates an inner area of 64x64

!!! Note

Tip: Use these sizes for buttons as well; it creates a clean look.

For larger buttons, use multiples of this size, e.g., 80x160, 160x160, etc.

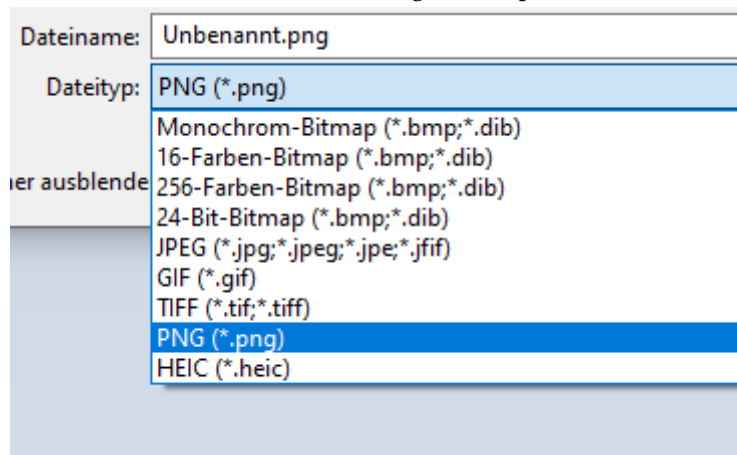
Recommendation:

If the symbol is to be used as a softkey and as a CCI A3 symbol, then make it 64x64 pixels. ... You can also crop the images before scaling them. ...

3.3.4 Reduce BMP Color Depth

For faster loading, reduce the color depth as much as possible.

Windows Paint offers the following color depths: ...



Format	Colors	Bit	Note
Monochrome Bitmap	21 = 2	1	
16-Color Bitmap	24 = 16	4	
256-Color Bitmap	28 = 256	8	
24-Bit Bitmap	224 = 16,777,216	24	not present in ISO 11783-6

3.4 Insert BMP into ISO Designer

The images can now be used in ISO Designer.

In the case of monochrome, white can be set as a transparent background.

In the case of other color depths, pink is often used.

3.4.1 Related topic subpages on ms-muc-docs.de

-  [Eclipse 4diac IDE & Color Reference on ms-muc-docs.de](#)

4. Entries from students

Student area/tree/main/Eintraege

1. Raphael Gross [Data_Mask-Button_erstellen_u_bearbeiten.pdf](#)
2. Markus Winter [Softkey_und_Container_erstellen.pdf](#)

5. Welcome to the ISOBUS-VT-Objects wiki

5.1 Podcast

- [logiBUS®: Revolutionizing agricultural IT – How ISOBUS becomes a smart home for farmers

6. ISOBUS Object IDs

See also:

- [ISOBUS Wiki - Colours](#)]
- [ISOBUS Wiki - Objectpool Objects Database](#)] Version: Jetter ISO-Designer 5.6.1 Here, the object numbers correspond to the IDs in the standard.

(The object numbers can take any value; except for macros.)

(The FmtStr naming scheme [File Name]_%ld for PictureGraphic is likewise fixed by ISO-Designer - switching it to e.g. PictureGraphic_%ld is not accepted by the tool, verified against MyProject.jot)

The definition in this table can therefore be considered good practice.

TypeName	StartID	FmtStr	ID	Link	Wiki
WorkingSet	0	WorkingSet_%ld	0	ID 0 - Working set - ISO-11783-6 - B.1	Wiki
DataMask	1000	DataMask_%ld	1	ID 1 - Data mask - ISO-11783-6 - B.2.	Wiki
AlarmMask	2000	AlarmMask_%ld	2	ID 2 - Alarm Mask - ISO 11783-6 - B.3	Wiki
Container	3000	Container_%ld	3	ID 3 - Container - ISO 11783-6 - B.4	Wiki
SoftKeyMask	4000	SoftKeyMask_%ld	4	ID 4 - Soft key mask - ISO 11783-6 - B.5	Wiki
SoftKey	5000	SoftKey_%ld	5	ID 5 - Key (Soft Key) - ISO 11783-6 - B.6	Wiki
KeyGroup	35000	KeyGroup_%ld	35	ID 35 - Key Group - ISO 11783-6 - B.20	Wiki
Button	6000	Button_%ld	6	ID 6 - Button - ISO 11783-6 - B.7	Wiki
InputBoolean	7000	InputBoolean_%ld	7	ID 7 - Input boolean - ISO 11783-6 - B.8.2	Wiki
InputString	8000	InputString_%ld	8	ID 8 - Input string - ISO 11783-6 - B.8.3	Wiki
InputNumber	9000	InputNumber_%ld	9	ID 9 - Input number - ISO 11783-6 - B.8.4	Wiki
InputList	10000	InputList_%ld	10	ID 10 - Input list - ISO 11783-6 - B.8.5	Wiki
OutputString	11000	OutputString_%ld	11	ID 11 - Output string - ISO 11783-6 - B.9.2	Wiki
OutputNumber	12000	OutputNumber_%ld	12	ID 12 - Output number - ISO 11783-6 - B.9.3	Wiki
OutputList	37000	OutputList_%ld	37	ID 37 - Output List - ISO 11783-6 - B.9.4	Wiki

Line	13000	Line_%ld	13	ID 13 – Output line – ISO 11783-6 – B.10.2	Wiki
Rectangle	14000	Rectangle_%ld	14	ID 14 – Output rectangle – ISO 11783-6 – B.10.3	Wiki
Ellipse	15000	Ellipse_%ld	15	ID 15 – Output ellipse – ISO 11783-6 – B.10.4	Wiki
Polygon	16000	Polygon_%ld	16	ID 16 – Output polygon – ISO 11783-6 – B.10.5	Wiki
Meter	17000	Meter_%ld	17	ID 17 – Output meter – ISO 11783-6 – B.11.2	Wiki
LinearBargraph	18000	LinearBargraph_%ld	18	ID 18 – Output linear bar graph – ISO 11783-6 – B.11.3	Wiki
ArchedBargraph	19000	ArchedBargraph_%ld	19	ID 19 – Output arched bar graph – ISO 11783-6 – B.11.4	Wiki
PictureGraphic	20000	[File Name]_%ld	20	ID 20 – Picture graphic – ISO 11783-6 – B.12.2	Wiki
GraphicsContext	36000	GraphicsContext_%ld	36	ID 36 – Graphics Context Object – ISO 11783-6 – B.18	Wiki
NumberVariable	21000	NumberVariable_%ld	21	ID 21 – Number variable – ISO 11783-6 – B.13.2	Wiki
StringVariable	22000	StringVariable_%ld	22	ID 22 – String variable – ISO 11783-6 – B.13.3	Wiki
FontAttributes	23000	FontAttributes_%ld	23	ID 23 – Font attributes – ISO 11783-6 – B.14.2	Wiki
LineAttributes	24000	LineAttributes_%ld	24	ID 24 – Line attributes – ISO 11783-6 – B.14.3	Wiki
FillAttributes	25000	FillAttributes_%ld	25	ID 25 – Fill attributes – ISO 11783-6 – B.14.4	Wiki
InputAttributes	26000	InputAttributes_%ld	26	ID 26 – Input attributes – ISO 11783-6 – B.14.5	Wiki
ExtendedInputAttributes	38000	ExtendedInputAttributes_%ld	38	ID 38 – Extended Input Attributes – ISO 11783-6 – B.14.6	Wiki
ObjectPointer	27000	ObjectPointer_%ld	27	ID 27 – Object pointer – ISO 11783-6 – B.15	Wiki
Macro	0	Macro_%ld	28	ID 28 – Macro – ISO 11783-6 – B.16	Wiki
AuxFunction1	29000	AuxFunction1_%ld	29	ID 29 – Auxiliary Function Type 1 – ISO 11783-6 – J.4.2	Wiki

AuxInput1	30000	AuxInput1_%ld	30	ID 30 – Auxiliary Input Type 1 – ISO 11783-6 – J.4.4	Wiki
AuxFunction2	31000	AuxFunction2_%ld	31	ID 31 – Auxiliary Function Type 2 – ISO 11783-6 – J.4.3	Wiki
AuxInput2	32000	AuxInput2_%ld	32	ID 32 – Auxiliary Input Type 2 – ISO 11783-6 – J.4.5	Wiki
AuxObjectPointer	33000	AuxObjectPointer_%ld	33	ID 33 - Auxiliary Control Designator Type 2 Object Pointer - ISO 11783-6 - J.4.7	Wiki
ColorMap	39000	ColorMap_%ld	39	ID 39 – Color Map – ISO 11783-6 – B.17	Wiki
WindowMask	34000	WindowMask_%ld	34	ID 34 – Window Mask – ISO 11783-6 – B.19	Wiki
ObjectLabelReferenceList	40000	ObjectLabelReferenceList_%ld	40	ID 40 - Object Label Reference List - ISO 11783-6 - B.21	Wiki
ExternalObjectDefinition	41000	ExternalObjectDefinition_%ld	41	ID 41 – External Object Definition – ISO 11783-6 – B.22	Wiki
ExternalReferenceName	42000	ExternalReferenceName_%ld	42	ID 42 – External reference NAME – ISO 11783-6 – B.23	Wiki
ExternalObjectPointer	43000	ExternalObjectPointer_%ld	43	ID 43 – External Object Pointer – ISO 11783-6 – B.24	Wiki
Animation	44000	Animation_%ld	44	ID 44 - Animation - ISO 11783-6 - B.25	Wiki
ScaledGraphic	48000	ScaledGraphic_%ld	48	ID 48 – Scaled Graphic – ISO 11783-6 – B.28	Wiki
GraphicData	46000	GraphicData_%ld	46	ID 46 – Graphic Data (PNG) – ISO 11783-6 – B.27	Wiki
ColorPalette	45000	ColorPalette_%ld	45	ID 45 – Color Palette – ISO 11783-6 – B.26	Wiki
WorkingSetSpecialControls	47000	WorkingSetSpecialControls_%ld	47	ID 47 – Working Set Special Controls – ISO 11783-6 – B.29	Wiki
IDsForTemporaryUse	64000	IDsForTemporaryUse_%ld	64		
Proxy	4194304	Proxy_%ld			

In versions prior to V. 5.6.0, i.e., up to ISO Designer 5.5.1, the following naming convention was common:
Here, the ID did not match the ObjectID.
For example, with AUX Function.

TypeName	StartID	FmtStr
WorkingSet	0	WorkingSet_%ld
DataMask	100	DataMask_%ld
AlarmMask	200	AlarmMask_%ld
Container	300	Container_%ld
SoftKeyMask	400	SoftKeyMask_%ld
SoftKey	500	SoftKey_%ld
KeyGroup	3600	KeyGroup_%ld
Button	600	Button_%ld
InputBoolean	700	InputBoolean_%ld
InputString	800	InputString_%ld
InputNumber	900	InputNumber_%ld
InputList	1000	InputList_%ld
OutputString	1100	OutputString_%ld
OutputNumber	1200	OutputNumber_%ld
OutputList	3000	OutputList_%ld
Line	1300	Line_%ld
Rectangle	1400	Rectangle_%ld
Ellipse	1500	Ellipse_%ld
Polygon	1600	Polygon_%ld
Meter	1700	Meter_%ld
LinearBargraph	1800	LinearBargraph_%ld
ArchedBargraph	1900	ArchedBargraph_%ld
PictureGraphic	2000	[File Name]_%ld
GraphicsContext	3100	GraphicsContext_%ld
NumberVariable	2100	NumberVariable_%ld
StringVariable	2200	StringVariable_%ld
FontAttributes	2300	FontAttributes_%ld
LineAttributes	2400	LineAttributes_%ld
FillAttributes	2500	FillAttributes_%ld
InputAttributes	2600	InputAttributes_%ld
ExtendedInputAttributes	3200	ExtendedInputAttributes_%ld
ObjectPointer	2700	ObjectPointer_%ld
Macro	0	Macro_%ld
AuxInput2	2800	AuxInput2_%ld
AuxFunction2	2900	AuxFunction2_%ld

AuxObjectPointer	3300	AuxObjectPointer_%ld
ColorMap	3400	ColorMap_%ld
WindowMask	3500	WindowMask_%ld
ObjectLabelReferenceList	3700	ObjectLabelReferenceList_%ld
ExternalObjectDefinition	3800	ExternalObjectDefinition_%ld
ExternalReferenceName	3900	ExternalReferenceName_%ld
ExternalObjectPointer	4000	ExternalObjectPointer_%ld
Animation	4100	Animation_%ld
IDsForTemporaryUse	64000	IDsForTemporaryUse_%ld
Proxy	4194304	Proxy_%ld

7. ISOBUS Object Versions

Which object is present in which version? The table at: https://extranet.epec.fi/Public/Manuals/EPEC_Programming_And_Libraries/projecttopics/topic000962.htm is unfortunately incorrect and outdated: An up-to-date table should be created here, based on ISO 11783-6:2018(en) Tractors and machinery for agriculture and forestry – Serial control and communications data network – Part 6: Virtual terminal

ID	Object description	VT 2	VT 3	VT 4	VT 5	VT 6	Standard	Jetter support
0	Working set	X	X	X	X	X	11783-6 – B.1	L2 - L6
1	Datamask	X	X	X	X	X	11783-6 – B.2	L2 - L6
2	Alarm Mask	X	X	X	X	X	11783-6 – B.3	L2 - L6
3	Containers	X	X	X	X	X	11783-6 – B.4	L2 - L6
4	Soft Key Mask	X	X	X	X	X	11783-6 – B.5	L2 - L6
5	Key	X	X	X	X	X	11783-6 – B.6	L2 - L6
6	Button	X	X	X	X	X	11783-6 – B.7	L2 - L6
7	Input Boolean	X	X	X	X	X	11783-6 – B.8.2	L2 - L6
8	Input String	X	X	X	X	X	11783-6 – B.8.3	L2 - L6
9	Input Number	X	X	X	X	X	11783-6 – B.8.4	L2 - L6
10	Input List	X	X	X	X	X	11783-6 – B.8.5	L2 - L6
11	Output String	X	X	X	X	X	11783-6 – B.9.2	L2 - L6
12	Output Number	X	X	X	X	X	11783-6 – B.9.3	L2 - L6
13	Output Line	X	X	X	X	X	11783-6 – B.10.2	L2 - L6
14	Output Rectangle	X	X	X	X	X	11783-6 – B.10.3	L2 - L6
15	Output Ellipse	X	X	X	X	X	11783-6 – B.10.4	L2 - L6
16	Output Polygon	X	X	X	X	X	11783-6 – B.10.5	L2 - L6
17	Output Meter	X	X	X	X	X	11783-6 – B.11.2	L2 - L6
18	Output Linear Bar Graph	X	X	X	X	X	11783-6 – B.11.3	L2 - L6
19	Output Arched Bar Graph	X	X	X	X	X	11783-6 – B.11.4	L2 - L6
20	Picture Graphic (BMP)	X	X	X	X	X	11783-6 – B.12.1 + B.12.2	L2 - L6
21	Number variable	X	X	X	X	X	11783-6 – B.13.2	L2 - L6
22	String variable	X	X	X	X	X	11783-6 – B.13.3	L2 - L6
23	Font Attributes	X	X	X	X	X	11783-6 – B.14.2	L2 - L6
24	Line Attributes	X	X	X	X	X	11783-6 – B.14.3	L2 - L6
25	Fill Attributes	X	X	X	X	X	11783-6 – B.14.4	L2 - L6

26	Input Attributes	X	X	X	X	X	11783-6 – B.14.5	L2 - L6
27	Object Pointer	X	X	X	X	X	11783-6 – B.15	L2 - L6
28	Macro	X	X	X	X	X	11783-6 – B.16	L2 - L6
29	Auxiliary Function Type 1	X					11783-6 - J.4.2	L6
30	Auxiliary Input Type 1	X					11783-6 - J.4.4	L6
31	Auxiliary Function Type 2		X	X	X	X	11783-6 – J.4.3	L3 - L6
32	Auxiliary Input Type 2		X	X	X	X	11783-6 – J.4.5	L3 - L6
33	Auxiliary Control DesignatorType 2 Object Pointer		X	X	X	X	11783-6 – J.4.7	L3 - L6
34	Windows Mask			X	X	X	11783-6 – B.19	L4 - L6
35	Key Group			X	X	X	11783-6 – B.20	L4 - L6
36	Graphics Context Object			X	X	X	11783-6 – B.18	L3 - L6
37	Output List			X	X	X	11783-6 – B.9.4	L4 - L6
38	Extended Input Attributes			X	X	X	11783-6 – B.14.6	L4 - L6
39	Color Map			X	X	X	11783-6 – B.17	L5 - L6
40	Object Label Reference List			X	X	X	11783-6 – B.21	L5 - L6
41	External Object Definition				X	X	11783-6 – B.22	L5 - L6
42	External reference NAME				X	X	11783-6 – B.23	L5 - L6
43	External Object Pointer				X	X	11783-6 – B.24	L5 - L6
44	Animation				X	X	11783-6 – B.25	L5 - L6
45	Color Palette				X	X	11783-6 – B.26	L6
46	Graphic Data (PNG)					X	11783-6 – B.27	L6
47	Working Set Special Controls					X	11783-6 - B.29	L6
48	Scaled Graphic					X	11783-6 - B.28	L6

8. ISOBUS Objects

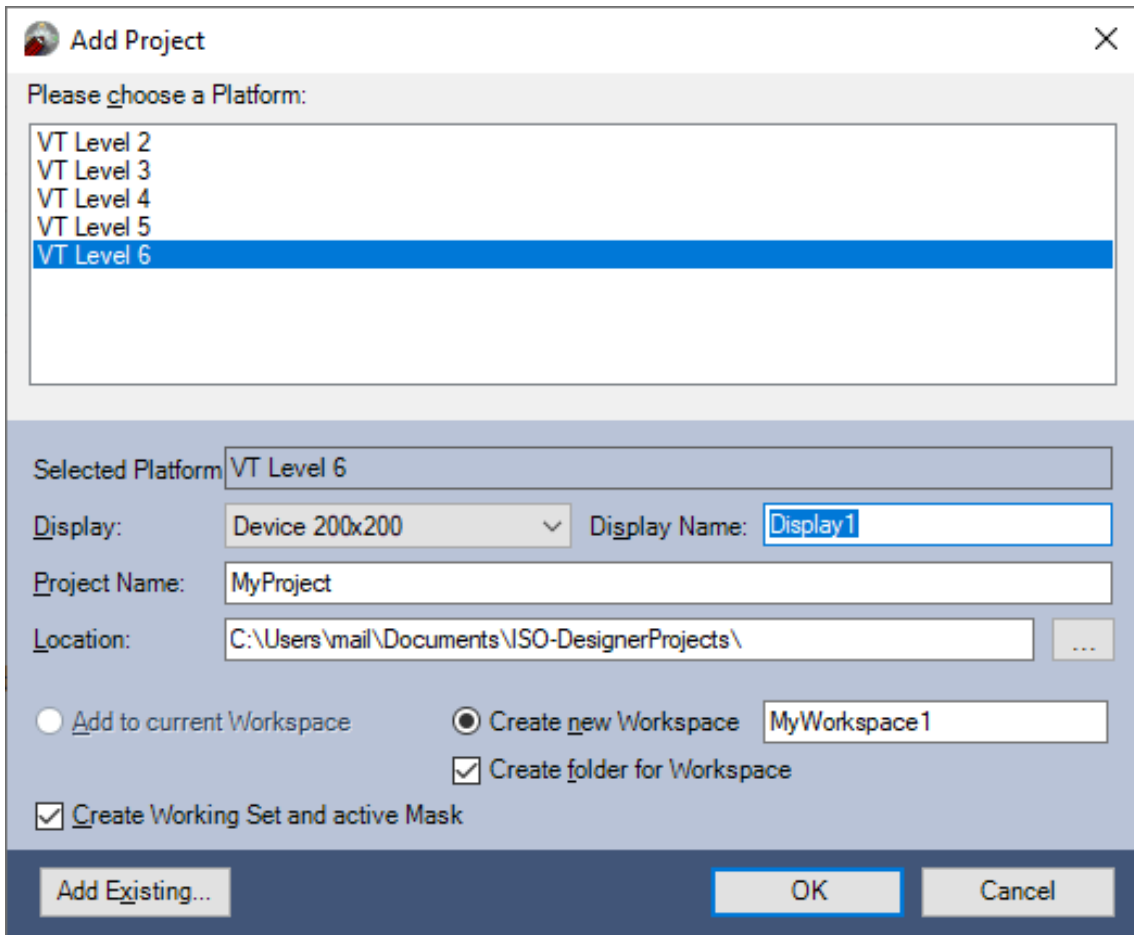
Take a close look at this link.

8.1 Programming

[Programming_And_Libraries](#)] [Future: ISOBUS Objects Versions](#)] [ISOBUS Wiki](#) by Tobias Tenberg:

- <https://isobus-studio.com/en/isobus-wiki>
- [ISOBUS Wiki - Colors](#)]
- [ISOBUS Wiki - Objectpool Objects Database](#)]

As you can see, there are various objects, but not all are supported in every ISOBUS version.



If you were to create a new project with ISO Designer, it would ask you for the “VT Level.”

If you search for devices in the AEF database, you will see something like this:

Options	Description
UT 2.0 CLIENT	
Field	Value
Functionality	UT
Functionality Type	Client
Generation	2.0

So what is the difference between UT and VT?

Or are they the same thing?

Is VT 2.0 the same as UT 2.0?

Here's the answer:

8.2 VT Levels

	ISO 11783-06	Jetter ISO Designer	UT according to AEF data-base	AUX according to AEF data-base
1	Unnamed	< nicht wählbar >	< nicht zertifizierbar>	no AUX
2	VT Version 2	VT Level 2	UT 1.0	AUX-N 1.0
3	VT Version 3	VT Level 3	UT 2.0	AUX-N 1.0
4	VT Version 4	VT Level 4	UT 2.0	AUX-N 1.0
5	VT Version 5	VT Level 5	UT 2.0	AUX-N 1.0
6	VT Version 6	VT Level 6	UT 2.0	AUX-N 1.0
7	VT Version 7	< nicht wählbar >	UT 3.0	AUX-N 1.0

So, they started with the first version of the standard and published it.

In the second version, VT version 2 was mentioned... and so on. Please refer to the standard for details.

The AEF does not adhere to this scheme but introduces its own versions for marketing purposes. UT 3.0 is currently under development and will likely include VT 4 and 5.

AUX is only mentioned in the ISOBUS standard in relation to the respective VT standard; the AEF has created AUX-O and AUX-N from this.

9. ISOBUS Wiki Links

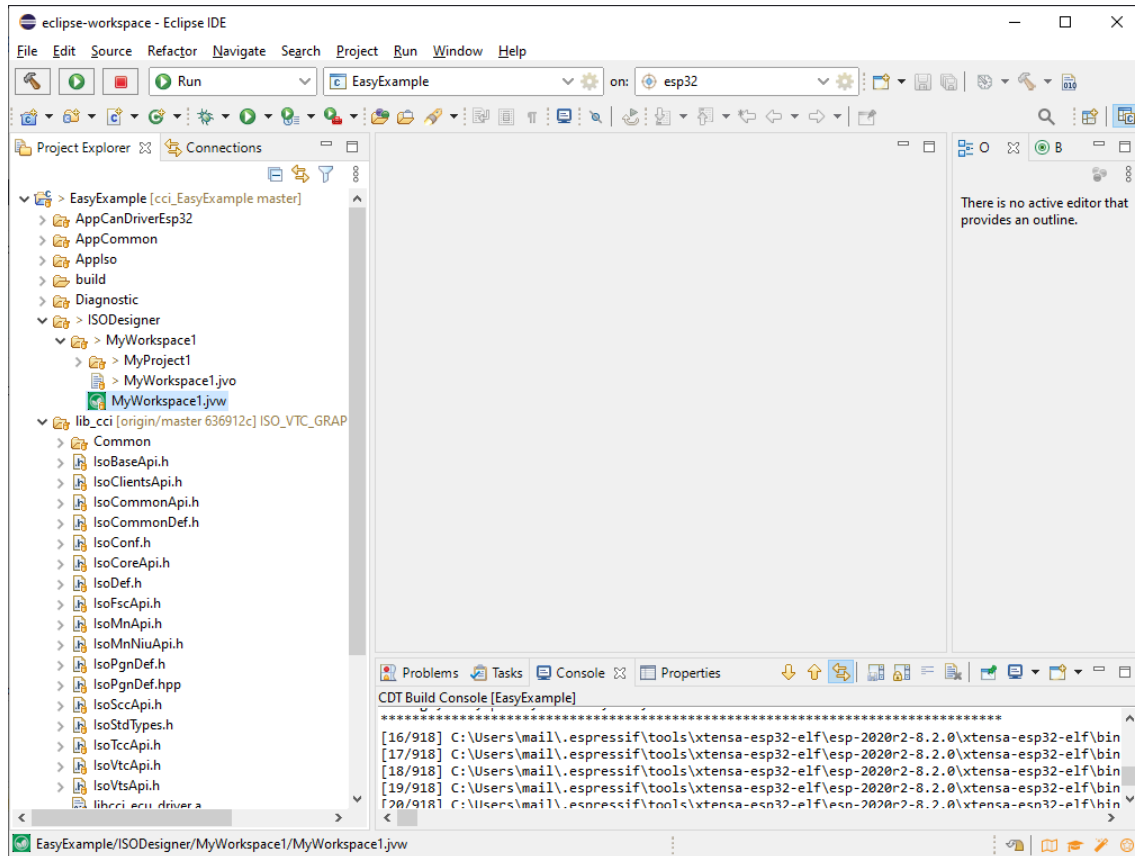
Here is a collection of links to the ISOBUS Wiki by Tobias Tenberg, providing detailed information on various ISOBUS objects. ISOBUS Wiki links ...```]

]] * [Alert Mask](#)

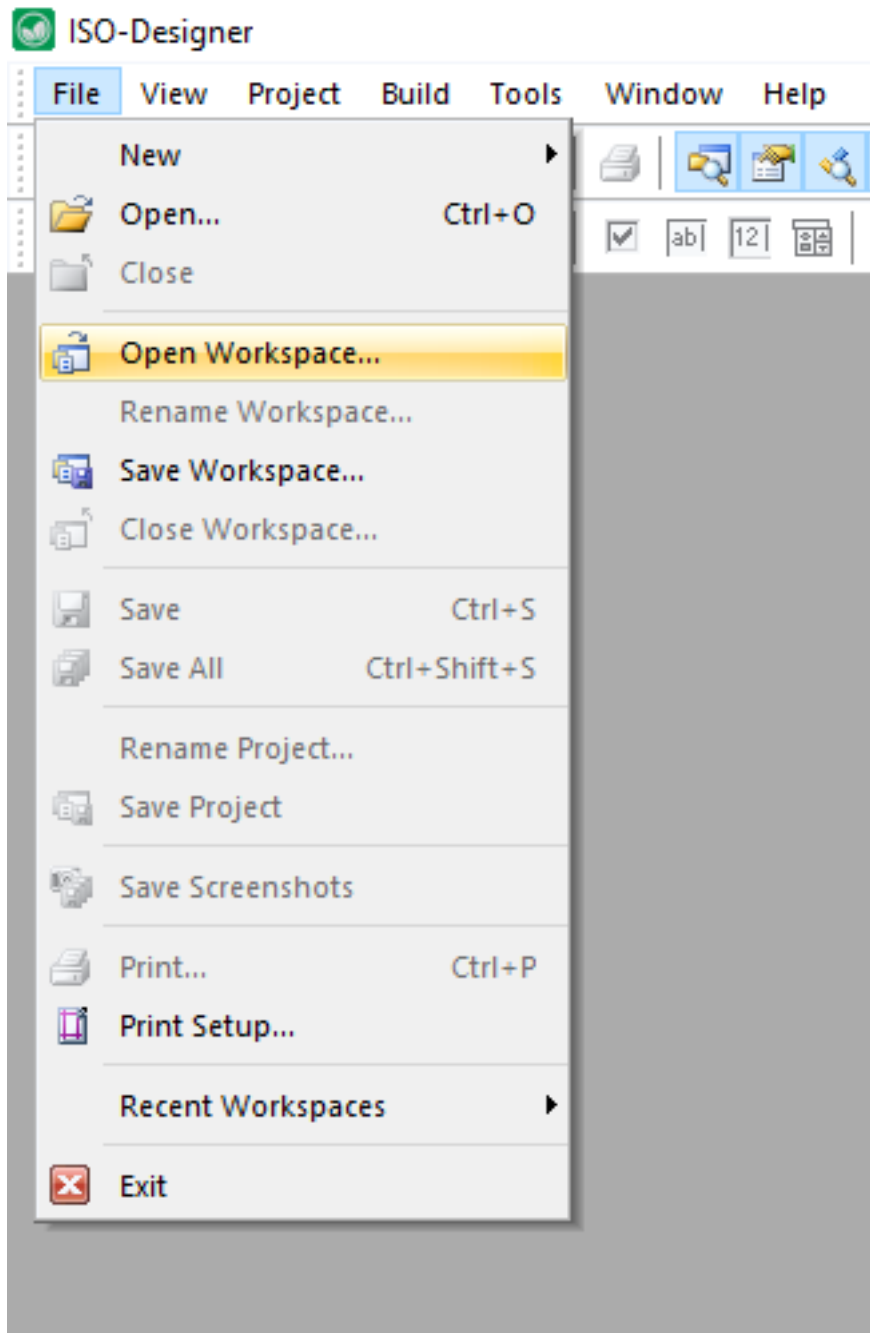
- [Arched Bar Graph](#)
- [Boolean](#)
- [Button](#)
- [Container](#)
- [Data Mask](#)
- [Ellipse](#)
- [Fill Attribute](#)
- [Font Attribute](#)
- [Line \(Output\)](#)
- [Line Attribute](#)
- [List \(Input\)](#)
- [List \(Output\)](#)
- [Meter](#)
- [Number \(Input\)](#)
- [Number \(Output\)](#)
- [Number Variable](#)
- [Object Pointer](#)
- [Picture Graphic object](#)
- [Polygon](#)
- [Rectangle](#)
- [Softkey](#)
- [Softkey Mask](#)
- [String \(Input\)](#)
- [String \(Output\)](#)
- [String Variable](#)
- [Window Mask](#)
- [Working Set](#)

10. Open Jetter ISO Designer Workspace

Double-click here:

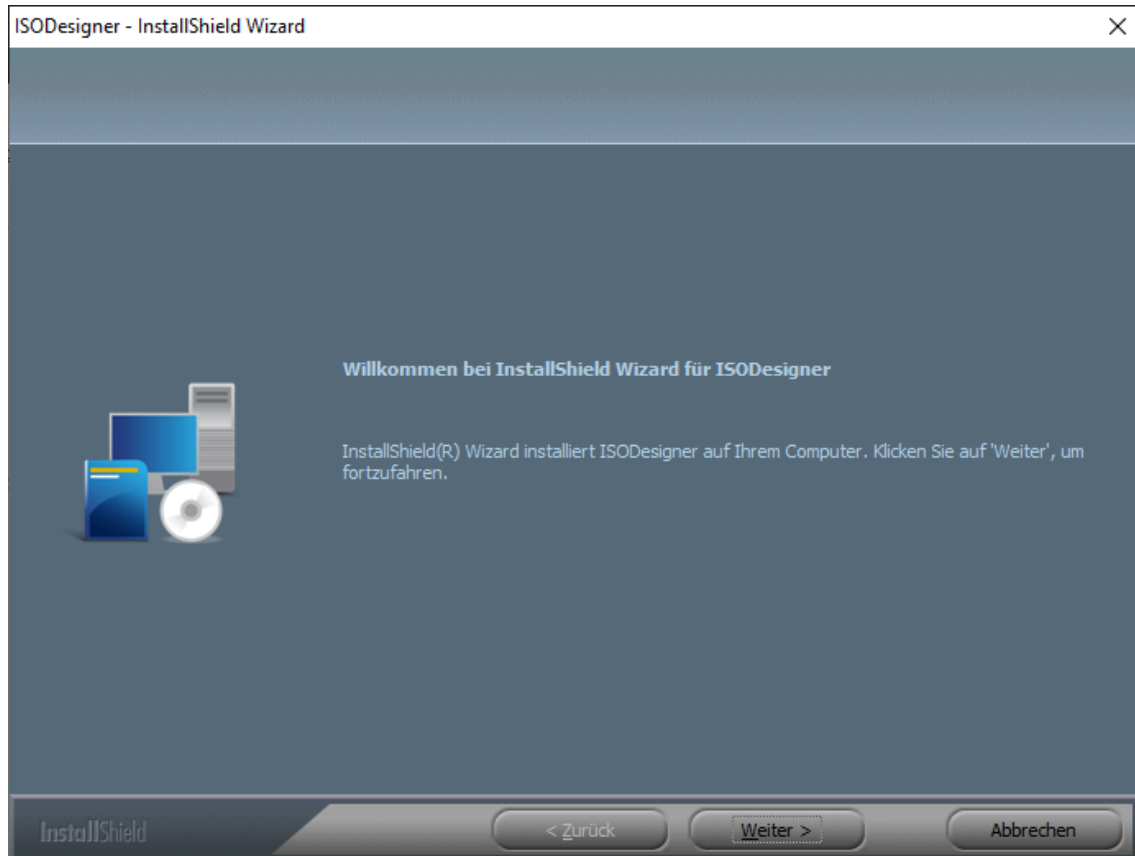


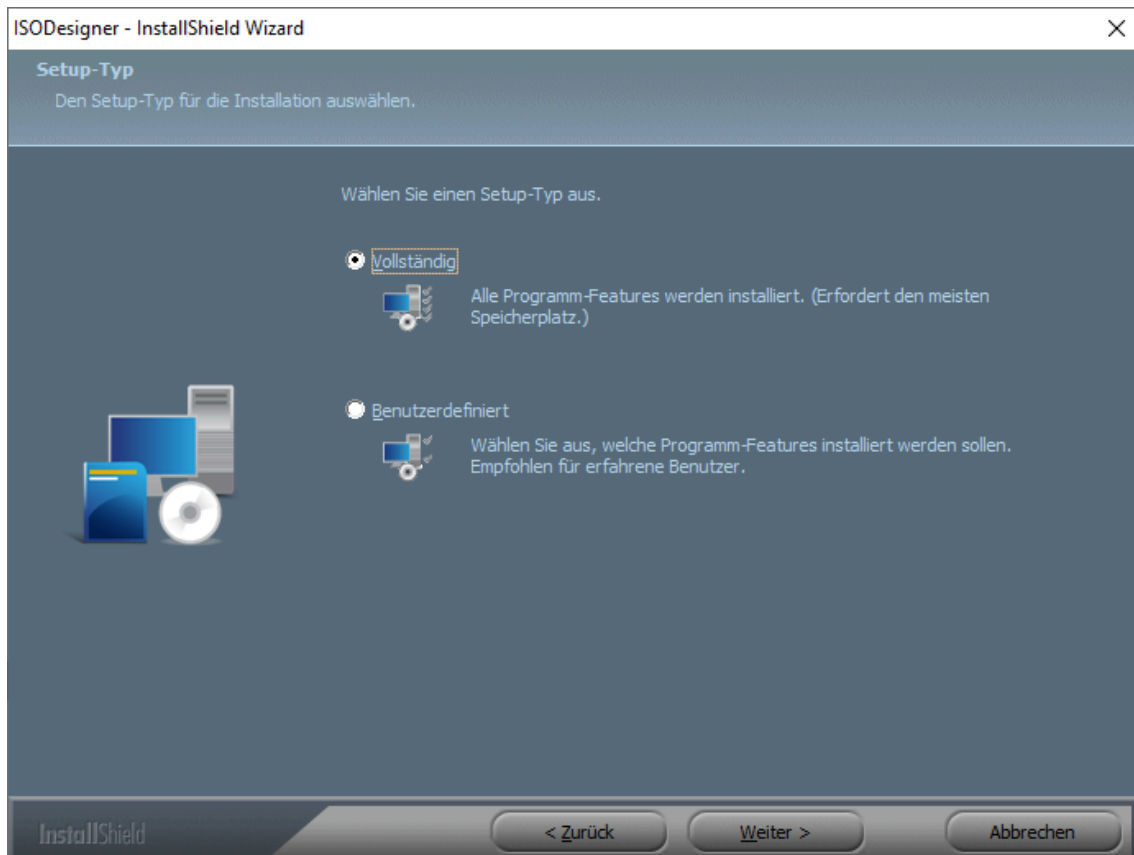
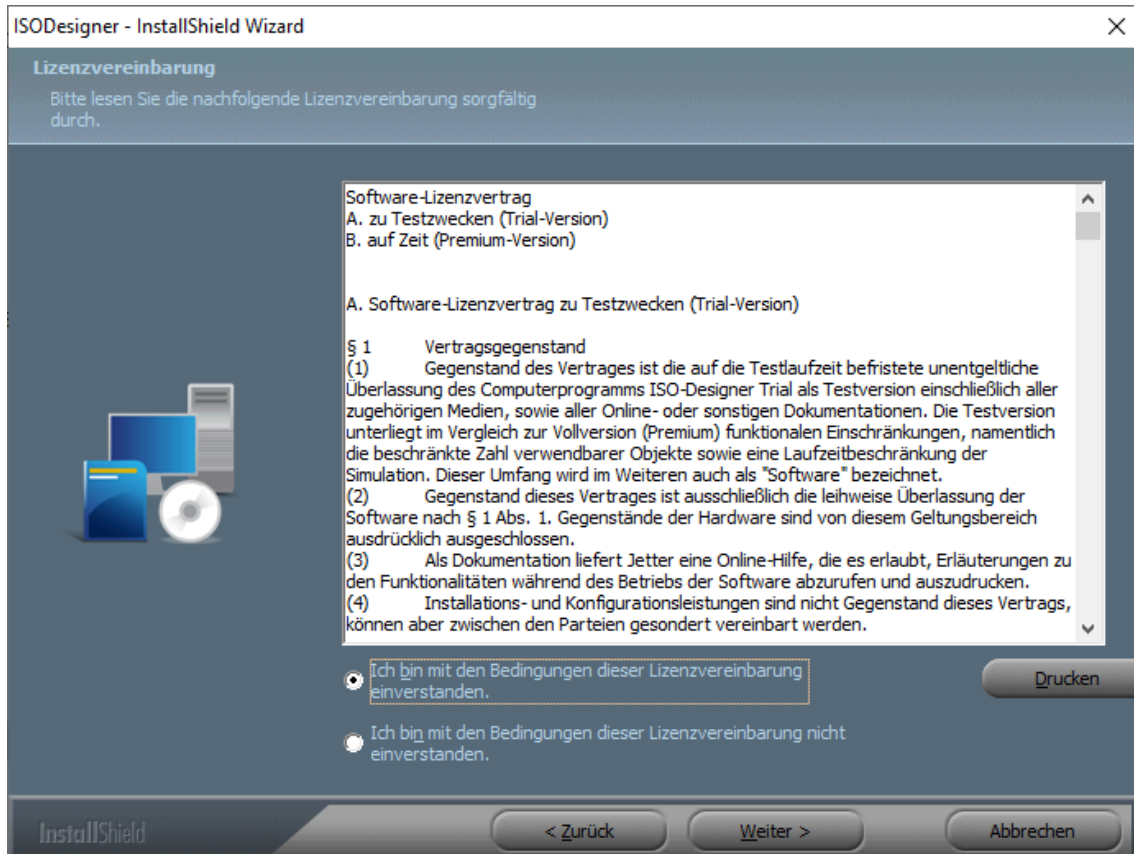
Alternatively:

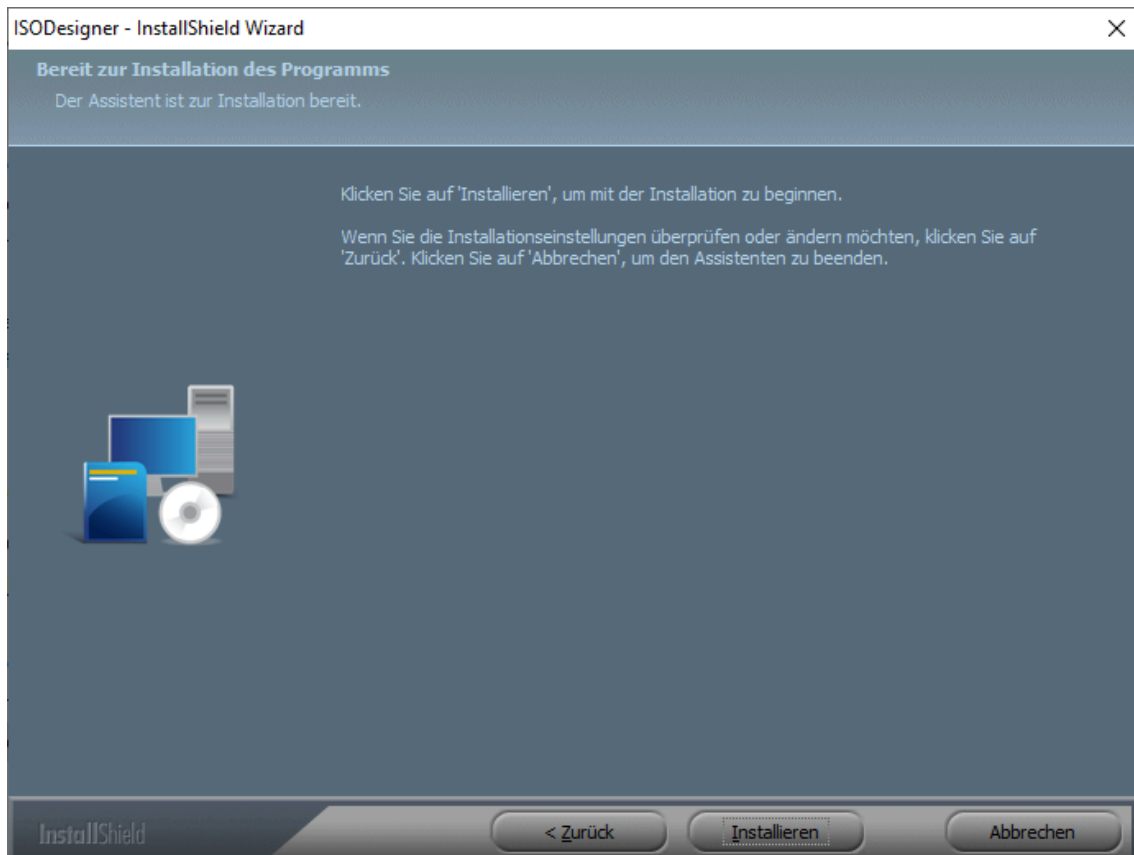
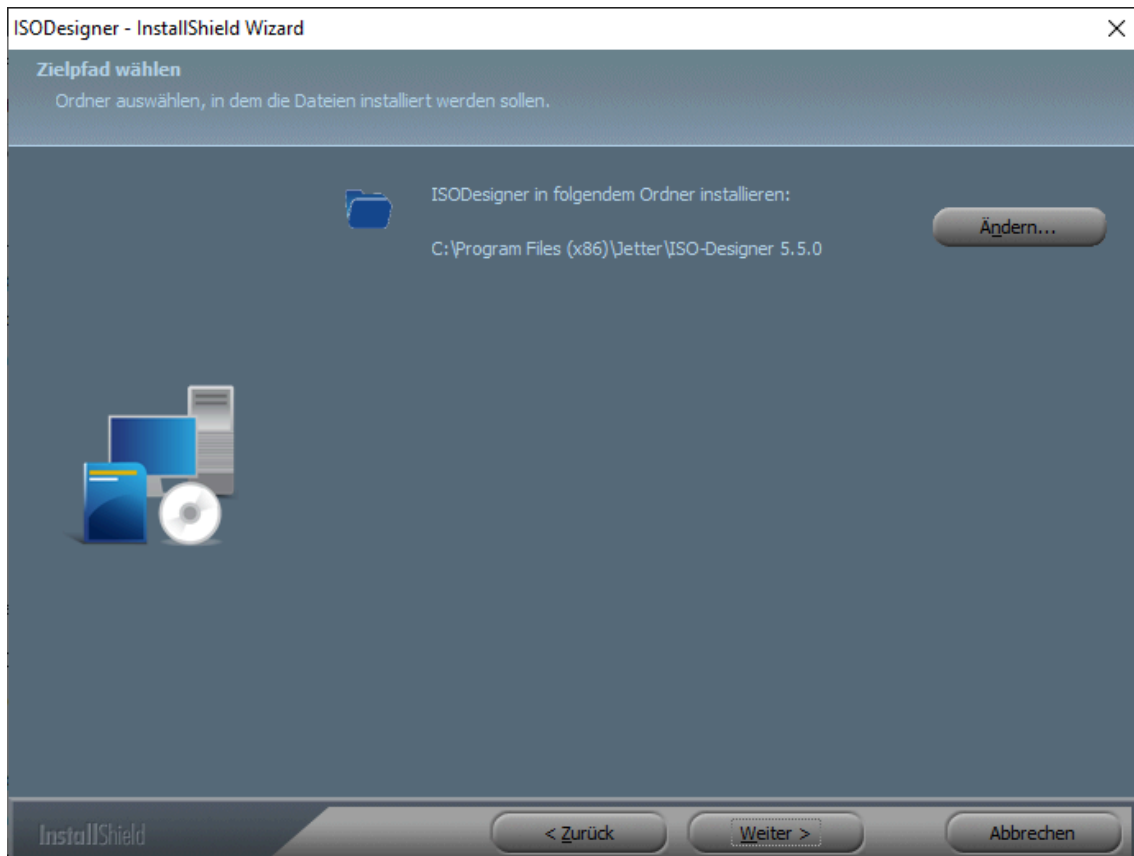


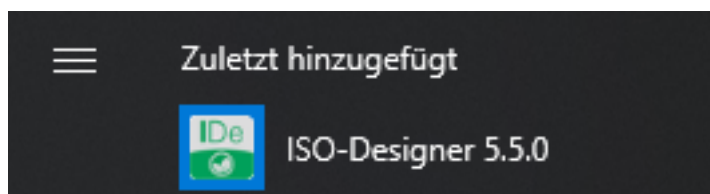
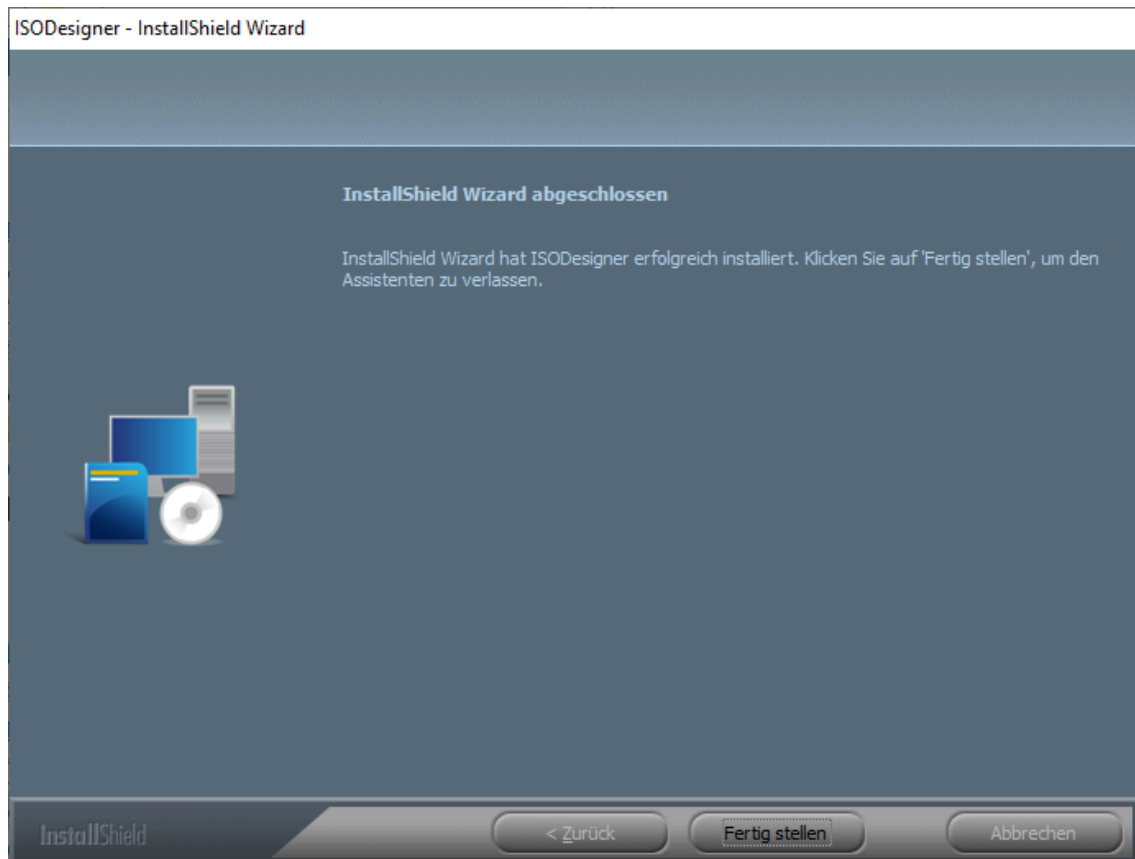
11. Jetter ISO-Designer

M:\Agricultural Machinery Mechanics\Instruction\SL\Jetter\ISODesignerSetup551.exe

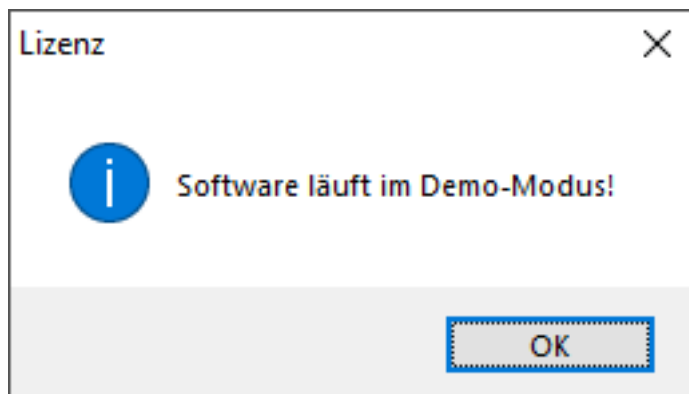








Yes



OK

12. Empty Objects (Minimum Child Objects)

The ISO 11783-6 standard makes an important distinction for container objects (objects that contain other objects): **Can the object be empty or not?** This rule is defined by the permissible range of values for the attribute “Number of objects to follow”.

12.1 Objects that MUST NOT be empty (Minimum 1 Child)

These objects must contain at least one child object, as they cannot fulfill their function in the system without content (usually because they must be visible to the user as a **designator** / icon).

Object ID	Name	Reason	Standard Reference
0	Working Set	Must be displayed for selection in the VT menu. Without a graphical designator, the device would be invisible to the operator and could not be selected.	Table B.2
29	Auxiliary Function Type 1	Must be displayed on the mapping screen so the user can assign the function (e.g., “Lift”) to a switch.	Table J.1
30	Auxiliary Input Type 1	Must be displayed on the mapping screen to represent the physical input (e.g., “Joystick Button A”).	Table J.3
31	Auxiliary Function Type 2	See ID 29.	Table J.2
32	Auxiliary Input Type 2	See ID 30.	Table J.4

Error Case: If such an object is defined with 0 child objects in the object pool, the pool is **invalid** and will be rejected by the VT (or a validation tool).

** _

12.2 Objects that may be empty (Minimum 0 children)

For most other containers, content such as 0 is permitted. This can be quite useful.

Object ID	Name	Behavior with 0 children
1	Data Mask	Displays only the background color.
2	Alarm Mask	Displays only the background color (warning without content).
3	Container	Is effectively invisible and takes up no space (unless an area with a background color is defined by Width/Height).
4	Soft Key Mask	Hides all soft keys or leaves them unlabeled.
5	Key	Displays an empty key in the defined background color.
6	Button	Displays an empty button in the defined background color.
35	Key Group	Technically allowed, but functionally pointless (an empty group).
44	Animation	An animation without frames displays nothing.

12.2.1 Special Feature: Input List (ID 10)

Even a `Input List` can theoretically be empty (`Number of list items = 0`). In this case, the list exists but contains no selectable options.

35 | **Key Group** | Technically, an animation without frames displays nothing.

13. Object Hierarchy (Parent-Child Relationships)

This page is based on **Table A.2 — Allowed hierarchical relationships of objects** of ISO 11783-6:2018 (Annex A). It defines which objects (child) may be contained within which other objects (parent). The specified number represents the minimum **VT version** required for this relationship to be supported.

13.1 Parent: Working Set object (ID 0)

The working set object is the root container of an object pool hierarchy.

Child Object	Min. VT Version
Container object (ID 3)	4
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output List object (ID 37)	4
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4
Graphics Context object (ID 36)	4
Picture Graphic object (ID 20)	2
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	4

13.2 Parent: Data Mask object (ID 1)

The data mask is the main display area for operation.

Child Object	Min. VT version
Working Set object (ID 0)	3
Container object (ID 3)	2
Button object (ID 6)	2
Input Boolean object (ID 7)	2
Input String object (ID 8)	2
Input Number object (ID 9)	2

Input List object (ID 10)	2
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output List object (ID 37)	4
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2
Output Meter object (ID 17)	2
Output Linear Bar Graph object (ID 18)	2
Output Arched Bar Graph object (ID 19)	2
Graphics Context object (ID 36)	4
Animation object (ID 44)	5
Picture Graphic object (ID 20)	2
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	2
External Object Pointer object (ID 43)	5
Auxiliary Function Type 2 object (ID 31)	3
Auxiliary Input Type 2 object (ID 32)	3
Auxiliary Control Designator Type 2 Object Pointer (ID 33)	3

13.3 Parent: Alarm Mask object (ID 2)

Alarm masks are used to display warnings.

Child Object	Min. VT version
Working Set object (ID 0)	3
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output List object (ID 37)	4
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2
Output Meter object (ID 17)	2
Output Linear Bar Graph object (ID 18)	2
Output Arched Bar Graph object (ID 19)	2
Graphics Context object (ID 36)	4
Animation object (ID 44)	5

Picture Graphic object (ID 20)	2
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	2
External Object Pointer object (ID 43)	5
Auxiliary Function Type 2 object (ID 31)	3
Auxiliary Input Type 2 object (ID 32)	3
Auxiliary Control Designator Type 2 Object Pointer (ID 33)	3

13.4 Parent: Container object (ID 3)

Containers are used to group objects.

Child Object	Min. VT version
Working Set object (ID 0)	3
Container object (ID 3)	2
Button object (ID 6)	2
Input Boolean object (ID 7)	2
Input String object (ID 8)	2
Input Number object (ID 9)	2
Input List object (ID 10)	2
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output List object (ID 37)	4
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2
Output Meter object (ID 17)	2
Output Linear Bar Graph object (ID 18)	2
Output Arched Bar Graph object (ID 19)	2
Graphics Context object (ID 36)	4
Animation object (ID 44)	5
Picture Graphic object (ID 20)	2
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	2
External Object Pointer object (ID 43)	5
Auxiliary Function Type 2 object (ID 31)	3
Auxiliary Input Type 2 object (ID 32)	3
Auxiliary Control Designator Type 2 Object Pointer (ID 33)	3

13.5 Parent: Window Mask object (ID 34)

Window masks are used in user layouts.

Child Object	Min. VT version
Working Set object (ID 0)	4
Container object (ID 3)	4
Button object (ID 6)	4
Input Boolean object (ID 7)	4
Input String object (ID 8)	4
Input Number object (ID 9)	4
Input List object (ID 10)	4
Output String object (ID 11)	4
Output Number object (ID 12)	4
Output List object (ID 37)	4
Output Line object (ID 13)	4
Output Rectangle object (ID 14)	4
Output Ellipse object (ID 15)	4
Output Polygon object (ID 16)	4
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4
Graphics Context object (ID 36)	4
Animation object (ID 44)	5
Picture Graphic object (ID 20)	4
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	4
External Object Pointer object (ID 43)	5
Auxiliary Function Type 2 object (ID 31)	6
Auxiliary Input Type 2 object (ID 32)	6
Auxiliary Control Designator Type 2 Object Pointer (ID 33)	4

13.6 Parent: Soft Key Mask object (ID 4)

Defines the assignment of the softkeys.

Child Object	Min. VT version
Key object (ID 5)	2
Object Pointer object (ID 27)	2
External Object Pointer object (ID 43)	5

13.7 Parent: Key object (ID 5)

Content of a key.

Child Object	Min. VT version
Working Set object (ID 0)	4
Container object (ID 3)	2
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output List object (ID 37)	4
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4
Graphics Context object (ID 36)	4
Animation object (ID 44)	5
Picture Graphic object (ID 20)	2
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	2
External Object Pointer object (ID 43)	5

13.8 Parent: Button object (ID 6)

Content of a button.

Child Object	Min. VT version
Working Set object (ID 0)	4
Container object (ID 3)	2
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output List object (ID 37)	4
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4

Graphics Context object (ID 36)	4
Animation object (ID 44)	5
Picture Graphic object (ID 20)	2
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	2
External Object Pointer object (ID 43)	5

13.9 Parent: Key Group object (ID 35)

Grouping of softkeys.

Child Object	Min. VT version
Key object (ID 5)	4
Object Pointer object (ID 27)	4
External Object Pointer object (ID 43)	5

13.10 Parent: Input List object (ID 10)

Options in a selection list.

Child Object	Min. VT version
Working Set object (ID 0)	4
Container object (ID 3)	4
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output List object (ID 37)	4
Output Line object (ID 13)	4
Output Rectangle object (ID 14)	4
Output Ellipse object (ID 15)	4
Output Polygon object (ID 16)	4
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4
Graphics Context object (ID 36)	4
Picture Graphic object (ID 20)	2
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	4
External Object Pointer object (ID 43)	5

13.11 Parent: Output List object (ID 37)

Options in an output list.

Child Object	Min. VT version
Working Set object (ID 0)	4
Container object (ID 3)	4
Button object (ID 6)	4
Input Boolean object (ID 7)	4
Input String object (ID 8)	4
Input Number object (ID 9)	4
Input List object (ID 10)	4
Output String object (ID 11)	4
Output Number object (ID 12)	4
Output List object (ID 37)	4
Output Line object (ID 13)	4
Output Rectangle object (ID 14)	4
Output Ellipse object (ID 15)	4
Output Polygon object (ID 16)	4
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4
Graphics Context object (ID 36)	4
Animation object (ID 44)	5
Picture Graphic object (ID 20)	4
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	4
External Object Pointer object (ID 43)	5
Auxiliary Function Type 2 object (ID 31)	6
Auxiliary Input Type 2 object (ID 32)	6
Auxiliary Control Designator Type 2 Object Pointer (ID 33)	4

13.12 Parent: Auxiliary Function Type 1 object (ID 29)

Auxiliary function designator (deprecated).

Child Object	Min. VT version
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2

Picture Graphic object (ID 20)	2
--------------------------------	---

13.13 Parent: Auxiliary Input Type 1 object (ID 30)

Auxiliary input designator (deprecated).

Child Object	Min. VT version
Output String object (ID 11)	2
Output Number object (ID 12)	2
Output Line object (ID 13)	2
Output Rectangle object (ID 14)	2
Output Ellipse object (ID 15)	2
Output Polygon object (ID 16)	2
Picture Graphic object (ID 20)	2

13.14 Parent: Auxiliary Function Type 2 object (ID 31)

Auxiliary function designator.

Child Object	Min. VT version
Container object (ID 3)	3
Output String object (ID 11)	3
Output Number object (ID 12)	3
Output List object (ID 37)	4
Output Line object (ID 13)	3
Output Rectangle object (ID 14)	3
Output Ellipse object (ID 15)	3
Output Polygon object (ID 16)	3
Output Meter object (ID 17)	3
Output Linear Bar Graph object (ID 18)	3
Output Arched Bar Graph object (ID 19)	3
Graphics Context object (ID 36)	4
Picture Graphic object (ID 20)	3
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	3

13.15 Parent: Auxiliary Input Type 2 object (ID 32)

Designator for auxiliary inputs.

Child Object	Min. VT version
Container object (ID 3)	3
Output String object (ID 11)	3

Output Number object (ID 12)	3
Output List object (ID 37)	4
Output Line object (ID 13)	3
Output Rectangle object (ID 14)	3
Output Ellipse object (ID 15)	3
Output Polygon object (ID 16)	3
Output Meter object (ID 17)	3
Output Linear Bar Graph object (ID 18)	3
Output Arched Bar Graph object (ID 19)	3
Graphics Context object (ID 36)	4
Picture Graphic object (ID 20)	3
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	3

13.16 Parent: Object Label graphic representation

Objects that may be used as icons/graphics in a label.

Child Object	Min. VT version
Container object (ID 3)	4
Output String object (ID 11)	4
Output Number object (ID 12)	4
Output List object (ID 37)	4
Output Line object (ID 13)	4
Output Rectangle object (ID 14)	4
Output Ellipse object (ID 15)	4
Output Polygon object (ID 16)	4
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4
Graphics Context object (ID 36)	4
Picture Graphic object (ID 20)	4
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	4

13.17 Parent: Object Label Reference List object (ID 40)

Defines assignments of labels to objects.

Child Object	Min. VT version
Working Set object (ID 0)	4

Data Mask object (ID 1)	4
Alarm Mask object (ID 2)	4
Container object (ID 3)	4
Window Mask object (ID 34)	4
Soft Key Mask object (ID 4)	4
Key object (ID 5)	4
Button object (ID 6)	4
Input Boolean object (ID 7)	4
Input String object (ID 8)	4
Input Number object (ID 9)	4
Input List object (ID 10)	4
Output String object (ID 11)	4
Output Number object (ID 12)	4
Output List object (ID 37)	4
Output Line object (ID 13)	4
Output Rectangle object (ID 14)	4
Output Ellipse object (ID 15)	4
Output Polygon object (ID 16)	4
Output Meter object (ID 17)	4
Output Linear Bar Graph object (ID 18)	4
Output Arched Bar Graph object (ID 19)	4
Graphics Context object (ID 36)	4
Animation object (ID 44)	5
Picture Graphic object (ID 20)	4
Scaled Graphic object (ID 48)	6
Number Variable object (ID 21)	5
String Variable object (ID 22)	5
Font Attributes object (ID 23)	5
Line Attributes object (ID 24)	5
Fill Attributes object (ID 25)	5
Input Attributes object (ID 26)	5
Extended Input Attributes object (ID 41)	5
Color Map object (ID 42)	5
Object Label Reference List object (ID 40)	5
Object Pointer object (ID 27)	5
External Object Definition object (ID 45)	5

External Reference NAME object (ID 46)	5
External Object Pointer object (ID 43)	5
Macro object (ID 47)	5
Auxiliary Function Type 1 object (ID 29)	5
Auxiliary Input Type 1 object (ID 30)	5
Auxiliary Function Type 2 object (ID 31)	4
Auxiliary Input Type 2 object (ID 32)	4
Auxiliary Control Designator Type 2 Object Pointer (ID 33)	5
Color Palette object (ID 49)	6
Graphic Data object (ID 50)	6
Working Set Special Controls object (ID 51)	6

13.18 Parent: Animation object (ID 44)

Sequence of objects.

Child Object	Min. VT version
Container object (ID 3)	5
Output String object (ID 11)	5
Output Number object (ID 12)	5
Output List object (ID 37)	5
Output Line object (ID 13)	5
Output Rectangle object (ID 14)	5
Output Ellipse object (ID 15)	5
Output Polygon object (ID 16)	5
Output Meter object (ID 17)	5
Output Linear Bar Graph object (ID 18)	5
Output Arched Bar Graph object (ID 19)	5
Graphics Context object (ID 36)	5
Animation object (ID 44)	5
Picture Graphic object (ID 20)	5
Scaled Graphic object (ID 48)	6
Object Pointer object (ID 27)	5

14. Invalid Object Pools

For an ISOBUS object pool (IOP) to be accepted and loaded by the Virtual Terminal (VT), it must adhere to strict structural rules. If the pool violates any of these rules, the VT sends an error code at the end of the transmission (End of Object Pool) and usually discards the entire pool. Here are the most important rules that every programmer needs to know:

14.1 1. Structural Integrity (Parsing Errors)

These are the most common errors that cause the VT parser to malfunction.

- **Incorrect Length Specifications:** If an object specifies “Five child objects follow,” but the byte stream only contains four (or six), the VT incorrectly reads the next object byte as an attribute of the previous one. The pool is then corrupted.
- **Unknown Object Types:** An object type ID that is not defined in the standard (and the supported VT version).
- **Missing ID 0 Object:** Each workset **must** contain exactly one object with the **ID 0** (Working Set). If this is missing, the VT does not know who owns the pool.

14.2 2. Referencing Errors (Dangling Pointers)

The VT checks whether all links are valid when loading (or at the latest when activating).

- **Dead References:** An object (e.g., a container) references an object ID (e.g., 100) in its child list, but there is no object with ID 100 in the entire pool.
- *Exception:* References to NULL (65535 or FFFFh) are allowed and expected (e.g., empty softkeys).
- **Incorrect Object Types:** An attribute requires a specific type, but the referenced ID belongs to a different type.
- *Example:* A Input Number object references a String Variable object at Variable Reference. -> **Invalid!**
- *Example:* A Output Rectangle references a Fill Attributes object at Line Attributes. -> **Invalid!**

14.3 3. Logical Errors & Recursion

- **Circular References:** Container A contains Container B, and Container B contains Container A. This leads to an infinite loop during drawing. The VT detects this and rejects the pool or (in the worst case) crashes.
- **Circular References:** Container A contains Container B, and Container B contains Container A. *
- **Multiple Definitions:** An object ID (e.g., ID 50) may only be defined once in the pool. Duplicate IDs are prohibited.

14.4 4. Specific Object Rules (Hard Constraints)

Some objects have very specific rules that are often overlooked:

- **Designator Requirement:** Objects such as Working Set, Auxiliary Function, and Auxiliary Input **must** have at least one child object (see page [Empty Objects](#)).
- **Input Lists:** The Value of an Input List may not be greater than Anzahl der Listeneinträge - 1.
- **Soft Key Masks:** May only contain Key objects (ID 5) or Object Pointer (ID 27), which in turn point to keys. A Button (ID 6) within a Soft Key Mask is invalid.
- **Variable Size:** When a String object references a String variable, the defined length in the String object should ideally match the length of the variable (although VTs are often tolerant here and will clip/pad).

14.5 5. Resource Limits

Even if the pool is syntactically correct, it may be rejected if it exceeds the hardware limits of the VT:

- **Memory Full:** The pool is larger than the available flash memory of the VT (Memory Out of Range).
- **Too many objects:** Some older VTs have limits on the absolute number of objects (e.g., max. 65534, which is determined by the 16-bit ID field, but actual limits are often lower).
- **Softkey limit:** A softkey mask that defines more keys than the VT can manage (although paging is required, there are often hard limits).

14.6 Troubleshooting tips

If the VT reports “Object Pool Error” during loading:

1. Check if **ID 0** exists.
2. Check if all referenced IDs are actually in the pool.
3. Check if all **Number of objects/macros/bytes** fields exactly match the following data.
4. Check if type references (**Number Var** vs. **String Var**) are correct.

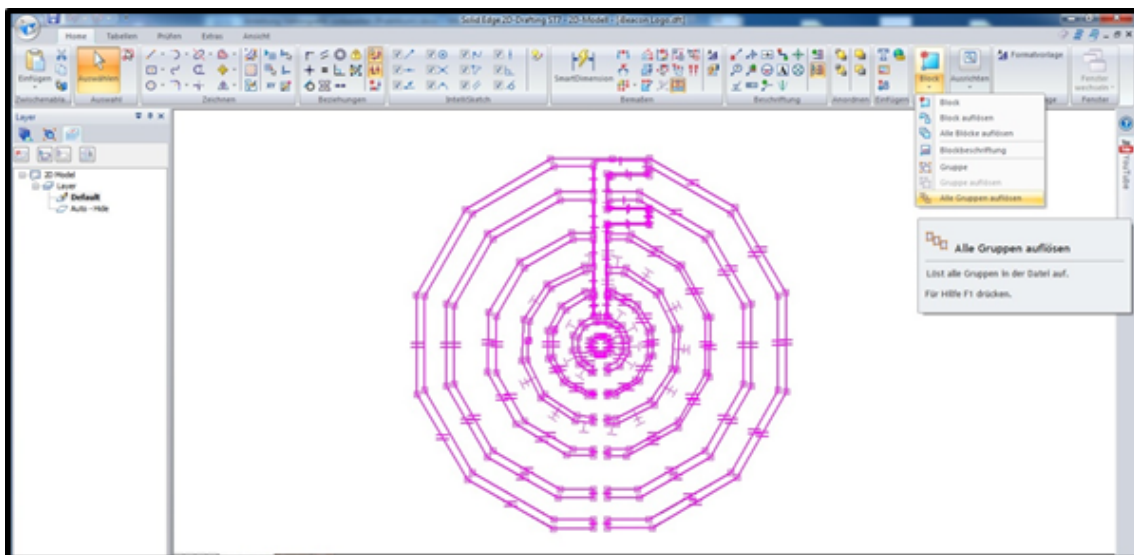
15. Creating Vector Graphics

15.1 CorelDRAW X5

If you need to extract the file from, for example, a PDF file, you can use CorelDRAW X5. The size can also be changed via Arrange → Modify → Resize. Caution: This is very imprecise and will require a few attempts to achieve the desired size. If there's another way, please correct it here (smile)

15.2 Solid Edge

- If something doesn't work, it helps to simply redo each line,
- to ensure clear connections between the lines. Time-consuming, but it should work. (wink)
- When drawing symbols for the displays, they must be 72x72. Quickly draw a rectangle and check it; that way you're on the safe side.
- When drawing symbols for the displays, they must have dimensions of 72x72. * Select the entire object → Block → Ungroup all



Save the finished, scaled vector file

Then Save As → Save as Translated (in .dxf format)

Attention! The object must not contain any curves. Everything must be a “Polygon over Center Point”. Ideally, nine sides.

Continuing to work with curves would otherwise cause problems in LibreCAD or ISO Designer!

15.3 LibreCAD

Open the saved .dxf file in LibreCAD

All lines must be connected:

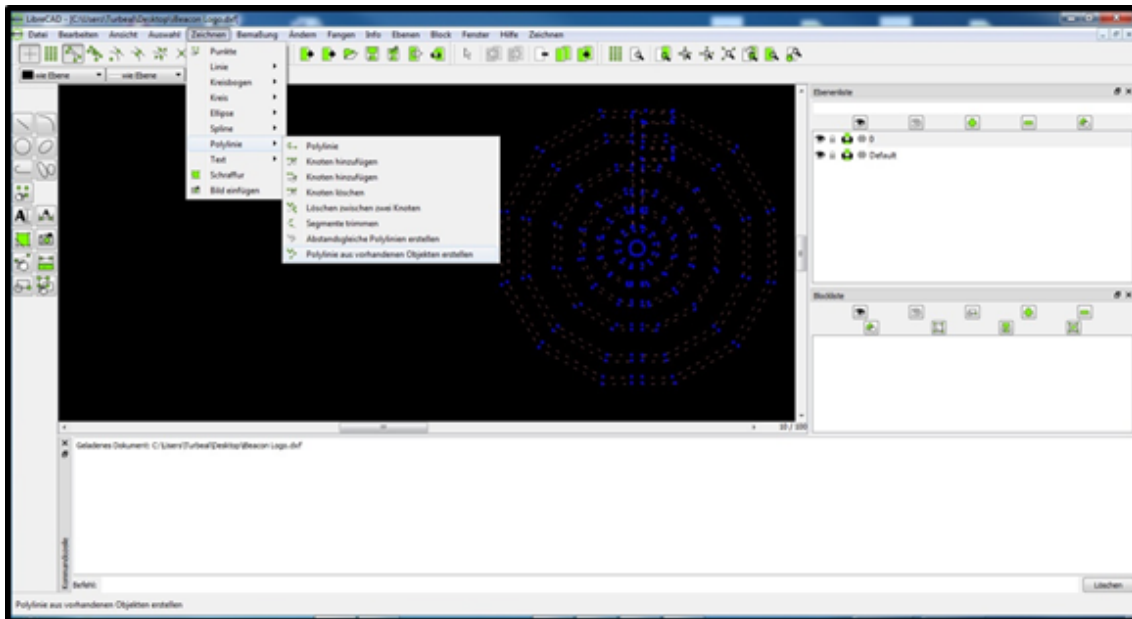
Select the entire object → Draw (Tools) → Polyline → Create Polyline from Existing Objects

Attention: It is possible that only fragments of the object will be connected as a polygon, and this will cause problems in ISO Designer (next step). The object will only be visible in these parts!

In this case, it helps to go through the object step by step again in Solid Edge.

If clicking on the line doesn't produce a purple/pink line, but instead a dashed “purple” line, you may have already found the problem.

Remove this dashed line using, for example, the Trim tool in Solid Edge, and connect the points with a new line.



Repeat this process with all individual objects.

Save Close

15.4 ISO-Designer

- Jetter has removed support for DXF vector graphics import.
- ISO-Designer 5.3.1 is the last version that can still import DXF files.

File → New → Workspace: Name of the .dxf file

File → New → Project: Select the newest platform (2010) Highest resolution (480x480) Name the Display name and Project name with the name of the .dxf file

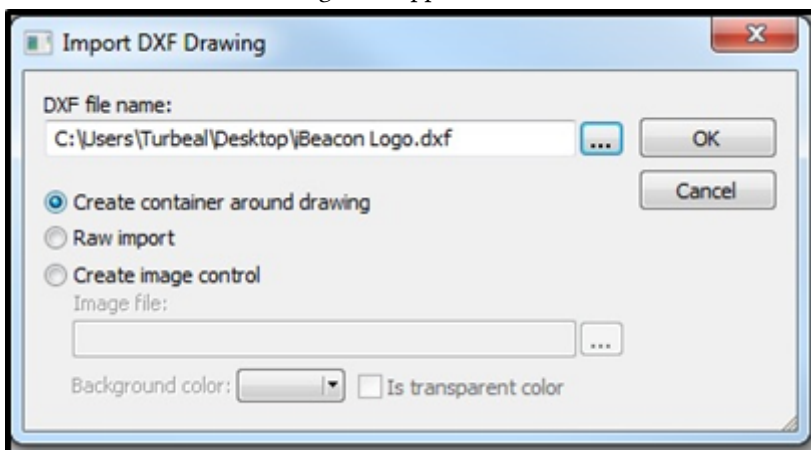
Draw a line or anything else in WorkingSet.jvi

Right-click in DataMask.jvi → Import DXF → Check “create container around drawing”

IMPORTANT: The line/dash must remain in the WorkingSet until the end.

The symbol is also inserted in DataMask. Two different tabs!!!

Otherwise, an error message will appear and it won't work!!!



Save

Close Workspace (due to a bug)

Reopen via Recent Workspaces

Fill in the fill attributes

Build → Build All

Build → Deploy

You should now see “0 Error(s)” under “Deployment”.

If this is not the case, check the line under WorkingSet. (see above)

15.5 Alternative with Autodesk Fusion 360 and Plugin

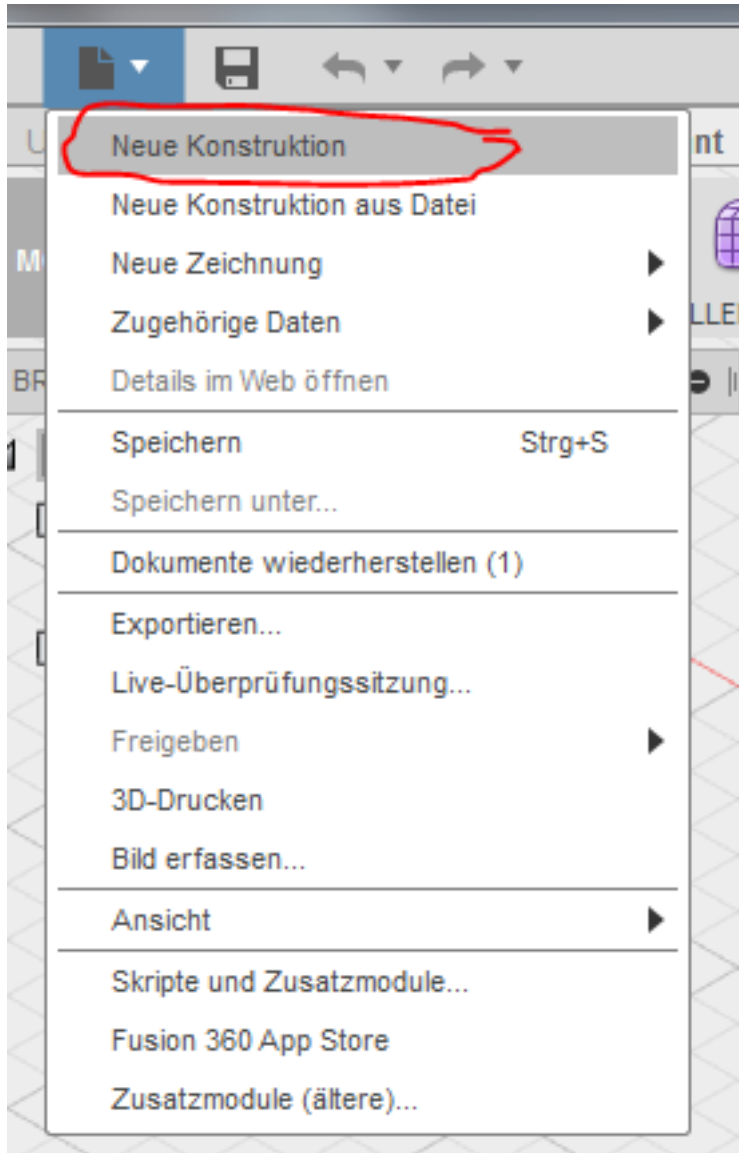
Install the software and plugin Download and install Fusion 360

Free for students.

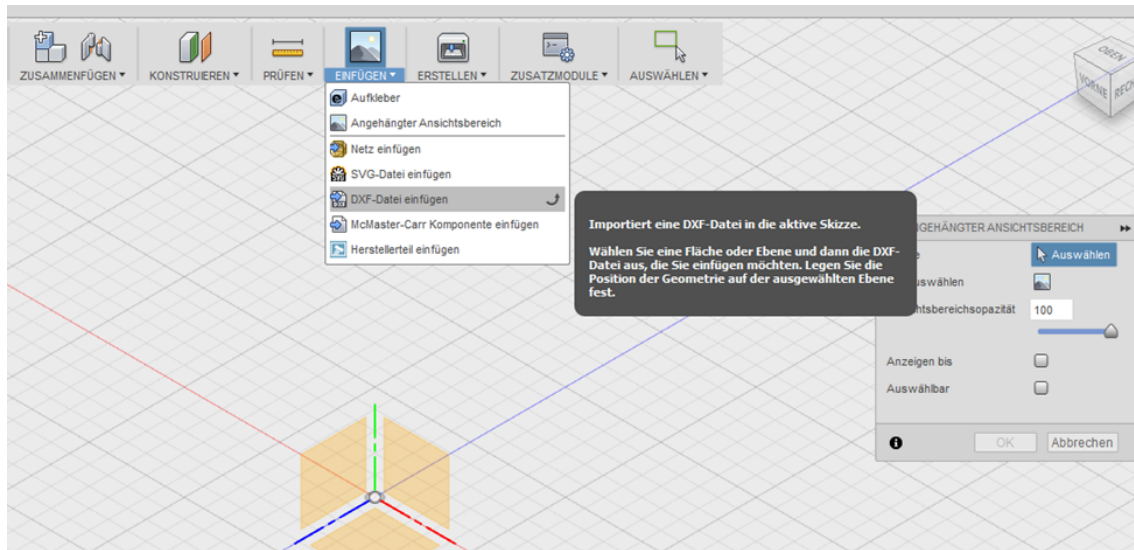
Install the “DXFSplineToPolyline-win64” plugin.

15.5.1 Use Fusion 360

Create a new design



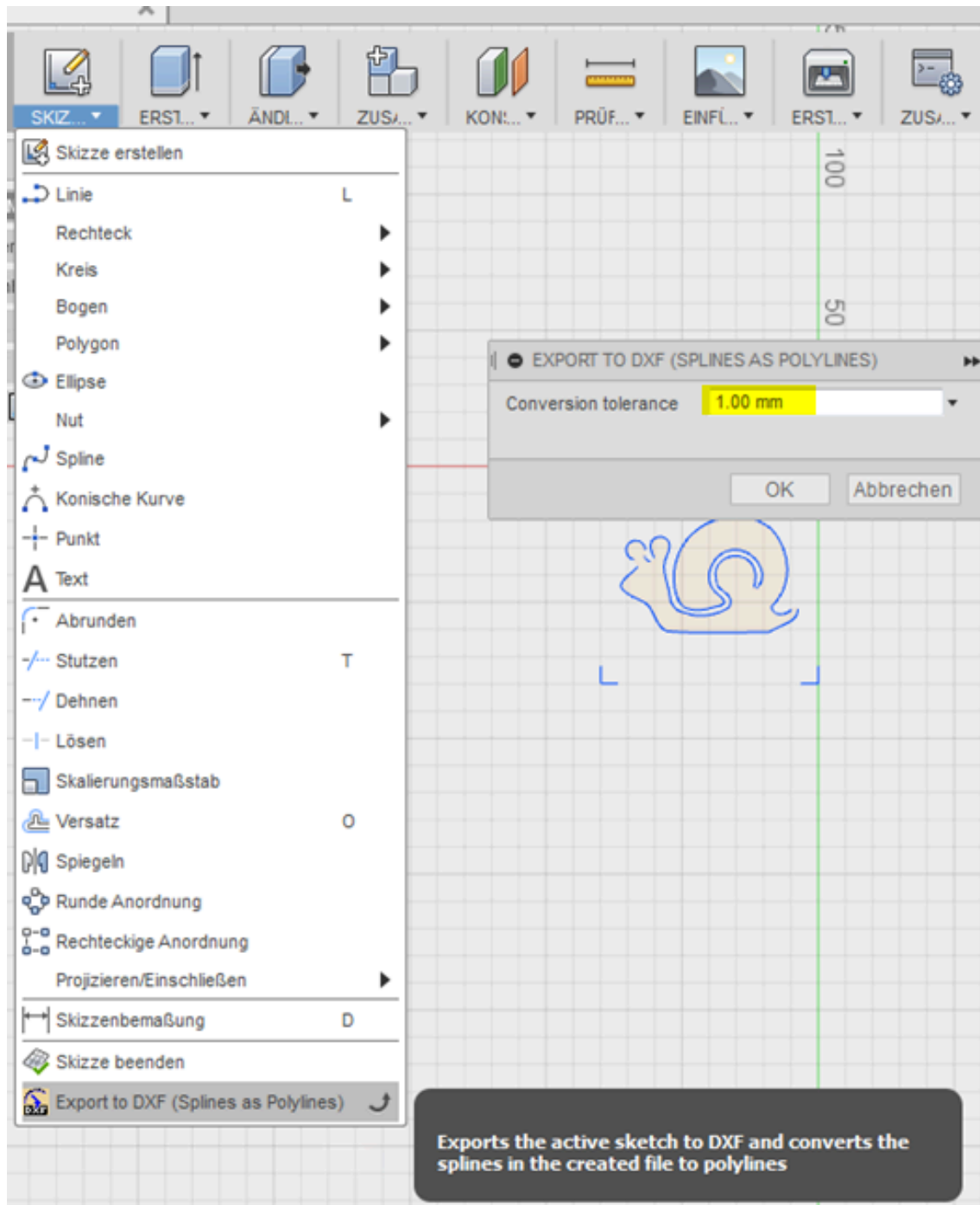
Import DXF (Insert/Insert DXF File) Select the lower layer and file



Optional: Edit as needed, scale (Sketch/Scale Scale (select the scale to match the target size, e.g., 80x80)) and move to the origin (Modify/Move)

The “Export to DXF (Splines as Polyline)” plugin can be run via the Sketch menu.

The tolerance is set to 1.00 mm.



Continue with LibreCAD

16. Videos

16.1 ISO Designer: German

- ISO Designer: German
- 1. ISO Designer: Copying Objects to a Project
- 2. ISO Designer: Aligning Objects
- 3. ISO Designer: Importing and Exporting Text to and from an Excel CSV File
- 4. ISO Designer: Switching the Language of Text and Images
- 5. ISO Designer: Convenient Search Functions
- 6. ISO Designer: Using the Object Library
- 7. ISO Designer: Object Properties adopt

16.2 ISO designer: English

- ISO Designer: English
- 1. ISO Designer: Copy Objects
- 2. ISO Designer: Arranging Objects
- 3. ISO Designer: Using Object-Library
- 4. ISO Designer: Search Functions
- 5. ISO Designer: Inheriting Object-Properties
- 6. ISO Designer: Importing/exporting texts into/from an Excel CSV file
- 7. ISO designer: Language selection within texts and images

17. This is on the curriculum: ISOBUS Virtual Terminal & HMI user interfaces

 **Source & Legal Basis:** **Source:** *Framework Curriculum for the Preparation for the Master Craftsman Examination in Agricultural and Construction Machinery Mechatronics* **Publisher:** German Association of Agricultural and Construction Machinery Mechatronics Technicians (LandBauTechnik-Bundesverband e. V.), Alfredstraße 102, 45131 Essen (as of February 25, 2025) **Regulation:** *Master Craftsman Examination Regulation (LandBauMechMstrV)* of September 9, 2024 (Federal Law Gazette 2024 I No. 277, effective August 1, 2025)

17.1 Relevance of the Virtual Terminal (ISO 11783-6) in the 2025 Curriculum Framework

The **design and configuration of mechatronic user interfaces (human-machine interfaces / HMI)** via the ISOBUS Virtual Terminal (VT according to ISO 11783-6) is a **central core competency** for the master craftsman's examination on modern agricultural and construction machinery.

Modern implements no longer have their own display; instead, they use the Universal Terminal (UT/VT) of the tractor or construction machine. The master craftsman examination regulations require proficiency in these interfaces, both in practical skills (Part I) and in theoretical knowledge (Part II).

17.2 Curriculum Modules & Focus Areas

17.2.1 1. Part I – Learning Unit 1.4: Automated Functions (80 units)

- **Practical Integration of HMI Interfaces:** Linking sensor and actuator values with visual masks on the Virtual Terminal.
- **Object Pool Creation:** Designing data masks, alarm masks, and softkeys using tools such as the ISO Designer.

17.2.2 2. Part II – Learning Unit 1.1: Systems Engineering (120 hours)

- **Human-Machine Interface (HMI/MMS):** Ergonomic design of operator terminals, structuring of object pools according to ISO 11783-6 (attributes, object IDs, graphic objects, output/input objects).
- **Interaction Design:** Configuration of key events (key press/release), parameter changes via touch-screen and rotary encoder.

17.2.3 3. Part II – Learning Unit 1.4: Technical Documentation & Drawings (80 hours)

- **Planning Operating Concepts:** Creation of functional diagrams, screen layouts, softkey hierarchies, and symbol libraries for attachments.
-

17.3 Curriculum topics in this wiki

-  [ISOBUS VT object pools & HMI \(ISO 11783-6\)](#)
-  [Mask layout, softkeys & ergonomics](#)

-  Input attributes, key events & interaction diagnostics

17.3.1 Related topic subpages on ms-muc-docs.de

-  Eclipse 4diac IDE & Color Reference on ms-muc-docs.de

17.4 Main portal & direct links to the 6 wikis

Wiki	Topic / Description	Link
Wiki 0	Wiki of the Master Schools General information, Curriculum 2025 & Main Wiki.	 Open
Wiki 1	C Programming in ISOBUS Fundamentals of C programming and the ISOBUS environment.	 Open
Wiki 2	ISOBUS Virtual Terminal (VT) User interfaces, ISO Designer, VT object pools & HMI.	 Open
Wiki 3	ISOBUS Technology & Machine Systems Hydraulics, Pneumatics, Motors, High-Voltage Technology, Gearboxes & Machine Safety.	 Open
Wiki 4	Visual Programming Languages Low-Code Development, IEC 61499 & 4diac IDE (Main Curriculum Topic).	 Open
Wiki 4.1	4diac Library Reference Function block & adapter reference for the 4diac libraries.	 Open
Wiki 4.2	4diac Exercises: Training Solutions Exercises and sample solutions for training with 4diac.	 Open
Wiki 5	Electronic Tools & Measuring Instruments Handbook for Crimping Technology, Soldering Stations, Oscilloscopes & Microscopes.	 Open

17.4.1 PDF Downloads for All Wikis

Wiki	PDF (DE)	PDF (EN)
Wiki 0 – Main Wiki	 DE	 EN
Wiki 1 – C Programming in ISOBUS	 DE	 EN
Wiki 2 – ISOBUS Virtual Terminal	 DE	 EN
Wiki 3 – ISOBUS Technology & Machine Systems	 DE	 EN
Wiki 4 – Visual Programming Languages	 DE	 EN
Wiki 4.1 – 4diac Library Reference	 DE	 EN
Wiki 4.2 – 4diac Exercises	 DE	 EN
Wiki 5 – Electronic Tools	 DE	 EN

 **Central Portal of the Munich Master Schools:**

Visit the main platform: www.ms-muc-docs.de |  Interactive Curriculum Framework 2025 |  Wiki
Overview |  PDF Downloads

18. logiBUS®

18.1 Podcast

- Digital memories: The SR flip-flop in the logiBUS® system explained
- ESP32 as an industrial PLC: Revolution with Eclipse 4diac and logiBUS®
- From Cable Chaos to Custom Control: How Logibus is Revolutionizing Agricultural Tech with Accessible ISOBUS
- Logibus Revolution: Unpacking the “No Mapping” Future and Streamlined Development
- Logibus Unleashed: How Eliminating “Mapping” Simplifies Complex Systems and Boosts Usability

18.2 Video

- LogiBUS project Update
-

18.2.1 Related topic subpages on ms-muc-docs.de

-  Eclipse 4diac IDE & Color Reference on ms-muc-docs.de
-  ESP32 & ESP32-S3 DevKit on ms-muc-docs.de

19. Objects

ISO 11783-6 specifies a serial data communication network for the control and communication of agricultural and forestry tractors, as well as mounted, semi-mounted, trailed, or self-propelled implements. More precisely, it is a virtual terminal that serves as an interface for controlling and monitoring various implements. With ISOBUS, farmers can control different types of implements via the same terminal, regardless of the manufacturer. This allows for improved clarity and efficiency on the tractor. 🚜

1. ISOBUS – Wikipedia. <https://de.wikipedia.org/wiki/ISOBUS>.
2. ISO 11783-6:2018. <https://www.iso.org/standard/71173.html>.
3. ISO 11783-6 - 2018-06 - DIN Media - Beuth. <https://www.dinmedia.de/de/norm/iso-11783-6/291680983>.
4. Rothmund Wodok. ISOBUS - Standard ISO 11783 - <gil-net.de>. https://gil-net.de/Publikationen/22_163.pdf.
5. en.wikipedia.org. https://en.wikipedia.org/wiki/ISO_11783.

19.1 Attributes

19.1.1 Recommended Reading

- Jetter ISO Designer Help
- ISO Objects
- The Individual ISO Objects
- Font, Line, Fill, and Input Attributes

The screenshot shows the ISO-Designer application window. On the left is a tree view of the software's content, with 'Font, Line, Fill- und Input-Attributes' selected. The main area on the right displays the help text for this topic. The title is 'Font-, Line-, Fill- und Input-Attributes'. Below it is an 'Einleitung' (Introduction) section, followed by a 'Font Attributes' section which includes a table of attributes and their descriptions.

Font-, Line-, Fill- und Input-Attributes

Einleitung

Die Objekte Font-, Line-, Fill- und Input-Attributes dienen dazu, bestimmte Eigenschaften von Objekten zu gruppieren. Die Objekte Font-, Line-, Fill- und Input-Attributes gruppieren Eigenschaften, die z. B. das Aussehen von Texten, Werten, Linien und Füllungen definieren. Sie können die Objekte als Unterobjekte wiederverwenden, wenn mehrere Objekte den gleichen Satz von Eigenschaften haben sollen. Die Objekte sind im **Object Pool** vorhanden. Sie können im [Fenster Properties](#) die einzelnen Eigenschaften anzeigen und anpassen.

Font Attributes

Die **Font Attributes** gruppieren Eigenschaften, die das Aussehen von Texten definieren.

Font Attributes/Font	Beschreibung
Size	Schriftgröße
Color	Farbe des Objekts
Bold	Fettschrift
Crossed Out	Durchgestrichene Schrift
Underline	Unterstrichene Schrift
Italic	Kursive Schrift
Inverted	Invertierte Schrift
Flash Inverted	Hintergrund- und Schriftfarben wechseln
Flash Hidden	Schrift blinkt

19.1.2 Podcast

- ISOBUS Interfaces: Background Color vs. Fill Attributes – How to Optimize Your UI and Performance]

19.2 Issue

19.3 Auxiliary Functions

19.4 Controls

ISO 11783-6 specifies various controls used in agricultural and forestry tractors and implements. Here are the relevant controls according to ISO 11783-6:

1. **Key (Soft Key)** (ID 5): Softkeys are programmable buttons on the virtual terminal. The softkey interface allows these keys to be customized depending on the context and implement. The operator can use them to perform specific functions or adjust settings.¹
2. **Button** (ID 6): Buttons are physical buttons or keys on the control panel. They are used to trigger specific actions, such as starting or stopping an implement or changing settings.¹ These controls are essential for the efficient control and monitoring of implements via the virtual terminal. If you have any further questions, please feel free to ask! 🚜

Source: Conversation with Copilot, July 18, 2024

1. ISOBUS in Functionalities - AEF Online. https://www.aef-online.org/fileadmin/user_upload/Content/pdfs/AEF_handfan_DE.pdf.
2. ISO 11783-6:2018. <https://www.iso.org/standard/71173.html>.
3. ISO 11783-6 - 2014-07 - DIN Media. <https://www.dinmedia.de/de/norm/iso-11783-6/215929001>.
4. ISO CONTROL by Reichhardt | PRILLINGER Spare Parts with Service Plus. <https://www.prillinger.at/de/p/ISO-CONTROL-Joystick~00570058>.
5. ISOBUS Retrofit - Reichhardt. <https://www.reichhardt.com/alle-produkte/isobus-kompetenz/>.
6. ISO 11783-6:2018(en), Tractors and machinery for agriculture and <https://www.iso.org/obp/ui/en/>.
7. ISO 11783 – STANDARD AND ITS IMPLEMENTATION - NTNU. <https://folk.ntnu.no/skoge/prost/proceedings/ifac2005/Fullpapers/02392.pdf>.
8. INTERNATIONAL ISO This is a preview of ISO 11783-6:2018. Click here to https://webstore.ansi.org/preview-pages/ISO/preview_ISO+11783-6-2018.pdf.
9. ISOBUS Maskmaker - ANEDO. <https://anedo.eu/produkt/isobus-mask-maker/>.
10. Breathing resistance of masks - DGUV Publications. <https://publikationen.dguv.de/widgets/pdf/download/article/4577>.
11. Overview: Types of respiratory masks | Properties of protective masks <https://www.heller-medizintechnik.de/informationen/patienten/schutzmasken/>.
12. Breathing resistance of masks - DGUV Information Portal. https://www.dguv.de/medien/ipa/publikationen/ipa-journale/ipa-journale2022/ipa_journal_1_2022_atemwiderstaende_masken.pdf.

19.5 Pictures

19.6 Container

19.6.1 Podcast

- [ISOBUS Container: Dynamic Control Panels for Clear View and Greater Efficiency

19.7 Input

19.8 External Objects

19.9 Functions

19.10 Masks

19.10.1 Podcast

- [ISOBUS Masks in the Virtual Terminal: The Heart of Agricultural and Construction Machinery Interfaces \(ISO 11783-6\)](#)

ISO 11783-6 describes a serial data communication network for agricultural and forestry tractors, as well as mounted, semi-mounted, trailed, or self-propelled implements. A central element is the virtual terminal, which serves as an interface for controlling and monitoring various implements. The mask types according to ISO 11783-6 are listed here:

1. **Working Set (ID 0):** This object defines the display of data from different working sets on a single mask. It enables the simultaneous display of information from multiple implements on the same virtual terminal.
2. **Data Mask (ID 1):** The data mask is a rectangular area on the display with a minimum resolution of 200 x 200 pixels. It displays various objects such as buttons, number and character fields, geometric shapes, gauges, and bar charts².
3. **Alarm Mask (ID 2):** This mask is used to display alarms and warnings. It informs the operator about critical conditions or events, such as engine overheating or low fuel level.
4. **Soft Key Mask (ID 4):** Softkeys are programmable buttons on the virtual terminal. The softkey mask allows these buttons to be customized depending on the context and implement. The operator can use them to perform specific functions or adjust settings¹.

I hope this helps! If you have any further questions, please feel free to ask. 🚜

Source: Conversation with Copilot, July 18, 2024

1. ISO 11783 – STANDARD AND ITS IMPLEMENTATION - NTNU. <https://folk.ntnu.no/skoge/prost/proceedings/ifac2005/Fullpapers/02392.pdf>.
2. ISO 11783-6:2018(en), Tractors and machinery for agriculture and <https://www.iso.org/obp/ui/en/>.
3. INTERNATIONAL ISO This is a preview of ISO 11783-6:2018. Click here to https://webstore.ansi.org/preview-pages/ISO/preview_ISO+11783-6-2018.pdf.
4. ISOBUS Maskmaker - ANEDO. <https://anedo.eu/produkt/isobus-mask-maker/>.
5. Breathing resistance of masks - DGUV Publications. <https://publikationen.dguv.de/widgets/pdf/download/article/4577>.
6. Overview: Types of respiratory masks | Properties of protective masks <https://www.heller-medizintechnik.de/informationen/patienten/schutzmasken/>.
7. Breathing resistance of masks - DGUV Information Portal. https://www.dguv.de/medien/ipa/publikationen/ipa-journale/ipa-journale2022/ipa_journal_1_2022_atemwiderstaende_masken.pdf.

19.11 Specials & Miscellaneous

...

19.12 Variables

19.12.1 Recommended reading

- ISO 11783-6
- Chapter B.13.2

- Chapter B.13.3
- Jetter ISO Designer Help
- ISO Objects
- The individual ISO objects
- Number and String variables

ISO-Designer

Ausblenden Zurück Drucken Optionen

Inhalt Index Suchen

Vorspann
 So können Sie uns erreichen
 Versionsupdates
 Produktbeschreibung
 Erstes ISO-Projekt anlegen
 ISO-Designer konfigurieren
 ISO-Objekte
 Objekteigenschaften (Properties)
 Die einzelnen ISO-Objekte
 Das Objekt Line
 Das Objekt Polyline
 Das Objekt Polygon
 Das Objekt Ellipse
 Das Objekt Rectangle
 Das Objekt Button
 Das Objekt Object Pointer
 Das Objekt Input Boolean
 Das Objekt Input Number
 Das Objekt Input String
 Das Objekt Input List
 Das Objekt Output String
 Das Objekt Output Number
 Das Objekt Output List
 Das Objekt Soft Key
 Das Objekt Linear Bargraph
 Das Objekt Arched Bargraph
 Das Objekt Meter
 Das Objekt Container
 Das Objekt Macro
 Das Objekt Graphics Context
 Das Objekt Color Map
 Das Objekt Picture Graphic
 Animation object
 External Object Pointer
 External Object Definition
 External Reference NAME
 Aux Input
 Aux Function
 Aux Object Pointer
Number- und Stringvariablen
 Font-, Line-, Fill- und Input-Att
 Objekte erstellen
 Die verschiedenen Maskentypen
 ISO-konforme Masken erstellen
 Handling von Objekten (Object Pool u
 Der Event Handler
 Mehrsprachigkeit und Ressourcenver
 IOP-Datei

Number- und Stringvariablen

Einleitung

Mit den Objekten **Number Variable** und **String Variable** speichern Sie Integer-Werte und Strings. Folgendes Bild zeigt ein Objekt **Input Number** auf einer Maske und einen Ausschnitt des zugehörigen **Properties-Fenster** mit einer **Number Variable**:

Eine Number Variable anlegen

Eine **Number Variable** dient zum Speichern eines Integer-Werts. Eine **Number Variable** legen Sie wie folgt an:

Schritt	Vorgehen
1	Wählen Sie das Objekt aus, dem Sie die Number Variable zuweisen wollen.
2	Wählen Sie im Fenster Properties unter Number Variable die Zeile Object ID aus.
3	Klappen Sie die Dropdown-Liste auf.
4	Wählen Sie aus der Liste eine Number Variable aus oder legen Sie über Add new

20. Reference Tables (Annex A)

This section contains important reference tables from Annex A of the ISO 11783-6 standard. These are particularly helpful for software development (C/C++ code) and creating macros.

20.1 Command Overview (ISO 11783-6 Table A.5)

This table lists all commands (messages) that can be exchanged between the ECU (Working Set) and the VT.

- **Function (Decimal/Hex):** The code in the first byte of the message (for Destination Specific PGN).
- **Allowed in Macro:** Whether the command may be stored in a macro object. | Clause | Message Name | PGN | Function (Hex) | Macro? | VT Vers. | | :---- | :----- | :-- | :----- | :-- | :--- | |
C.2.3 | Object Pool Transfer | ECU->VT | 11 | No | 2 | | C.2.4 | End of Object Pool | ECU->VT | 12 | No | 2 | | D.4 | Get Number of Soft Keys | ECU->VT | C2 | No | 2 | | D.6 | Get Text Font Data | ECU->VT | C3 | No | 2 | | D.8 | Get Hardware Message | ECU->VT | C7 | No | 2 | | D.12 | Get Window Mask Data | ECU->VT | C4 | No | 4 | | D.14 | Get Supported Objects | ECU->VT | C5 | No | 4 | | D.16 | Screen Capture | ECU->VT | C6 | No | 6 | | D.18 | Identify VT | ECU->VT | BB | No | 4 | | E.2 | Get Versions | ECU->VT | DF | No | 2 | | E.4 | Store Version | ECU->VT | D0 | No | 2 | | E.6 | Load Version | ECU->VT | D1 | No | 2 | | E.8 | Delete Version | ECU->VT | D2 | No | 2 | | E.10 | Extended Get Versions | ECU->VT | D3 | No | 5 | | E.12 | Extended Store Version | ECU->VT | D4 | No | 5 | | E.14 | Extended Load Version | ECU->VT | D5 | No | 5 | | E.16 | Extended Delete Version | ECU->VT | D6 | No | 5 | | F.2 | Hide/Show Object | ECU->VT | A0 | Yes | 2 | | F.4 | Enable/Disable Object | ECU->VT | A1 | Yes | 2 | | F.6 | Select Input Object | ECU->VT | A2 | Yes | 2 | | F.8 | ESC Command | ECU->VT | 92 | No | 2 | | F.10 | Control Audio Signal | ECU->VT | A3 | Yes | 2 | | F.12 | Set Audio Volume | ECU->VT | A4 | Yes | 2 | | F.14 | Change Child Location | ECU->VT | A5 | Yes | 2 | | F.16 | Change Child Position | ECU->VT | B4 | Yes | 2 | | F.18 | Change Size | ECU->VT | A6 | Yes | 2 | | F.20 | Change Background Color | ECU->VT | A7 | Yes | 2 | | F.22 | Change Numeric Value | ECU->VT | A8 | Yes | 2 | | F.24 | Change String Value | ECU->VT | B3 | Yes | 2 | | F.26 | Change End Point | ECU->VT | A9 | Yes | 2 | | F.28 | Change Font Attributes | ECU->VT | AA | Yes | 2 | | F.30 | Change Line Attributes | ECU->VT | AB | Yes | 2 | | F.32 | Change Fill Attributes | ECU->VT | AC | Yes | 2 | | F.34 | Change Active Mask | ECU->VT | AD | Yes | 2 | | F.36 | Change Soft Key Mask | ECU->VT | AE | Yes | 2 | | F.38 | Change Attributes | ECU->VT | AF | Yes | 2 | | F.40 | Change Priority | ECU->VT | B0 | Yes | 2 | | F.42 | Change List Item | ECU->VT | B1 | Yes | 2 | | F.44 | Delete Object Pool | ECU->VT | B2 | No | 2 | | F.46 | Lock/Unlock Mask | ECU->VT | BD | Yes | 4 | | F.48 | Execute Macro | ECU->VT | BE | Yes | 4 | | F.50 | Change Object Label | ECU->VT | B5 | Yes | 4 | | F.52 | Change Polygon Point | ECU->VT | B6 | Yes | 4 | | F.54 | Change Polygon Scale | ECU->VT | B7 | Yes | 4 | | F.56 | Graphics Context Command | ECU->VT | B8 | No | 4 | | F.58 | GetAttributeValue | ECU->VT | B9 | No | 4 | | F.60 | Select Color Map/Palette | ECU->VT | BA | Yes | 4 | | F.62 | Execute Extended Macro | ECU->VT | B.C. | Yes | 5 | | F.64 | Select Active Working Set | ECU->VT | 90 | Yes | 6 |

20.2 Event IDs (ISO 11783-6 Table A.3)

This table lists all event IDs that can be used in macros (Macro object, ID 28) or as triggers for attributes.

Note: Starting with VT version 5, 16-bit macro references are supported.

Event ID	Event Name	Trigger (Occurs when)
0	Reserved	-
1	On Activate	Working set becomes active.
2	On Deactivate	Working set becomes inactive.
3	On Show	Container/mask becomes visible.
4	On Hide	Container/mask becomes invisible.

5	On Enable	Object is enabled (input enabled).
6	On Disable	Object is disabled (Input disabled).
7	On Change Active Mask	The active data mask has been changed.
8	On Change Soft Key Mask	The soft key mask has been changed.
9	On Change Attribute	An attribute has been changed.
10	On Change Background Colour	Background color has been changed.
11	On Change Font Attributes	Font attributes have been changed.
12	On Change Line Attributes	Line attributes have been changed.
13	On Change Fill Attributes	Fill attributes have been changed.
14	On Change Child Location	The position of a child object has been changed.
15	On Change Size	The size of an object has been changed.
16	On Change Value	The value (number/string) has changed.
17	On Change Priority	The alarm priority has changed.
18	On Change End Point	The endpoint of a line has been changed.
19	On Input Field Selection	Focus retained (navigation).
20	On Input Field Deselection	Focus lost.
21	On ESC	Input canceled (ESC).
22	On entry of a value	Value confirmed (ENTER), no change needed.
23	On entry of a new value	Value confirmed (ENTER) AND changed.
24	On Key Press	Key/button pressed.
25	On Key Release	Key/button released.
26	On Change Child Position	Relative position changed in the container (scrolling).
27	On Pointing Event Press	Touch display pressed.
28	On Pointing Event Release	Touch display released.
29-239	Reserved	-
240-254	Proprietary Events	Manufacturer-specific events.
255	Reserved	-
255	Use Extended Macro Ref.	(Only in byte stream) Indicates 16-bit macro ID.

20.3 Standard Color Palette (ISO 11783-6 Table A.4)

The VT standard color palette defines 256 colors. The first 16 colors are identical to the standard VGA palette.

Index	Name / Description	R, G, B Value	Hex Code (RGB)
0	Black	0, 0, 0	#000000
1	White	255, 255, 255	#FFFFFF
2	Green	0, 153, 0	#009900
3	Teal	0, 153, 153	#009999

4	Maroon	153, 0, 0	#990000
5	Purple	153, 0, 153	#990099
6	Olive	153, 153, 0	#999900
7	Silver	204, 204, 204	#CCCCCC
8	Gray	153, 153, 153	#999999
9	Blue	0, 0, 255	#0000FF
10	Lime	0, 255, 0	#00FF00
11	Cyan	0, 255, 255	#00FFFF
12	Red	255, 0, 0	#FF0000
13	Magenta	255, 0, 255	#FF00FF
14	Yellow	255, 255, 0	#FFFF00
15	Navy	0, 0, 153	#000099
...	...	...	...
232-255	Proprietary	-	VT manufacturer specific

Note: For indices 16 to 231, an algorithmic distribution is defined in the color cube (6x6x6). See ISO 11783-6 Table A.4 for the complete list.

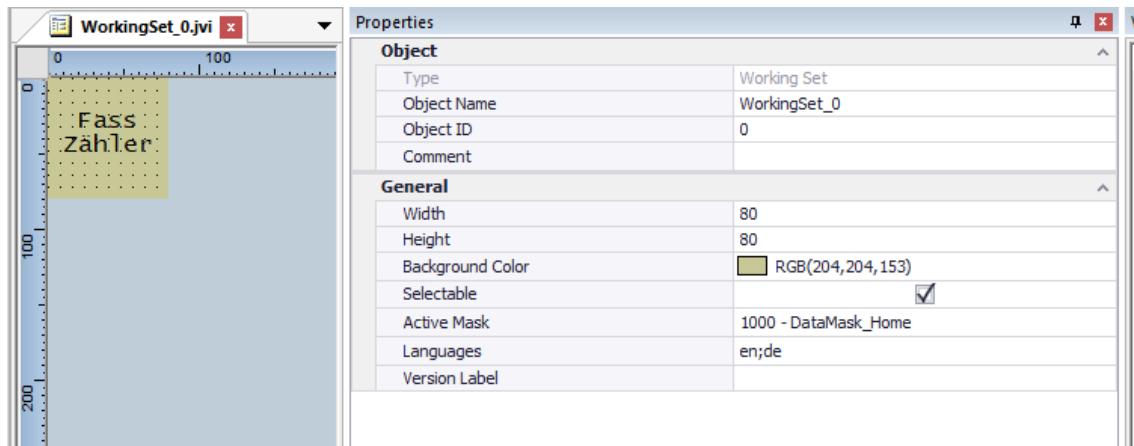
Further information and visualizations can be found in the [ISOBUS Wiki - Colours](#) by Tobias Tenberg.]

20.3.1 Transparency

The color index specified in the “Transparency Colour” attribute (e.g., in Picture Graphics) is not drawn. Often, a “signal color” such as magenta (index 13) is used if it is not present in the image.

21. Isobus objects

21.1 ID 0 – Working set – ISO 11783-6 – B.1



The **Working Set** object with **ID 0** is the central management element of a workgroup (Working Set) in ISOBUS. It defines how the workgroup presents itself to the Virtual Terminal (VT) and which interface is initially displayed.

21.1.1 Key properties (according to ISO 11783-6, Annex B.1)

Each workgroup must define **exactly one** Working Set object in its object pool. Only the VT can activate this object.

21.1.1.1 Attributes and Record Format (Table B.2)

The following table describes the structure of the Working Set object in the object pool (byte stream).

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	0	3	Object type = Working Set.
[1]	Background colour	Integer	1	0 – 255	4	Background color.
[2]	Selectable	Boolean	1	0 or 1	5	0 = FALSE, 1 = TRUE. Indicates whether the working set can be selected by the operator.
[3]	Active mask	Integer	2	0 – 65534	6 – 7	Object ID of the data or alarm mask displayed

						when the working set is active.
-	Number of objects to follow	Integer	1	1 – 255	8	Number of following child objects (designator). Must be at least 1.
-	Number of macros to follow	Integer	1	0 – 255	9	Number of following macro references.
-	Number of languages to follow	Integer	1	0 – 255	10	Number of following language codes.
-	Repeat: {Object ID}	Integer	2	0 – 65534	11 + ...	Object ID of a child object (part of the designator). List of all objects <i>before</i> the macros.
-	{X Location}	Signed Integer	2	-32768 to +32767	13 + ...	Relative X position of the child object (pixels).
-	{Y Location}	Signed Integer	2	-32768 to +32767	15 + ...	Relative Y position of the child object (pixels).
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After the objects) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed (or reference to the 16-bit Macro ID).
-	Repeat: {Language Code}	String	2	-	var.	(After the macros) Two-letter language code according to ISO 639 (e.g., “de”, “en”).

21.1.1.2 Designator (Child Objects)

The Working Set object serves as a container for a graphical identifier (designator) that represents the device (e.g., icon and name).

- **Structure:** The number of objects specified under Number of objects to follow follows directly in the record. Each child object is defined by its ID, X-Pos, and Y-Pos.
- **Space Limit:** All of these objects must fit within the area of a **softkey**. The VT truncates anything that extends beyond this area.
- **Coordinates:** Relative to the upper left corner of the Working Set object.

21.1.1.3 Macros (Events)

The working set defines events that can trigger macros. See Table B.1 for the event definitions.

- **On Activate:** Triggered when the working set is selected by the operator.
- **On Deactivate:** Triggered when the working set is exited.
- **On Change Active Mask:** Triggered by the command Change Active Mask.

21.1.1.4 Language Support

The list of language codes (Language Code) informs the VT which languages the working set supports. Each code consists of two ASCII characters (e.g., “de”, “en”, “fr”). This is for informational purposes; the actual language switching is performed by loading the corresponding language objects or texts.

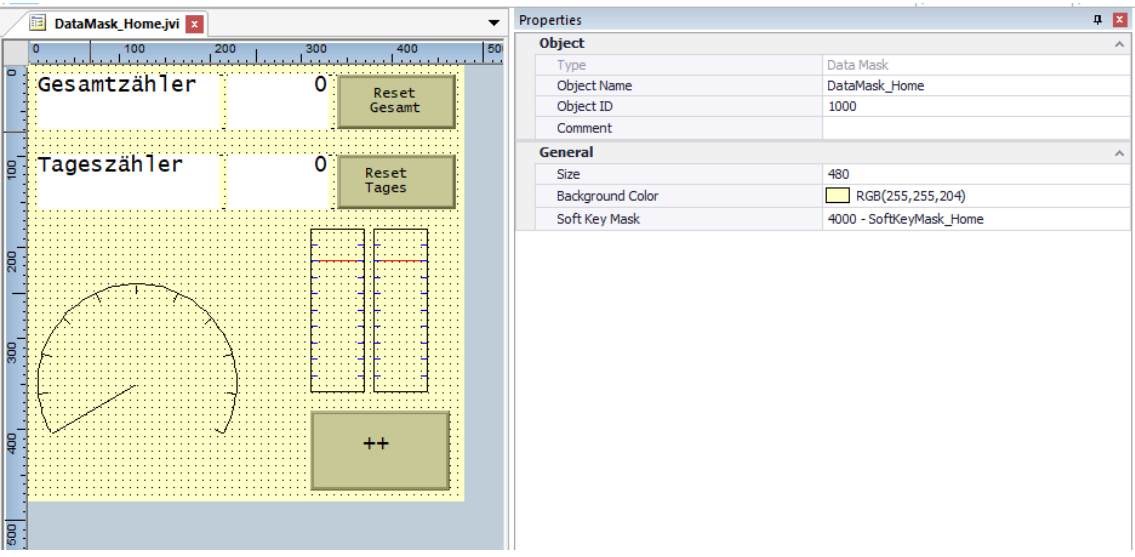
21.1.2 Implementation

The working set object is the “anchor” of an application on the VT. Without a correctly defined object ID 0, the VT cannot identify the working group or load the first form. Developers must ensure that the Active mask AID points to a valid data form in the pool.

Further information and examples can be found in the [ISOBUS Wiki - Working Set](#) by Tobias Tenberg.

Note: For detailed implementation information on data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.2 ID 1 – Data mask – ISO 11783-6 – B.2



The **Data Mask** with **ID 1** is the primary display element for the user interface of a workgroup. It serves as the main container for all visual objects (buttons, number fields, graphics) displayed to the operator on the Virtual Terminal (VT).

21.2.1 Attributes and Record Format (Table B.4)

The following table describes the structure of the Data Mask object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	1	3	Object type = Data mask.
[1]	Back-ground color	Integer	1	0 – 255	4	Back-ground color of the mask.
[2]	Soft Key Mask	Integer	2	0 – 65534, 65535	5 – 6	Object ID of the associated soft key mask (65535 = NULL).
[-]	Number of objects to follow	Integer	1	0 – 255	7	Number of directly contained objects.
[-]	Number of macros to follow	Integer	1	0 – 255	8	Number of following macro references.
[-] Repeat: {Object ID}	Integer	2	0 – 65534	9 + ...	Object ID of a contained object.	
-	{X Location}	Signed Integer	2	-32768 to +32767	11 + ...	X-position relative to the mask (pixels).
-	{Y Location}	Signed Integer	2	-32768 to +32767	13 + ...	Y-position relative to the mask (pixels).
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After objects) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.2.1.1 Structure and Child Objects

The data mask serves as a container for all visible elements.

- **Object List:** Each child object is defined by its ID and its position (X/Y). Each set of ID and position occupies 6 bytes.

- **Order:** Objects are drawn in the listed order (Z-order). Higher indices are positioned above lower ones.
- **Coordinates:** Positioning is absolute in VT pixels, relative to the upper left corner of the mask (0,0).

21.2.2 Events (Table B.3)

The data mask responds to the following events:

- **On Show:** Triggered when the mask becomes visible. The VT draws the background, the children, and the soft key mask.
- **On Hide:** Triggered when the mask is removed from the display.
- **On Refresh:** Triggered when changes are made to child objects that require redrawing.
- **On Change Background Colour:** Response to a change in the background colour.
- **On Change Soft Key Mask:** Response to a change in the assigned soft key mask.
- **Pointing Events:** press and release during touch operation on the mask area.

21.2.3 Behavior and Limitations

- **Relationship with Softkeys:** Every data mask “has” a softkey mask. When the data mask is changed, the VT usually also changes the softkey layout.
- **Refresh:** When a child object is changed, the VT redraws the affected areas.
- **Visibility:** Only one data mask (or alarm mask) per workgroup can be active and in focus at any given time.

21.2.4 Implementation Importance

The data mask is the core of the HMI design. Developers must ensure that the mask resolution matches the capabilities of the VT (standard minimum resolution is often 200x200 pixels, but modern VTs offer significantly higher resolutions). Efficient use of macros on mask events (e.g., On Show) can facilitate initializations directly within the VT.

Further information and examples can be found in the ISOBUS Wiki - Data Mask](<https://isobus-studio.com/isobus-wiki/isobus-objectpool-objects/datamask>) by Tobias Tenberg.

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.3 ID 2 – Alarm Mask – ISO 11783-6 – B.3

The **Alarm Mask** with **ID 2** is used to display critical information or warnings. It takes precedence over normal data masks and, depending on its priority, can overlay the entire display or parts of it.

21.3.1 Attributes and Record Format (Table B.6)

The following table describes the structure of the Alarm Mask object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	2	3	Object type = Alarm Mask.
[1]	Background color	Integer	1	0 – 255	4	Background color of the mask.
[2]	Soft Key Mask	Integer	2	0 – 65534, 65535	5 – 6	Object ID of the associated soft key mask (65535 = NULL).

[3]	Priority	Integer	1	0 – 2	7	0 = High, 1 = Medium, 2 = Low.
[4]	Acoustic signal	Integer	1	0 – 3	8	0 = Highest priority, 1 = Medium, 2 = Low, 3 = Off.
[-]	Number of objects to follow	Integer	1	0 – 255	9	Number of directly contained objects.
-	Number of macros to follow	Integer	1	0 – 255	10	Number of following macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65534	11 + ...	Object ID of a contained object.
-	{X Location}	Signed Integer	2	-32768 to +32767	13 + ...	X position relative to the mask (pixels).
-	{Y Location}	Signed Integer	2	-32768 to +32767	15 + ...	Y position relative to the mask (pixels).
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(By Object) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.3.1.1 Priority Levels and Display

The priority not only controls the order of the alarms, but often also their visual display on the VT:

- **High Priority (0):** The operator is in danger or a serious malfunction has occurred. These alarms must be acknowledged immediately or force the terminal to focus.
- **Medium Priority (1):** Normal machine malfunction.
- **Low Priority (2):** For information only (status messages).

21.3.2 Events (Events - Table B.5)

The alarm screen reacts to the following events:

- **On Show:** Triggered when the screen becomes visible. The VT sends a VT Status message.
- **On Hide:** Triggered when the mask is removed from the display.
- **On Refresh:** Triggered when objects within the mask are changed.
- **On Change Priority:** When the priority changes, the VT re-evaluates all active alarms.
- **On Change Soft Key Mask:** Changes the softkey assignment when an alarm mask is active.
- **On Change Child Location / Position:** Updates child objects.
- **On Change Attribute:** Reacts to general attribute changes.

21.3.3 Behavior with Multiple Alarms

If multiple alarm masks from different workgroups are active simultaneously, the priority (AID 3) determines which mask takes precedence. If the priority is the same, the VT usually decides (often based on chronological order).

21.3.4 Implementation Implementation Implementation Implementation Masks should be used sparingly and only for genuine warnings, as they interrupt the operator's workflow. It is important to select the appropriate audible alert (AID 4) to guide attention without stressing the operator

Further information and examples can be found in the ISOBUS Wiki - Alarm Mask](<https://isobus-studio.com/isobus-wiki/isobus-objectpool-objects/alarm-mask>) by Tobias Tenberg.

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018.

21.4 ID 3 – Container – ISO 11783-6 – B.4

The **Container** object with **ID 3** is used to logically group multiple objects. A container itself is not visible, but it allows you to move, show/hide, or split an entire group of objects.

21.4.1 Attributes and Record Format (Table B.8)

The following table describes the structure of the Container object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	3	3	Object type = Container.
[1]	Width	Integer	2	0 – 65535	4 – 5	Maximum width of the container in pixels. Clipping occurs outside this range.
[2]	Height	Integer	2	0 – 65535	6 – 7	Maximum height of the container in pixels. Clipping occurs outside this range.
[3]	Hidden	Boolean	1	0 or 1	8	0 = FALSE (Visible), 1 = TRUE (Hidden). Indicates whether the container is initially hidden.

[-]	Number of objects to follow	Integer	1	0 – 255	9	Number of objects directly contained within.
[-]	Number of macros to follow	Integer	1	0 – 255	10	Number of subsequent macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65534	11 + ...	Object ID of a contained object.
-	{X Location}	Signed Integer	2	-32768 to +32767	13 + ...	X-position relative to the container (pixels).
-	{Y Location}	Signed Integer	2	-32768 to +32767	15 + ...	Y-position relative to the container (pixels).
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After objects) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.4.1.1 Relative Positioning and Clipping

Within a container, a **separate coordinate system** begins:

- **Relative Coordinates:** The X and Y positions of child objects are referenced to the upper left corner of the container.
- **Clipping:** All objects or parts of objects that lie outside the area defined by Width and Height are clipped by the container and not displayed.
- **Group Move:** When the container is moved (e.g., via `Change Child Position`), all objects within it move automatically.

21.4.2 Events (Table B.7)

The container reacts to the following events:

- **On Show:** Triggered by the command `Show Object` (even if the container is already visible). The VT redraws the contained objects.
- **On Hide:** Triggered by the command `Hide Object`. The container is overdrawn with the background color of the parent mask.
- **On Refresh:** Triggered when changes are made to child objects that require redrawing.
- **On Change Child Location / Position:** Updates the position of child objects.
- **On Change Size:** Reacts to changes in the container's size.

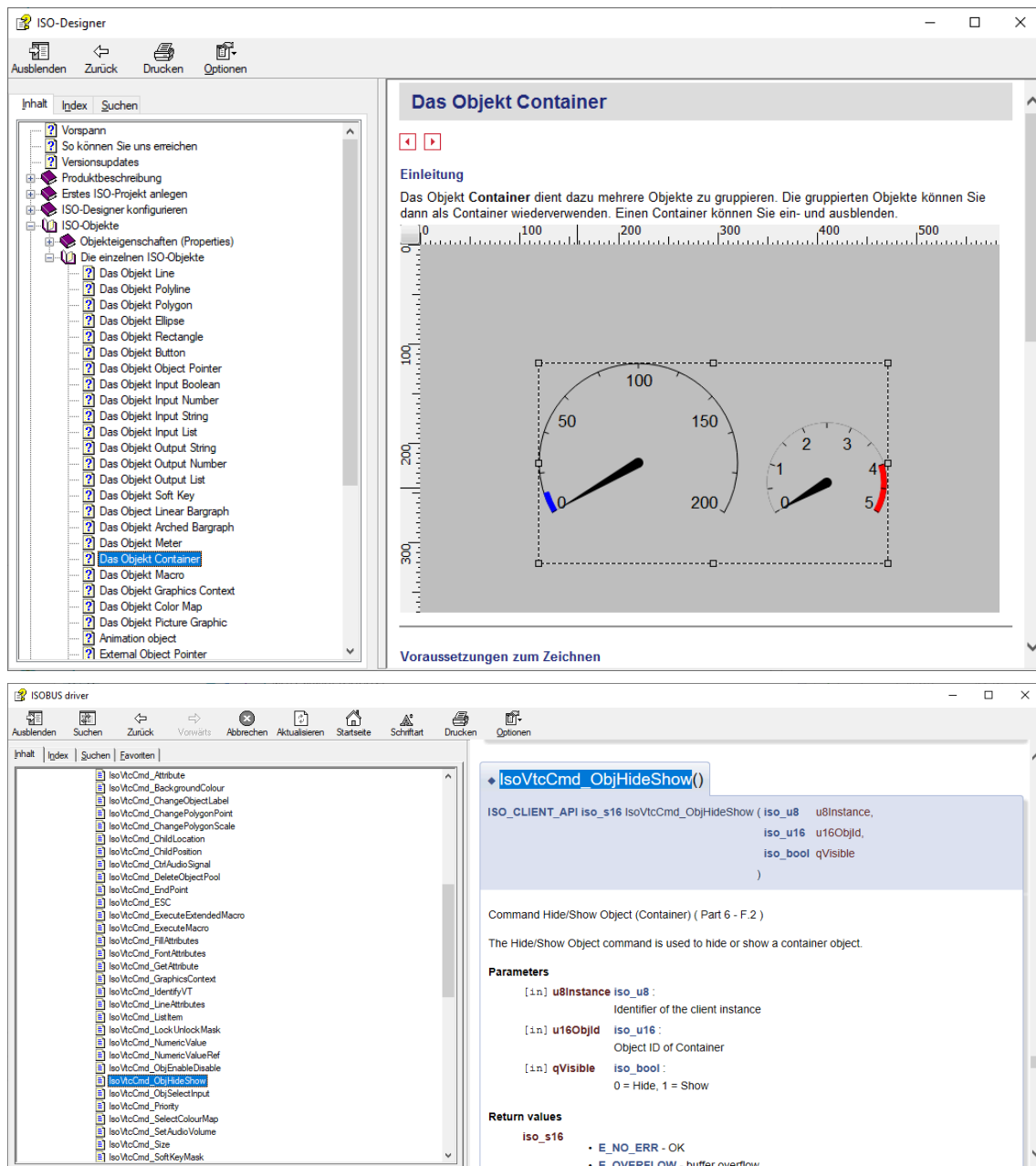
21.4.3 Practical Use

Containers are essential for dynamic user interfaces:

- **Show/Hide:** Complex control panels or status indicators can be shown or hidden at the touch of a button using the command `IsoVtcCmd_ObjHideShow`.

- **Space Saving:** Multiple containers can reside in the same location; by cleverly switching their visibility, different “tabs” or modes can be implemented.

21.4.3.1 Examples from the ISO Designer



21.4.4 Implementation Implications

Since the container is a logical element, it consumes very little processing power itself, but is powerful in controlling the Z-order and grouping. Developers should ensure that Width and Height are set correctly to avoid unwanted clipping.

Further information and examples can be found in the [ISOBUS Wiki - Container](#) by Tobias Tenberg].

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.5 ID 4 – Soft Key Mask – ISO 11783-6 – B.5

The **Soft Key Mask** (softkey mask) with **ID 4** is a special container that defines the assignment of physical or virtual softkeys at the edge of the terminal. It is typically permanently assigned to a data mask or alarm mask.

21.5.1 Attributes and Record Format (Table B.10)

The following table describes the structure of the Soft Key Mask object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	4	3	Object type = Soft Key Mask.
[1]	Background color	Integer	1	0 – 255	4	Background color. The key object has its own background attribute, which this can override.
-	Number of objects to follow	Integer	1	0 – 255	5	Number of contained objects (keys or pointers).
-	Number of macros to follow	Integer	1	0 – 255	6	Number of following macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65534	7 + ...	Object ID of a contained key object or pointer.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(By Object) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.5.1.1 Functionality and Assignment

A softkey mask contains a list of **Key Objects** (ID 5), **Object Pointers** (ID 27), or **External Object Pointers** (ID 43).

- **Order:** The assignment to the physical keys on the terminal is strictly in the order of the list.
- **NULL Pointers:** Pointers to the NULL Object ID reserve a position (the following keys do not move up). Pointers to NULL at the end of the list are not displayed and are not considered for paging.
- **Paging:** If the defined keys exceed the capacity of the terminal, it automatically creates navigation aids (arrow keys) for page turning.
-

21.5.2 Events (Table B.9)

The softkey mask reacts to the following events:

- **On Show:** Triggered when the mask becomes visible. The VT draws all child objects in the defined order.

- **On Hide:** Triggered when the mask is removed.
- **On Change Background Colour:** Reacts to a change in the background color of the bar.
- **On Change Attribute:** Reacts to general attribute changes.

21.5.3 Interaction with Data Masks

Each data mask (ID 1) references a softkey mask (ID 4).

- If the data mask is changed, the VT automatically changes the softkey assignments.
- The key assignments can also be changed at runtime without changing the main mask using the command Change Soft Key Mask.

21.5.4 Importance for Implementation

The design of the softkey masks is crucial for ergonomics. Developers should ensure that important functions (e.g., “Back” or “Home”) are always in the same position. Using NULL pointers can prevent keys from jumping when switching between different screens.

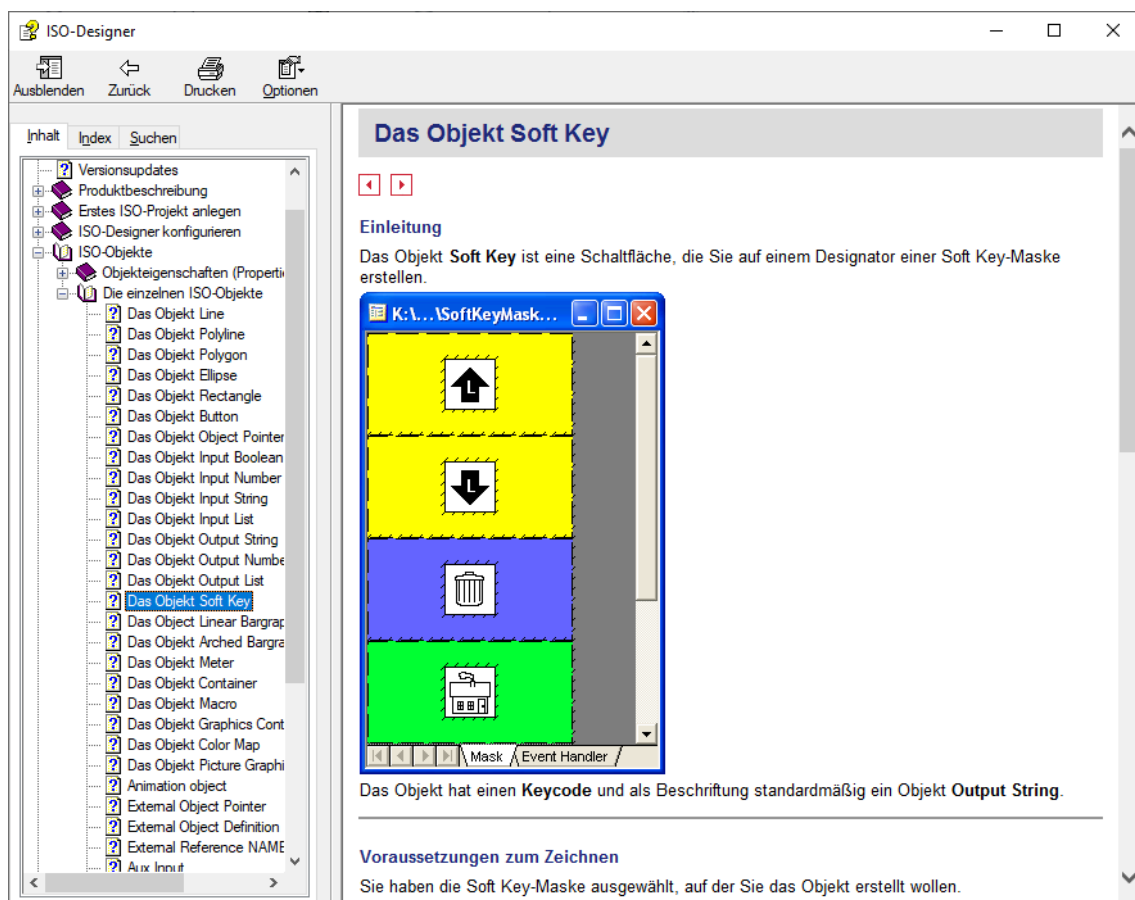
Further information and examples can be found in the [ISOBUS Wiki - Softkey Mask](#) by Tobias Tenberg.]

21.5.5 Podcast

- [ISO 11783-6: Understanding Softkeys and the Virtual Terminal – Your Key to Agricultural Machinery Mechatronics](#)

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.6 ID 5 – Key (Soft Key) – ISO 11783-6 – B.6



21.6.1 Podcast

- [ISO 11783-6: Understanding Softkeys and the Virtual Terminal – Your Key to Agricultural Machinery Mechatronics](#)

The **Key Object** (ID 5) defines the appearance and functional code of a softkey. It is the interactive element within a softkey interface.

21.6.1.1 Attributes and Record Format (Table B.12)

The following table describes the structure of the Key Object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	5	3	Object type = Key.
[1]	Background colour	Integer	1	0 – 255	4	Background color of the key.
[2]	Key code	Integer	1	1 – 255	5	Code sent in the Soft Key Activation message. (0 is reserved for ACK).
-	Number of objects to follow	Integer	1	0 – 255	6	Number of directly included objects (symbols, text).
-	Number of macros to follow	Integer	1	0 – 255	7	Number of subsequent macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65534	8 + ...	Object ID of a contained object.
-	{X Location}	Signed Integer	2	-32768 to +32767	10 + ...	X position relative to the button (pixels).
-	{Y Location}	Signed Integer	2	-32768 to +32767	12 + ...	Y position relative to the button (pixels).
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After objects) Event ID that triggers the macro.

-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.
---	------------	---------	---	---------	------	---------------------------------------

21.6.1.2 Designator and Child Objects

A Key object acts as a container for the graphical content of the key (e.g., icons or text).

- **Coordinates:** The X and Y positions of the child objects are relative to the upper left corner of the softkey designator.
- **Clipping:** Any objects that lie outside the physical area of the softkey are clipped by the VT. Since softkey sizes vary, content should be placed centrally.

21.6.2 Events (Events - Table B.11)

The Key object responds to the following events:

- **On Key Press:** Triggered when the key is pressed. Sends Soft Key Activation.
- **On Key Release:** Triggered when the key is released. Sends Soft Key Activation.
- **On Change Background Colour:** Responds to a change in the key color.
- **On Change Child Location / Position:** Updates the graphical content.
- **On Change Attribute:** Responds to general attribute changes.
- **On Input Field Selection / De-selection:** Triggered when the key receives or loses focus (during navigation via encoder/cursor).

21.6.3 The Key Code

The **Key Code** (AID 2) is the crucial link to the software logic (C code). While Object ID identifies the key in the graphics pool, the application uses Key Code to assign the function. This allows different graphical keys (different IDs) to be assigned the same functional code.

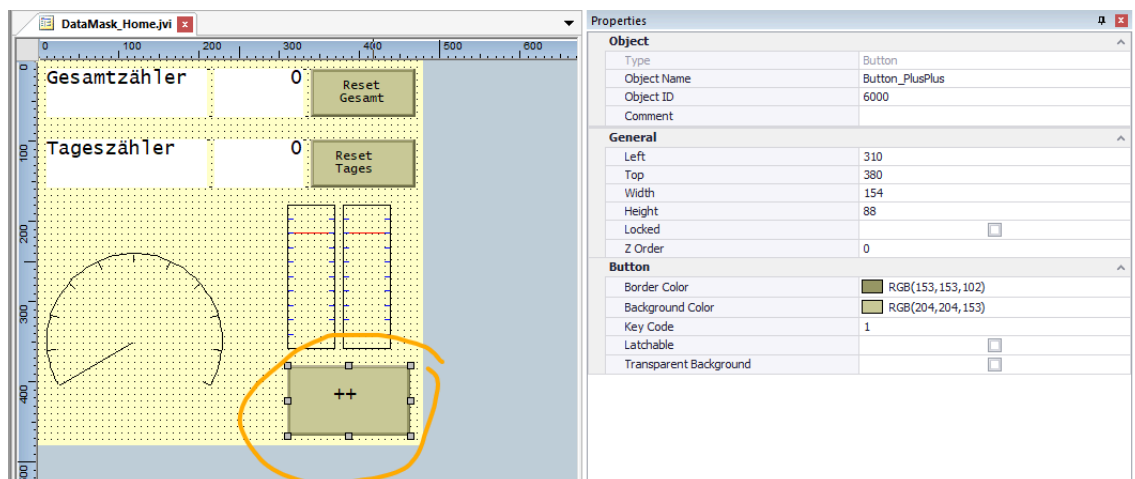
21.6.4 Implementation Implementation Implications

Since the actual size and shape of softkeys can vary from VT to VT (e.g., portrait vs. landscape, touch vs. hardware buttons), it is best practice to choose sufficiently small graphical content (bitmaps) within the key and center it. A key object without child objects appears as an empty area in the chosen background color.

Further information and examples can be found in the [ISOBUS Wiki - Softkey](#) by Tobias Tenberg].

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.7 ID 6 – Button – ISO 11783-6 – B.7



21.7.1 Annex B.7 – Button

Annex B.7 of ISO 11783-6:2018 is dedicated to the detailed definition of the “button” object in the context of the Virtual Terminal (VT). Buttons are basic interactive user interface elements that allow the operator to trigger actions or send commands to the agricultural machine or implement.

21.7.2 Overview of the Button Object

The button object, as defined in B.7, is a graphical element displayed on the VT’s data screen. It is designed so that the operator can interact with it by touch (on touchscreens) or selection using a navigation tool (such as cursor keys). Each button is associated with a specific function or command that is executed when the button is pressed.

21.7.3 Attributes and Record Format (Table B.14)

The following table describes the structure of the Button object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	6	3	Object type = Button.
[1]	Width	Integer	2	0 – 65535	4 – 5	Maximum width of the button area in pixels.
[2]	Height	Integer	2	0 – 65535	6 – 7	Maximum height of the button area in pixels.
[3]	Background color	Integer	1	0 – 255	8	Background color.
[4]	Border color	Integer	1	0 – 255	9	Border color (if not suppressed).
[5]	Key Code	Integer	1	0 – 255	10	Code sent in the Button Activation message.
[6]	Options	Bitmask	1	0 – 63	11	Bit 0: Latchable (0=Buttoning, 1=Hanging)Bit 1: State (only for Latchable: 0=Released, 1=Pressed)Bit 2: Suppress Border (1=No border drawn)Bit 3: Transparent (1=Back-

						ground transparent)Bit 4: Disabled (1=Disabled/Grayed out)Bit 5: No Border (1=Border area omitted, Button Face = Button Area).
-	Number of objects to follow	Integer	1	0 – 255	12	Number of directly contained objects (symbols, text).
-	Number of macros to follow	Integer	1	0 – 255	13	Number of subsequent macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65534	14 + ...	Object ID of a contained object.
-	{X Location}	Signed Integer	2	–32768 to +32767	16 + ...	X position relative to the upper left corner of the button face.
-	{Y Location}	Signed Integer	2	–32768 to +32767	18 + ...	Y position relative to the upper left corner of the button face.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After objects) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.7.3.1 Structure and Appearance

The button consists of three areas:

1. **Button Area:** The entire area defined by Width/Height.
2. **Button Border:** A proprietary 8-pixel border (unless removed by Option Bit 5).
3. **Button Face:** The inner area for content (Button Area minus Border).

21.7.3.2 Container Structure

The button is a container. It can contain other objects, which are displayed in the **Button Face**. Objects that extend beyond this area are clipped.

21.7.4 Events (Events - Table B.13)

The button responds to the following events:

- **On Key Press:** Triggered when the button is pressed. Sends Button Activation.
- **On Key Release:** Triggered when the button is released. Sends Button Activation.
- **On Enable:** When the button is enabled by a command.
- **On Disable:** When the button is disabled.
- **On Input Field Selection:** When the button is focused (navigation).
- **On Input Field De-selection:** When focus is lost.
- **On Change Background Colour:** Responds to a change in background color.
- **On Change Size:** Responds to a change in size (deletes the old area, redraws).
- **On Change Attribute:** Responds to general attribute changes.

21.7.5 Role of the Button Object in the Virtual Terminal

Buttons are crucial for the operator's interaction with the Virtual Terminal and connected agricultural equipment. They enable:

- **Triggering Functions:** Starting or stopping processes, activating implements, switching operating modes.
- **Sending commands:** Controlling actuators, adjusting settings, navigating menus.
- **Confirming entries:** Acknowledging alarms, confirming settings.

The standardized definition of the Button object in Annex B.7 ensures that buttons function and are displayed consistently across different virtual terminals and from different manufacturers. This contributes to the interoperability and user-friendliness of the ISO 11783 system.

21.7.6 Implementation implications

For developers of virtual terminals, attachments, or software components that work with the ISO 11783 standard, a thorough understanding of the specifications in Annex B.7 is essential. They must ensure that their implementation of the Button object complies with the normative requirements to guarantee correct functionality and compatibility within the ISO 11783 network.

21.7.7 Note

For complete and detailed specifications of all attributes and properties of the Button object, including the exact data types, value ranges, and linking mechanisms, please refer to the official document "ISO 11783-6:2018". Annex B.7 provides the normative basis for implementing buttons in the Virtual Terminal.

Button Evaluation:

When the button is pressed, the following messages are sent to the ISOBUS:

- BUTTON_STATE_PRESSED
- at the moment the button was pressed
- BUTTON_STATE_HELD
- if the button was held down for a longer period
- TODO: Reference to ISO
- BUTTON_STATE_RELEASED
- when the button was released
- BUTTON_STATE_ABORTED
- if the button was pressed but then released
- TODO: Describe better.

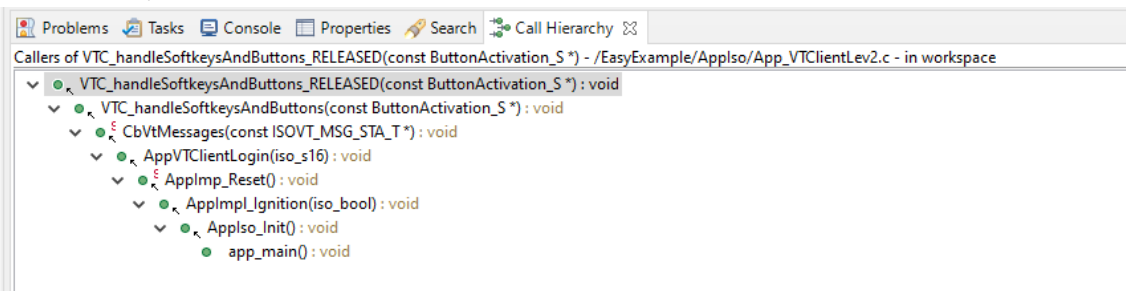
This results in the following:

Normal button presses may only be evaluated as "released" or "held," because evaluating "pressed" would eliminate the possibility of aborting.
 TODO: Reference to international standards, safety, etc.

There are two evaluation methods:

1. Macro (ISO Designer)
2. Via a callback in C code (Eclipse)

Call Hierarchy:



```

iso_u32 Tageszaehler = 0;
iso_u32 Gesamtzaehler = 0;
iso_u32 Hugo = 0;
  
```

```

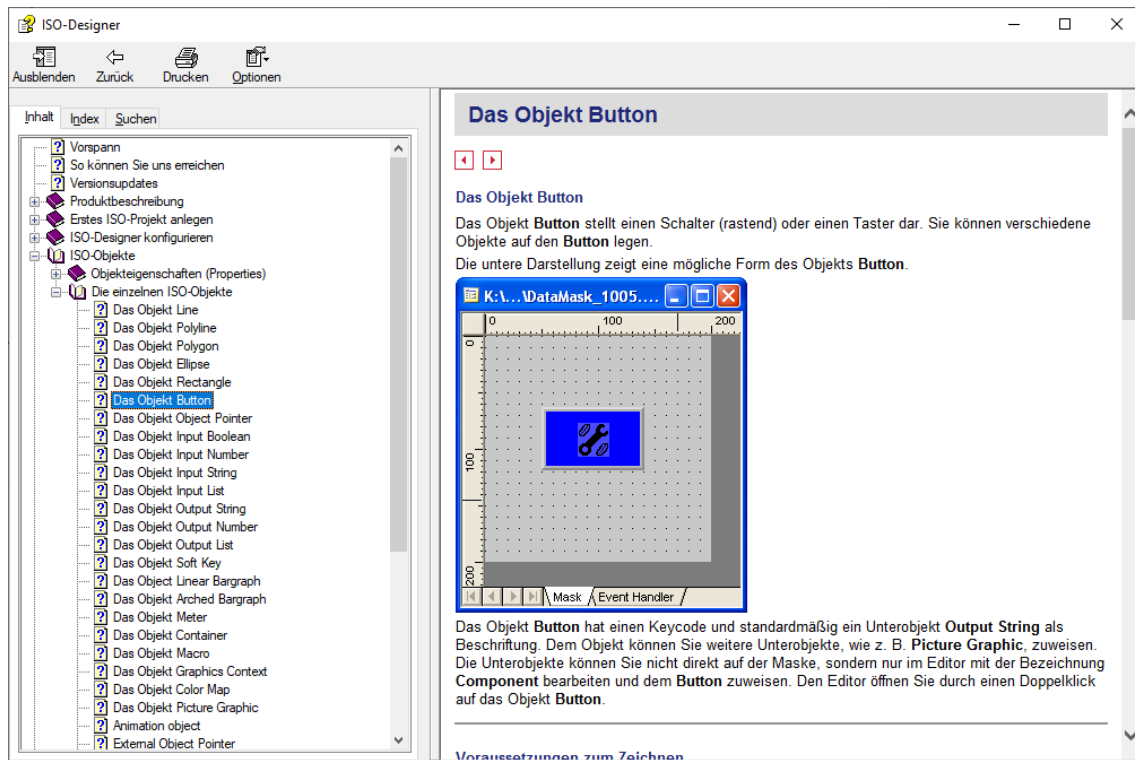
void VTC_handleSoftkeysAndButtons_RELEASED(const struct ButtonActivation_S *pButtonData)
{
    // what button was released
    switch (pButtonData->objectIdOfButtonObject) {

        case SoftKey_PlusPlus:
        case Button_PlusPlus:
            Tageszaehler++;
            Gesamtzaehler++;
            break;

        case SoftKey_Reset_Gesamtzaehler:
        case Button_Reset_Gesamtzaehler:
            Gesamtzaehler = 0;
            break;

        case SoftKey_Reset_Tageszaehler:
        case Button_Reset_Tageszaehler:
            Tageszaehler = 0;
            break;

        default:
            break;
    }
    IsoVtcCmd_NumericValue(pButtonData->u8Instance, NumberVariable_Tageszaehler,
        Tageszaehler);
    IsoVtcCmd_NumericValue(pButtonData->u8Instance, NumberVariable_Gesamtzaehler,
        Gesamtzaehler);
    setU32("CF-A", "Tageszaehler", Tageszaehler);
    setU32("CF-A", "Gesamtzaehler", Gesamtzaehler);
}
  
```



Further information and examples can be found in the [ISOBUS Wiki - Button](#)] by Tobias Tenberg.

21.8 ID 7 – Input Boolean – ISO 11783-6 – B.8.2

The **Input Boolean** object with ID 7 allows the operator to enter a TRUE/FALSE value (e.g., in the form of a checkbox).

21.8.1 Attributes and Record Format (Table B.16)

The following table describes the structure of the Input Boolean object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	7	3	Object type = Input Boolean.
[1]	Background color	Integer	1	0 – 255	4	Background color.
[2]	Width	Integer	2	0 – 65535	5 – 6	Width and height of the square field in pixels.
[3]	Foreground color	Integer	2	0 – 65534	7 – 8	Object ID of a Font Attribute object for the indicator color (only font color is relevant).
[4]	Variable reference	Integer	2	0 – 65534, 65535	9 – 10	Object ID of a Number vari-

						able to store the value. (65535 = value directly in attribute 5).
[5]	Value	Integer	1	0, 1 – 255	11	Value: 0 = FALSE, >0 = TRUE. (Only used if variable reference is NULL).
[6]	Enabled	Integer	1	0 or 1	12	0 = Disabled, 1 = Enabled.
-	Number of macros to follow	Integer	1	0 – 255	13	Number of following macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.8.1.1 Display

The VT visualizes the Boolean value (e.g., as a checkbox). * **Value 0:** Background color is drawn.

- **Value > 0:** Indicator is drawn in the foreground color on the background.

21.8.2 Events (Table B.15)

The Input Boolean object reacts to the following events:

- **On Enable:** When the object is enabled.
- **On Disable:** When the object is disabled.
- **On Input Field Selection:** When the operator focuses/selects the field.
- **On Input Field De-selection:** When focus is lost.
- **On Entry of Value:** When the operator confirms a new value (ENTER). Sends Change Numeric Value.
- **On Change Value:** When the value is changed (e.g., by a variable).
- **On Change Background Colour:** Responds to a color change.
- **On Change Attribute:** Responds to general attribute changes.

21.8.3 Implementation Implementation Implementation

The Boolean input is ideal for simple yes/no options or enabling/disabling machine functions. Since the graphical representation (checkmark style) depends on the VT manufacturer, this object ensures a consistent appearance within the terminal interface.

Further information and examples can be found in the [ISOBUS Wiki - Boolean](#) by Tobias Tenberg].

Note: For detailed specifications on data types and message formats, refer to the official ISO 11783-6:2018.

21.9 ID 8 – Input string – ISO 11783-6 – B.8.3

The **Input String** object with **ID 8** is used for inputting and displaying text strings by the operator.

21.9.1 Attributes and Record Format (Table B.17)

The following table describes the structure of the Input String object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	8	3	Object type = Input String.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the input field in pixels. Clipping occurs outside this range.
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the input field in pixels. Clipping occurs outside this range.
[3]	Background color	Integer	1	0 – 255	8	Background color (only when transparency is disabled).
[4]	Font attributes	Integer	2	0 – 65534	9 – 10	Object ID of a font attribute object (color, size, font).
[5]	Input attributes	Integer	2	0 – 65534, 65535	11 – 12	Object ID of an input attribute object for validation or NULL.
[6]	Options	Bitmask	1	0 – 7	13	Bit 0: Transparent Bit 1: Auto-Wrap Bit 2: Wrap on Hyphen.
[7]	Variable reference	Integer	2	0 – 65534, 65535	14 – 15	Reference to a string variable object. If NULL, the value is stored directly in the “Value” attribute.
[8]	Justification	Integer	1	0 – 15	16	Text alignment: Bits 0-1 (Horizontal): 0=Left, 1=Middle, 2=Right. Bits 2-3 (Vertical): 0=Top, 1=Middle, 2=Bottom.
-	Length	Integer	1	0 – 255	17	Maximum length in bytes. If Variable

						Reference != NULL, this can be 0.
-	Value	String	Length	-	18 ...	Initial value of the string (only if Variable Reference == NULL).
[9]	Enabled	Integer	1	0 or 1	var.	0 = Disabled, 1 = Enabled. Position in the record depends on the length of the Value field.
-	Number of macros to follow	Integer	1	0 – 255	var.	Number of subsequent macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.9.2 Functionality and Options

The Input String object offers flexible text display options:

- **Auto-Wrap:** When enabled (Bit 1), the VT automatically wraps the text if the field width is exceeded.
- **Adjustment:** Horizontal and vertical alignment are controlled via AID 8.
- **Validation:** By linking to a Input Attributes object, input can be restricted to specific character sets.

21.9.3 Events (Table B.15)

The Input String object reacts to the following events:

- **On Enable:** When the object is enabled.
- **On Disable:** When the object is disabled.
- **On Input Field Selection:** When the operator focuses/selects the field.
- **On Input Field De-selection:** When focus is lost.
- **On Entry of Value:** When the operator confirms the text input (ENTER). Sends Change String Value.
- **On Change Value:** When the value is changed (e.g., by a variable).
- **On ESC:** When the operator cancels the input.
- **On Change Background Colour:** Responds to a change in background color.
- **On Change Attribute:** Responds to general attribute changes.

21.9.4 Implementation Implementation Implementation Significance

Input strings are frequently used for names (e.g., field names, customer data) or passwords. Since text input on terminals without a keyboard (only touch or rotary push-button) can be cumbersome, default values or input lists should be preferred when the range of values is limited.

Further information and examples can be found in the ISOBUS Wiki - String (Input)](<https://isobus-studio.com/isobus-wiki/isobus-objectpool-objects/string-input>) by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, refer to the official ISO 11783-6:2018.

21.10 ID 9 – Input number – ISO 11783-6 – B.8.4

The **Input Number** object with **ID 9** is one of the most complex and important input objects. It is used for inputting and displaying numeric values and supports automatic scaling, formatting, and limit checking directly in the terminal.

21.10.1 Attributes and Record Format (Table B.18)

The following table describes the structure of the Input Number object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	9	3	Object type = Input Number.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the input field in pixels.
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the input field in pixels.
[3]	Background color	Integer	1	0 – 255	8	Background color.
[4]	Font attributes	Integer	2	0 – 65534	9 – 10	Object ID of a font attribute object (color, size, font).
[5]	Options	Bitmask	1	0 – 15	11	Bit 0: Transparent Bit 1: Display leading zeros Bit 2: Display zero as blank Bit 3: Truncate (1=Trim, 0=Round).
[6]	Variable reference	Integer	2	0 – 65534, 65535	12 – 13	Reference to a Number Variable object for the raw value. If NULL, the “Value” attribute is used.
[14]	Value	Integer	4	$0 - 2^{32}-1$	14 – 17	Raw value (unsigned 32-bit). Only if Variable Reference == NULL.

[7]	Min value	Integer	4	$0 - 2^{32}-1$	18 – 21	Minimum raw value (unsigned).
[8]	Max value	Integer	4	$0 - 2^{32}-1$	22 – 25	Maximum raw value (unsigned).
[9]	Offset	Signed Integer	4	$-2^{31}-2^{31}-1$	26 – 29	Scaling offset.
[10]	Scale	Float	4	-	30 – 33	Scaling factor.
[11]	Number of decimals	Integer	1	0 – 7	34	Number of decimal places.
[12]	Format	Boolean	1	0 or 1	35	0 = Fixed point, 1 = Exponential.
[13]	Justification	Integer	1	0 – 15	36	Text alignment: Bits 0-1 (Horizontal), Bits 2-3 (Vertical).
[15]	Options 2	Bitmask	1	0 – 3	37	Bit 0: Enabled (0=Disabled, 1=Enabled)Bit 1: Real-time editing (1=Send value immediately).
[-]	Number of macros to follow	Integer	1	0 – 255	38	Number of following macro references.
[-]	Repeat: {Event ID}	Integer	1	0 – 255	39...	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	40...	Macro ID of the macro to be executed.

21.10.2 Scaling Logic

The VT automatically calculates the displayed value using the following formula:

$$\text{Displayed Value} = (\text{Raw Value} + \text{Offset}) \times \text{Scaling Factor}$$

This allows physical values (e.g., 12.5 bar) to be processed as simple integers in memory (e.g., 125), while the VT handles the conversion and decimal representation.

21.10.3 Validation

Limit values are also checked based on the scaled values. The VT only allows the input field to be closed (ENTER) if the new value is within the scaled min/max limits:

$$\text{Scaled Min} \leq \text{Neuer Wert} \leq \text{Scaled Max}$$

21.10.4 Events (Table B.15)

The Input Number object responds to the following events:

- **On Enable / On Disable:** Change of object state.
- **On Input Field Selection / De-selection:** Focus events.
- **On Entry of Value:** When the operator confirms a new value. Sends Change Numeric Value.
- **On Change Value:** When the value is changed by the program.
- **On ESC:** Cancel input.
- **On Change Background Colour:** Change background color.
- **On Change Attribute:** General attribute change.

21.10.5 Real-Time Editing (AID 15, Bit 1)

When this bit is set, the VT sends the current intermediate value to the workgroup with every change (e.g., with each key press on the incremental encoder). This allows the machine to react immediately to changes (e.g., real-time speed control) even before the operator finalizes the input.

21.10.6 Implementation Implications

The Input Number object significantly reduces the workload for the machine control unit (ECU) in terms of formatting and validation. Developers should ensure that the scaling factor and number of decimal places are chosen to prevent rounding errors from distorting the display.

Further information and examples can be found in the [ISOBUS Wiki - Number \(Input\)](#) by Tobias Tenberg.

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018.

21.11 ID 10 – Input list – ISO 11783-6 – B.8.5

The **Input List** object with **ID 10** allows the operator to select an item from a predefined list of objects. It is frequently used for drop-down menus or selection lists.

21.11.1 Attributes and Record Format (Table B.20)

The following table describes the structure of the Input List object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	10	3	Object type = Input List.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the field (when closed).
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the field (when closed).
[3]	Variable reference	Integer	2	0 – 65534, 65535	8 – 9	Reference to a Number Variable object for the index.
[4]	Value	Integer	1	0 – 255	10	Currently selected index (0-254). 255 = no selection.

						(Only if Variable Reference == NULL).
--	--	--	--	--	--	---------------------------------------

- | **Number of list items** | Integer | 1 | 0 – 255 | 11 | Number of objects in the list. | | [5] | **Options** | Bitmask | 1 | 0 – 3 | 12 | Bit 0: Enabled (0=Disabled, 1=Enabled) Bit 1: Real-time editing (1=Send value on change). | | - | **Number of macros to follow** | Integer | 1 | 0 – 255 | 13 | Number of following macro references. | | - | **Repeat: {Object ID}** | Integer | 2 | 0 – 65534, 65535 | 14 + ... | Object ID of a list entry (displayed as an option). | | - | **Repeat: {Event ID}** | Integer | 1 | 0 – 255 | var. | (By Object) Event ID that triggers the macro. | | - | {Macro ID} | Integer | 1 | 0 – 255 | var. | Macro ID of the macro to be executed. |

21.11.2 Functionality and Display

In its normal state, the Input List displays only the currently selected item.

- **Selection:** When the user opens the object, the Input List displays the list of available entries.
- **Index:** The transmitted value is the **zero-based index** of the selected item in the list.
- **Special value 255:** Signals “no selection”.

21.11.3 Events (Events - Table B.19)

The Input List object reacts to the following events:

- **On Enable / On Disable:** Change of the object’s state.
- **On Input Field Selection / De-selection:** Focus events.
- **On Entry of Value:** When the user confirms a selection (ENTER). Sends Change Numeric Value.
- **On Change Value:** When the index is changed by the program.
- **On Entry of New Value:** Triggered when the value changes (often redundant to “On Entry of Value”).
- **On ESC:** Cancels the selection.
- **On Change Attribute:** General attribute change.
- **On Change Size:** Responds to a size change.

21.11.4 Implementation Implications

Input lists are ideally suited to prevent incorrect entries, as the operator can only select from valid options. Since the display (e.g., font size in the expanded list) is controlled by the VT, good readability is ensured on various devices.

Further information and examples can be found in the [ISOBUS Wiki - List \(Input\)](#) by Tobias Tenberg.

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.12 ID 11 – Output string – ISO 11783-6 – B.9.2

The **Output String** object with **ID 11** is used for the purely visual display of text strings on the Virtual Terminal. Unlike the *Input String*, this object does not allow direct editing by the user.

21.12.1 Attributes and Record Format (Table B.22)

The following table describes the structure of the Output String object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	11	3	Object type = Output String.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the text field in pixels. Clipping

						occurs outside this range.
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the text field in pixels. Clipping occurs outside this range.
[3]	Background color	Integer	1	0 – 255	8	Background color (only when transparency is disabled).
[4]	Font attributes	Integer	2	0 – 65534	9 – 10	Object ID of a font attribute object (color, size, font).
[5]	Options	Bitmask	1	0 – 7	11	Bit 0: Transparent Bit 1: Auto-Wrap Bit 2: Wrap on Hyphen
[6]	Variable reference	Integer	2	0 – 65534, 65535	12 – 13	Reference to a string variable object. If NULL, the value is stored directly in the “Value” attribute.
[7]	Justification	Integer	1	0 – 15	14	Text alignment: Bits 0-1 (Horizontal): 0=Left, 1=Middle, 2=Right. Bits 2-3 (Vertical): 0=Top, 1=Middle, 2=Bottom.
-	Length	Integer	2	0 – 65535	15 – 16	Length of the fixed text value in bytes. If variable Ref != NULL, this can be 0.
-	Value	String	Length	-	17...	Static text content (only if variable Reference == NULL).
-	Number of macros to follow	Integer	1	0 – 255	var.	Number of following macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.

-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.
---	------------	---------	---	---------	------	---------------------------------------

21.12.2 Functionality and Special Features

- **Transparency:** When bit 0 is set, the background of the text field is transparent, and the background color of the mask or an underlying object remains visible.
- **Auto-Wrap:** Enables the display of multi-line text within the defined Width and Height.
- **Alignment:** Alignment is pixel-perfect within the frame. This is especially important for vertical centering with different fonts.
- **Clipping:** Text that extends beyond the defined Width or Height is clipped by the VT.

21.12.3 Static Text vs. Dynamic Variable

- **Static Text:** The text is defined directly in the object pool and its properties can only be changed at runtime using the command `Change Attribute` (AID 5 or 7).
- **Dynamic Text:** By linking it to a **string variable** (AID 6), the control unit (ECU) can update the text content at any time using `Change String Value` without having to reload the object itself.

21.12.4 Events (Events - Table B.21)

The output string object reacts to the following events:

- **On Change Value:** Triggered when the displayed value changes (e.g., variable updated). The VT redraws the object.
- **On Refresh:** Triggered when the VT needs to redraw the object (e.g., mask change).
- **On Change Background Colour:** Responds to a color change.
- **On Change Attribute:** Responds to general attribute changes.
- **On Change Size:** Responds to a size change.

21.12.5 Implementation Implementation Implementation Implementation (IM)

Output strings are the primary method for status messages, labels, and unit displays. For texts available in multiple languages, it is recommended to control the text via variables or to maintain a separate mask/pool for each language.

Further information and examples can be found in the [ISOBUS Wiki - String \(Output\)](#) by Tobias Tenberg].

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.13 ID 12 – Output number – ISO 11783-6 – B.9.3

The **Output Number** object with **ID 12** is used to display numeric values. Like the *Input Number* object, it supports automatic scaling and formatting, but is designed purely for display (read-only for the operator).

21.13.1 Attributes and Record Format (Table B.23)

The following table describes the structure of the Output Number object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.

[0]	Type	Integer	1	12	3	Object type = Output Number.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the display field in pixels.
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the display field in pixels.
[3]	Background color	Integer	1	0 – 255	8	Background color.
[4]	Font attributes	Integer	2	0 – 65534	9 – 10	Object ID of a font attribute object.
[5]	Options	Bitmask	1	0 – 15	11	Bit 0: Transparent Bit 1: Display leading zeros Bit 2: Display zero as blank Bit 3: Truncate (1=Trim, 0=Round).
[6]	Variable reference	Integer	2	0 – 65534, 65535	12 – 13	Reference to a Number Variable object for the raw value.
[12]	Value	Integer	4	$0 - 2^{32}-1$	14 – 17	Raw value (unsigned 32-bit). Only if Variable Reference == NULL.
[7]	Offset	Signed Integer	4	$-2^{31}-2^{31}-1$	18 – 21	Scaling offset.
[8]	Scale	Float	4	-	22 – 25	Scaling factor.
[9]	Number of decimals	Integer	1	0 – 7	26	Number of decimal places.
[10]	Format	Boolean	1	0 or 1	27	0 = Fixed point, 1 = Exponential.
[11]	Justification	Integer	1	0 – 15	28	Text alignment: Bits 0-1 (horizontal), Bits 2-3 (vertical).
-	Number of macros to follow	Integer	1	0 – 255	29	Number of following

						macro refer- ences.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.13.2 Scaling Logic

The VT automatically calculates the displayed value using the following formula:

Displayed Value = (Raw Value + Offset) × Scaling Factor

Example: A raw value of 2500 with an offset of 0 and a scaling factor of 0.01 is displayed as **25.00** (with 2 decimal places).

21.13.3 Formatting Options

- **Leading Zeros (Bit 1):** The field is padded with zeros on the left (e.g., “0012”).
- **Zero as Space (Bit 2):** If the scaled value is exactly zero, the field remains completely empty.
- **Round vs. Truncate (Bit 3):** Controls how to handle values with more decimal places than defined in AID 9.

21.13.4 Events

The Output Number object responds to the following events:

- **On Change Value:** Triggered when the displayed value changes (e.g., variable updated or Change Numeric Value command). The VT redraws the object.
- **On Refresh:** Triggered when the VT needs to redraw the object.
- **On Change Background Colour:** Responds to a change in background color.
- **On Change Attribute:** Responds to general attribute changes.
- **On Change Size:** Responds to a change in size.

21.13.5 Implementation Implementation Implementation

Output Numbers are ideal for displaying sensor data (pressure, temperature, speed). Since the ECU only needs to transmit integers (raw values), the CAN bus load is minimized and the formatting complexity is shifted to the terminal. Pixel-precise alignment (AID 11) ensures that the numbers are perfectly aligned within the design frame, even with different fonts.

Further information and examples can be found in the [ISOBUS Wiki - Number \(Output\)](#) by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.14 ID 13 – Output line – ISO 11783-6 – B.10.2

The **Output Line** object with **ID 13** is used to draw a simple line between two points within a virtual rectangle.

21.14.1 Attributes and Record Format (Table B.27)

The following table describes the structure of the Output Line object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.

[0]	Type	Integer	1	13	3	Object type = Output Line.
[1]	Line attributes	Integer	2	0 – 65534	4 – 5	Object ID of a line attribute object (color, width, style).
[2]	Width	Integer	2	0 – 65535	6 – 7	Width of the bounding virtual rectangle.
[3]	Height	Integer	2	0 – 65535	8 – 9	Height of the bounding virtual rectangle.
[4]	Line Direction	Integer	1	0 or 1	10	0 = Top-Left to Bottom-Right 1 = Bottom-Left to Top-Right.
-	Number of macros to follow	Integer	1	0 – 255	11	Number of following macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.14.2 Functionality and Geometry

The line is drawn within an imaginary rectangle defined by the object's position and Width and Height.

- **Line Direction 0:** The line runs diagonally downwards.
- Start point: (X, Y)
- End point: (X + Width - Line Width, Y + Height - Line Width)
- **Line Direction 1:** The line runs diagonally upwards.
- Start point: (X, Y + Height - Line Width)
- End point: (X + Width - Line Width, Y)

21.14.3 Events (Events - Table B.26)

The Output Line object reacts to the following events:

- **On Change End Point:** Triggered when the geometry of the line is changed.
- **On Change Attribute:** Triggered when the line properties (e.g., color) change.
- **On Change Size:** Reacts to a change in size (e.g., by the Change Size command).
- **On Refresh:** Triggered when the VT needs to redraw the object.

21.14.4 Implementation Implications

Lines are frequently used as separators in forms or for the simple graphical representation of relationships. By linking them to variables (via the Line Attributes), lines can change their color at runtime to indicate states (e.g., active/inactive).

Further information and examples can be found in the [ISOBUS Wiki - Line \(Output\)\]](#) by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.15 ID 14 – Output Rectangle – ISO 11783-6 – B.10.3

The **Output Rectangle** object with **ID 14** is used to draw rectangles, which can be displayed as outlines, filled, or in combination.

21.15.1 Attributes and Record Format (Table B.29)

The following table describes the structure of the Output Rectangle object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	14	3	Object type = Output Rectangle.
[1]	Line attributes	Integer	2	0 – 65534	4 – 5	Object ID of a Line Attributes object for the frame.
[2]	Width	Integer	2	0 – 65535	6 – 7	Width of the rectangle in pixels.
[3]	Height	Integer	2	0 – 65535	8 – 9	Height of the rectangle in pixels.
[4]	Line suppression	Bitmask	1	0 – 15	10	Suppression of sides: Bit 0=Top, Bit 1=Right, Bit 2=Bottom, Bit 3=Left. (1 = do not draw).
[5]	Fill attributes	Integer	2	0 – 65534, 65535	11 – 12	Object ID of a Fill Attribute object (for filling) or NULL for no fill.
-	Number of macros to follow	Integer	1	0 – 255	13	Number of following macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.15.2 Appearance and Properties

The rectangle combines line and fill properties:

• **Outline:** If AID 1 is linked, a border is drawn according to the Line Attributes.

Repeat: * Fill: If AID 5 is linked, the interior of the rectangle is filled according to the Fill attributes.

- **Line Suppression (AID 4):** Allows you to selectively omit drawing individual edges of the rectangle. This is useful for table structures or open frames.
- **Clipping:** The rectangle defines its own graphic boundaries using Width and Height.

21.15.3 Geometric Calculation

The corners of the rectangle are determined by the object's starting position (StartX, StartY):

- **Top-Left Corner:** (StartX, StartY)
- **Bottom-Right Corner:** (StartX + Width - 1, StartY + Height - 1) The line width must be considered during planning, as it can grow inwards or outwards depending on the VT implementation.

21.15.4 Events (Table B.28)

The output Rectangle object responds to the following events:

- **On Change Size:** Triggered when the size of the rectangle changes at runtime.
- **On Change Attribute:** Triggered when line or fill attributes (e.g., colors) change.
- **On Refresh:** Triggered when the VT needs to redraw the object.

21.15.5 Implementation Implications

Rectangles are the most frequently used graphic primitives. They serve as backgrounds for text fields, as borders for groups, or for creating bars (e.g., by dynamically changing the Width or Height via an ECU command). In combination with transparent backgrounds, complex layouts can be implemented.

Further information and examples can be found in the [ISOBUS Wiki - Rectangle](#) by Tobias Tenberg].

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.16 ID 15 – Output Ellipse – ISO 11783-6 – B.10.4

The **Output Ellipse** object with **ID 15** is used to draw circles, ellipses, arcs, segments, and sectors (pie chart segments).

21.16.1 Attributes and Record Format (Table B.31)

The following table describes the structure of the Output Ellipse object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	15	3	Object type = Ellipse.
[1]	Line attributes	Integer	2	0 – 65534	4 – 5	Object ID of a Line Attributes object (for the outline).
[2]	Width	Integer	2	0 – 65535	6 – 7	Width of the enclosing virtual rectangle.
[3]	Height	Integer	2	0 – 65535	8 – 9	Height of the enclosing virtual rectangle.
[4]	Ellipse type	Integer	1	0 – 3	10	0=Closed Ellipse, 1=Open Ellipse (arc), 2=Segment

						(chord), 3=Sector (pie slice).
[5]	Start angle	Integer	1	0 – 180	11	Start angle / 2 (in degrees, counterclockwise from the positive X-axis).
[6]	End angle	Integer	1	0 – 180	12	End angle / 2 (in degrees, counterclockwise from the positive X-axis).
[7]	Fill attributes	Integer	2	0 – 65534, 65535	13 – 14	Object ID of a Fill Attributes object (for filling) or NULL for no filling.
-	Number of macros to follow	Integer	1	0 – 255	15	Number of following macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.16.2 Ellipse Types and Geometry

The ellipse is fitted into a virtual rectangle (Width x Height).

- **Closed Ellipse (0):** A complete ellipse or a circle (if Width = Height).
- **Open Ellipse (1):** Only the arc between the start and end angles is drawn.
- **Segment (2):** A segment of a circle (the chord between the angle points is closed).
- **Sector (3):** A sector of a circle (the angle points are connected to the center point, ideal for pie charts).

21.16.3 Angle Calculation (Important!)

The angle values in AID 5 and 6 are transferred **halved** (range 0-180 corresponds to 0-360°).

- **90° (Top):** Value 45
- **180° (Left):** Value 90
- **270° (Bottom):** Value 135

Special note for scaled ellipses: If the ellipse is not a circle (Width != Height), the VT must ensure that the angles are drawn mathematically correctly and not just a scaled circular arc (see ISO standard Figure B.8).

21.16.4 Events (Events - Table B.30)

The Output Ellipse object reacts to the following events:

- **On Change Size:** Triggered when the size of the rectangle changes at runtime.
- **On Change Attribute:** Triggered when line or fill attributes (e.g., colors) change.
- **On Refresh:** Triggered when the VT needs to redraw the object.

21.16.5 Implementation Implications

Ellipses and sectors are essential for creating analog pointer instruments (meters) or progress indicators. Dynamically changing the End angle via ECU command allows for the creation of filled circular arcs that intuitively visualize states.

Further information and examples can be found in the [ISOBUS Wiki - Ellipse](#) by Tobias Tenberg].

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.17 ID 16 – Output polygon – ISO 11783-6 – B.10.5

The **Output Polygon** object with **ID 16** allows the drawing of complex, polygonal shapes by defining a list of vertices.

21.17.1 Attributes and Record Format (Table B.33)

The following table describes the structure of the Output Polygon object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	16	3	Object type = Polygon.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the enclosing virtual rectangle.
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the enclosing virtual rectangle.
[3]	Line attributes	Integer	2	0 – 65534	8 – 9	Object ID of a Line Attributes object (for the outline).
[4]	Fill attributes	Integer	2	0 – 65534, 65535	10 – 11	Object ID of a Fill Attributes object (fill) or NULL.
[5]	Polygon type	Integer	1	0 – 3	12	0=Convex, 1=Non-convex, 2=Complex, 3=Open.
-	Number of points	Integer	1	3 – 255	13	Number of vertices (minimum 3).
-	Number of macros to follow	Integer	1	0 – 255	14	Number of following macro references.
-	Repeat: {Point X}	Integer	2	0 – 65535	15 + ...	X-position of the point relative to the top left corner of the object.

-	{Point Y}	Integer	2	0 – 65535	17 + ...	Y-position of the point relative to the top left corner of the object.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(By Points) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.17.2 Polygon Types and Fill Rules

The polygon type (AID 5) informs the VT about the complexity of the shape, which affects drawing efficiency:

- **Convex (0):** Simple shapes (e.g., triangle, hexagon) where each horizontal line intersects the edges only twice.
- **Non-Convex (1):** Indentations are possible, but edges do not intersect.
- **Complex (2):** Edges may intersect (e.g., a star). Here, the **even-odd rule** is applied for filling.
- **Open (3):** The points are only connected by lines; the polygon is not closed and not filled.

21.17.3 Geometry and Points

- **Relative Position:** All points (Point X, Point Y) are referenced to the upper left corner of the polygon object.
- **Automatic Closing:** If the type is not “Open” and the last point is not identical to the first, the VT automatically closes the polygon with a line from the last to the first point.
- **Number of Points:** Up to 255 points can be defined.

21.17.4 Events (Table B.32)

The output polygon object responds to the following events:

- **On Refresh:** Triggered by the Change Polygon Point or Change Polygon Scale command.
- **On Change Attribute:** Triggered when line or fill attributes change.
- **On Change Size:** Responds to a change in size (of the enclosing rectangle).

21.17.5 Implementation Implementation Implementation

Polygons are very powerful for representing irregular shapes, such as tank contents in asymmetrical containers or for visualizing field outlines. Due to the computational load of complex polygons, ECU developers should, where possible, prefer convex polygon types if the shape allows it.

Further information and examples can be found in the ISOBUS Wiki - Polygon](<https://isobus-studio.com/isobus-wiki/isobus-objectpool-objects/polygon>) by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.18 ID 17 – Output meter – ISO 11783-6 – B.11.2

The **Output Meter** object with ID 17 is a circular display (pointer instrument). It visualizes a numerical value by the position of a needle on a circular arc.

21.18.1 Attributes and Record Format (Table B.35)

The following table describes the structure of the Output Meter object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	17	3	Object type = Output Meter.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width and height of the enclosing square.
[2]	Needle color	Integer	1	0 – 255	6	Needle color.
[3]	Border color	Integer	1	0 – 255	7	Border color (when drawn).
[4]	Arc and tick color	Integer	1	0 – 255	8	Arc and tick color.
[5]	Options	Bitmask	1	0 – 15	9	Bit 0: Draw Arc Bit 1: Draw Border Bit 2: Draw Ticks Bit 3: Deflection Direction (0 = min->max counterclockwise, 1 = min->max clockwise).
[6]	Number of ticks	Integer	1	0 – 255	10	Number of scale markings.
[7]	Start angle	Integer	1	0 – 180	11	Start angle / 2 (in degrees, counterclockwise from the positive x-axis).
[8]	End angle	Integer	1	0 – 180	12	End angle / 2 (in degrees, counterclockwise from the positive x-axis).
[9]	Min value	Integer	2	0 – 65535	13 – 14	Value at the start angle.
[10]	Max value	Integer	2	0 – 65535	15 – 16	Value at the end angle.
[11]	Variable reference	Integer	2	0 – 65534, 65535	17 – 18	Reference to a Number Variable object.
[12]	Value	Integer	2	0 – 65535	19 – 20	Current value. Only if Variable Reference == NULL.

-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.					
[0]	Type	Integer	1	18	3	Object type = Linear Bar Graph.					
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the bounding rectangle in pixels.					
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the bounding rectangle in pixels.					
[3]	Colour	Integer	1	0 – 255	8	Color of the bar fill and border.					
[4]	Target line color	Integer	1	0 – 255	9	Color of the target value line (when drawn)					
[5]	Options	Bit-mask	1	0 – 63	10	Bit 0: Draw Border	Bit 1: Draw Target Line	Bit 2: Draw Ticks	Bit 3: Type (0=Filled, 1=Line)	Bit 4: Axis Orientation (0=Vertical, 1=Horizontal)	Bit 5: Direction (0=Negative/Descending, 1=Positive/Ascending).

[6]	Number of ticks	Integer	1	0 – 255	11	Number of ticks.					
[7]	Min value	Integer	2	0 – 65535	12 – 13	Minimum value.					
[8]	Max value	Integer	2	0 – 65535	14 – 15	Maximum value.					
[9]	Variable reference	Integer	2	0 – 65535	16 – 17	Reference to a Number variable for the current value.					
[12]	Value	Integer	2	0 – 65535	18 – 19	Current value. Only if Variable Reference == NULL.					
[10]	Target value var.	Integer	2	0 – 65535	20 – 21	Reference to a Number variable for the target value.					
[11]	Target value	Integer	2	0 – 65535	22 – 23	Current target value. Only if Target value variable ref == NULL.					

-	Number of macros to follow	Integer	1	0 – 255	24	Number of following macro references.					
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.					
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.					

21.19.2 Functionality and Options

The bar graph is fitted into a rectangle and is transparent by default, so background graphics remain visible.

- **Alignment (AID 5, Bits 4 & 5):** Allows horizontal (left-to-right) or vertical (bottom-to-top) bars.
- **Display Type (Bit 3):** In addition to the classic fill bar, the object can also function as a “single pointer,” where only a line is drawn at the current position.
- **Target Line:** An additional marker (e.g., a red line) that indicates a target value or a warning range.

21.19.3 Scaling

The bar is calculated proportionally to the current value between `Value` and `Min` value and `Max` value. If the value is outside this range, the bar is either completely empty or completely filled.

21.19.4 Events (Events - Table B.36)

The Output Linear Bar Graph object reacts to the following events:

- **On Change Value:** Triggered when `Value` or `Target` value changes. The VT updates the graph.
- **On Change Attribute:** Triggered when attributes change.
- **On Change Size:** Responds to size changes.
- **On Refresh:** Triggered when the VT needs to redraw the object.

21.19.5 Implementation Implications

Bar charts are ideal for level indicators (fuel, seeds), temperature displays, or load indicators. The `Target` line option allows the operator to immediately visualize whether they are within the optimal operating range. Combining this with scale ticks improves readability.

21.19.6 Podcast

- [ISOBUS bar chart: The Output Linear Bar Graph object of ISO 11783-6 decoded](#)

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.20 ID 19 – Output Arched Bar Graph – ISO 11783-6 – B.11.4

The **Output Arched Bar Graph** object with **ID 19** is an arched bar graph. It combines the properties of a linear bar chart with the circular geometry of a meter object.

21.20.1 Attributes and Record Format (Table B.39)

The following table describes the structure of the Output Arched Bar Graph object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	19	3	Object type = Arched Bar Graph.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the bounding rectangle in pixels.
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the bounding rectangle in pixels.
[3]	Colour	Integer	1	0 – 255	8	Color of the bar fill and border.
[4]	Target line color	Integer	1	0 – 255	9	Color of the target value line (if drawn).
[5]	Options	Bitmask	1	0 – 31	10	Bit 0: Draw Border Bit 1: Draw Target Line Bit 3: Type (0=Filled, 1=Line/Pointer) Bit 4: Deflection (0=Counter-clockwise, 1=Clockwise).
[6]	Start angle	Integer	1	0 – 180	11	Start angle / 2 (in degrees, counter-clockwise from the positive X-axis).
[7]	End angle	Integer	1	0 – 180	12	End angle / 2 (in degrees, counter-clockwise from the positive X-axis).
[8]	Bar graph width	Integer	2	0 – 65535	13 – 14	Arc thickness in pixels.
[9]	Min value	Integer	2	0 – 65535	15 – 16	Minimum value.

[10]	Max value	Integer	2	0 – 65535	17 – 18	Maximum value.
[11]	Variable reference	Integer	2	0 – 65534, 65535	19 – 20	Reference to a Number variable for the current value.
[14]	Value	Integer	2	0 – 65535	21 – 22	Current raw value. Only if Variable Reference == NULL.
[12]	Target value var.	Integer	2	0 – 65534, 65535	23 – 24	Reference to a Number variable for the target value.
[13]	Target value	Integer	2	0 – 65535	25 – 26	Current target value. Only if Target value variable ref == NULL.
-	Number of macros to follow	Integer	1	0 – 255	27	Number of subsequent macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.20.2 Functionality and Geometry

The arc-shaped bar is drawn based on a virtual ellipse object within the enclosing rectangle.

- **Bar Thickness (AID 8):** Defines the width of the arc itself.
- **Angle Logic:** Angle values are transmitted in halves (as with the meter object), e.g., 45 for 90°.
- **Deflection (AID 5, Bit 4):** Controls whether the bar “grows” clockwise or counterclockwise.
- **Transparency:** The object is transparent, allowing it to be overlaid with background images.
- **Transparency:** The object is transparent, enabling it to be overlaid with background images.

21.20.3 Events (Table B.38)

The output Arched Bar Graph object responds to the following events:

- **On Change Value:** Triggered when the Value or Target value changes. The VT updates the graph.
- **On Change Attribute:** Triggered when attributes change.
- **On Change Size:** Responds to a size change.
- **On Refresh:** Triggered when the VT needs to redraw the object.

21.20.4 Implementation Implications

Arched bar graphs are ideal for modern cockpit designs where multiple scales are nested within each other to save space (e.g., temperature and fuel). The Target line allows the operator to visualize a target range, while the arc shape enables intuitive level measurement.

Further information and examples can be found in the [ISOBUS Wiki - Arched Bar Graph](#) by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.21 ID 20 – Picture graphic – ISO 11783-6 – B.12.2

The **Picture Graphic** object with **ID 20** is used to display raster graphics (bitmaps) on the Virtual Terminal. It allows the integration of logos, icons, and complex visual elements.

21.21.1 Attributes and Record Format (Table B.41)

The following table describes the structure of the Picture Graphic object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	20	3	Object type = Picture Graphic.
[1]	Width	Integer	2	0 – 65535	4 – 5	Target width in pixels (height is scaled proportionally).
[4]	Actual width	Integer	2	0 – 65535	6 – 7	Actual width of the raw data.
[5]	Actual height	Integer	2	0 – 65535	8 – 9	Actual height of the raw data.
[6]	Format	Integer	1	0 – 2	10	0=Monochrome (1 bit), 1=16 colors (4 bits), 2=256 colors (8 bits).
[2]	Options	Bitmask	1	0 – 7	11	Bit 0: Transparency (0=Opaque, 1=Transparent) Bit 1: Blinking Bit 2: Data format (0=Raw, 1=Run-Length Encoded).
[3]	Transparency color	Integer	1	0 – 255	12	Color index to be displayed transparently.
-	Number of bytes in raw data	Integer	4	0 – 2 ³² -1	13 – 16	Size of image data.
-	Number of macros to follow	Integer	1	0 – 255	17	Number of following macro references.
-	Repeat: {raw data}	Integer	1	0 – 255	18 ...	Image data (bytes).
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After image data) Event ID

						that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.21.2 Functionality and Display

The Picture Graphic object stores pixel graphics in binary form within the object pool file (.IOP).

21.21.2.1 Scaling and Formats

- **Aspect Ratio:** The VT scales the graphic based on the target width (Width). To avoid distortion, the VT automatically calculates the target height from the ratio of Actual width to Actual height.
- **Color Depth:** 1-bit (monochrome), 4-bit (16 colors), and 8-bit (256 colors) are supported.

21.21.2.2 Transparency and Effects

- **Transparency (Bit 0):** When enabled, the color defined in AID 3 is not drawn; instead, the background shows through.
- **Flashing (Bit 1):** The image can blink (frequency and style depend on the VT).
- **RLE Compression (Bit 2):** Run-Length Encoding can save memory for simple graphics (many areas of the same color).

21.21.3 Events (Events - Table B.40)

The Picture Graphic object reacts to the following events:

- **On Refresh:** Triggered when options change (e.g., transparency, blinking) or when the mask is refreshed.
- **On Change Attribute:** Reacts to general attribute changes.

21.21.4 Implementation Importance

Picture graphics are essential for a modern HMI.

- **Avoid Waste:** Since bitmaps occupy a lot of memory in the VT, they should be kept as small as possible.
- **Reuse:** An image can be defined once in the pool and referenced by many objects (e.g., multiple buttons).
- **Icons:** They are frequently used as “designators” for softkeys (ID 5) or as symbols in containers (ID 3).

Further information and examples can be found in the [ISOBUS Wiki - Picture Graphic object](#) by Tobias Tenberg].

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.22 ID 21 – Number variable – ISO 11783-6 – B.13.2

The **Number Variable** object with **ID 21** is a pure data object. It stores a numeric value that can be referenced by other display or input objects.

21.22.1 Attributes and Record Format (Table B.43)

The following table describes the structure of the Number Variable object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	21	3	Object Type = Number Variable.

[1]	Value	Integer	4	$0 - 2^{32}-1$	4 – 7	32-bit unsigned integer value.
-----	--------------	---------	---	----------------	-------	--------------------------------

21.22.2 Functionality and Referencing

Variables are not visible objects. They are never directly inserted into a mask or container as a “child” object, but rather serve as a data source for other objects:

- **Referencing:** Objects such as *Input Number* (ID 9), *Output Number* (ID 12), or *Output Meter* (ID 17) reference the ID of a Number Variable via their attribute `Variable` reference.
- **Central Data Storage:** Multiple display objects can reference the same variable. If the value of the variable is changed, the VT automatically updates all affected displays.

21.22.3 Events (Table B.42)

The Number Variable object reacts to the following events:

- **On Change Value:** Triggered when the value changes (by a command or operator input). The VT redraws all objects that reference this variable.

21.22.4 Implementation Implications

Number variables are the backbone of communication between the machine and the terminal.

- **Efficiency:** Instead of updating each display object individually, the ECU only changes the value of the central variable.
- **Consistency:** Using variables ensures that the same current value is always displayed in different parts of the user interface (e.g., main screen and settings menu).

Further information and examples can be found in the [ISOBUS Wiki - Number Variable](#) by Tobias Tenberg.

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.23 ID 22 – String variable – ISO 11783-6 – B.13.3

The **String Variable** object with ID 22 is used to store text strings that can be referenced by display or input objects.

21.23.1 Attributes and Record Format (Table B.44)

The following table describes the structure of the String Variable object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	22	3	Object type = String Variable.
-	Length	Integer	2	0 – 65535	4 – 5	Maximum fixed length of the string in bytes.
-	Value	String	Length	-	6 ...	String consisting of characters. Must be padded with spaces to reach the specified length.

21.23.2 Functionality and Special Features

Like the *Number Variable*, the *String Variable* is a pure data object without its own visual representation.

- **Fixed Length:** The length of the variable is set in the pool during creation and cannot be increased at runtime.
- **Padding:** If a string shorter than the defined length is stored, the VT automatically pads the remainder with spaces.
- **Data Types:** Supports both 8-bit characters (default) and WideStrings (for special characters). The ECU can switch the type between these formats at runtime.
- **Fixed Length:**

21.23.3 Referencing and Updating

- **Referencing:** Objects such as *Input String* (ID 8) or *Output String* (ID 11) reference the ID of a string variable via their attribute *Variable* reference.
- **Automatic Redraw:** As soon as the ECU changes the value of the variable via the *Change String Value* command, the VT automatically updates all visible objects that use this variable.

21.23.4 Events (Table B.42)

The string variable object reacts to the following events:

- **On Change Value:** Triggered when the value changes (by the *Change String Value* command or operator input). The VT redraws all objects that reference this variable.

21.23.5 Implementation Significance

String variables are essential for dynamic text such as plain-text error messages, work order names, or driver names. Since text input and modification via the CAN bus (ISOBUS) are resource-intensive, string variables should be defined as concisely as possible.

Further information and examples can be found in the [ISOBUS Wiki - String Variable](#) by Tobias Tenberg.

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.24 ID 23 – Font attributes – ISO 11783-6 – B.14.2

The **Font Attributes** object with **ID 23** defines the appearance of text (font, size, color, style). It is a central attribute object referenced by all text-based display and input objects.

21.24.1 Attribute and Record Format (Table B.46)

The following table describes the structure of the Font Attributes object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	23	3	Object type = Font Attributes.
[1]	Font colour	Integer	1	0 – 255	4	Text color.
[2]	Font size	Integer	1	0 – 255	5	Font size (see below).
[3]	Font type	Integer	1	0 – 255	6	Character set (see ISO Table K.1).

[4]	Font style	Bitmask	1	0 – 255	7	Bit 0: BoldBit 1: Crossed OutBit 2: UnderlinedBit 3: ItalicBit 4: InvertedBit 5: Flashing (Inverted/Style)Bit 6: Flashing (Hidden/Style)Bit 7: Proportional (0=Fixfont, 1=Prop.).
-	Number of macros to follow	Integers	1	0 – 255	8	Number of following macro references.
-	Repeat: {Event ID}	Integers	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.24.2 Font Sizes and Render Modes

The interpretation of AID 2 depends heavily on bit 7 in the Font style options:

21.24.2.1 Non-proportional fonts (bit 7 = 0)

Here, predefined grid sizes are used (width x height in pixels):

- **0:** 6x8
- **1:** 8x8
- **2:** 8x12
- ...
- **14:** 128x192

21.24.2.2 Proportional fonts (bit 7 = 1)

In this mode, the value in AID 2 directly represents the **font height in pixels**. The width of individual characters varies.

21.24.3 Events (Table B.45)

The Font Attributes object responds to the following events:

- **On Change Font Attributes:** Triggered by the command Change Font Attributes. The VT updates all objects that use these attributes.
- **On Change Attribute:** Responds to general attribute changes.

21.24.4 Implementation Implications

Font attributes allow for a consistent design. Instead of defining color and size individually for each text object, all objects reference a common attribute object. If this single object is changed (e.g., from white to yellow font), the entire HMI appearance changes immediately.

Further information and examples can be found in the [ISOBUS Wiki - Font Attributes](#) by Tobias Tenberg].

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.25 ID 24 – Line attributes – ISO 11783-6 – B.14.3

The **Line Attributes** object with **ID 24** defines the graphic properties of lines and outlines (color, width, style). It is referenced by all geometric objects such as *Line* (ID 13), *Rectangle* (ID 14), *Ellipse* (ID 15), and *Polygon* (ID 16).

21.25.1 Attribute and Record Format (Table B.48)

The following table describes the structure of the Line Attributes object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	24	3	Object type = Line Attributes.
[1]	Line colour	Integer	1	0 – 255	4	Line color.
[2]	Line width	Integer	1	0 – 255	5	Line thickness in pixels. 0 = do not draw.
[3]	Line art	Bitmask	2	0 – 65535	6 – 7	Bit pattern for line style (1 = draw, 0 = background).
[-]	Number of macros to follow	Integer	1	0 – 255	8	Number of following macro references.
[-] Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.	
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.25.2 Line Thickness and Appearance

The VT uses a square “brush” of size Line width x Line width to draw the line.

- **Width = 0:** The line is not drawn.
- **Width > 1:** The line appears thicker.

21.25.3 Line Style (Line Art - AID 3)

A 16-bit mask defines whether a line appears solid, dashed, or dotted:

- Each set bit (1) represents a drawn brushstroke.
- Each unset bit (0) represents a gap (background shows through).
- **Example 0xFFFF:** Solid line (all bits 1).
- **Example 0xCCCC (11001100...):** Dashed line.
- **Special Feature:** The length of a line scales with the Line width.

21.25.4 Events (Table B.47)

The Line Attributes object reacts to the following events:

- **On Change Line Attributes:** Triggered by the Change Line Attributes command. The VT updates all objects that use this attribute.
- **On Change Attribute:** Reacts to general attribute changes.

21.25.5 Implementation Implementation Implementation

Line attributes enable efficient control of the graphical display. By changing a single attribute object, for example, all borders in a mask can be switched simultaneously from “Thin/Black” to “Thick/Red” to visualize an alarm condition.

Further information and examples can be found in the [ISOBUS Wiki - Line Attribute](#)] by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.26 ID 25 – Fill attributes – ISO 11783-6 – B.14.4

The **Fill Attributes** object with **ID 25** defines how closed geometric shapes (rectangles, ellipses, polygons) are filled.

21.26.1 Attribute and Record Format (Table B.50)

The following table describes the structure of the Fill Attributes object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	25	3	Object type = Fill Attributes.
[1]	Fill type	Integer	1	0 – 3	4	0=No fill, 1=Line color, 2=Fill color, 3=Pattern.
[2]	Fill color	Integer	1	0 – 255	5	Fill color (only relevant for type 2).
[3]	Fill pattern	Integer	2	0 – 65534, 65535	6 – 7	Object ID of a Picture Graphic object for pattern fill (only for type 3).
[-]	Number of macros to follow	Integer	1	0 – 255	8	Number of following macro references.
[-]	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.26.2 Fill Types and Logic

AID 1 controls which source is used for the fill:

- **No Fill (0):** The fill remains transparent; only the outline (if defined) is drawn.
- **Line Colour (1):** The fill is the same color as the border (from the Line Attributes).
- **Specified Colour (2):** The color defined in AID 2 is used.
- **Pattern (3):** The fill is a repeating graphic (tiling).

21.26.3 Using Fill Patterns (Important!)

When a pattern (AID 3) is used, strict rules apply to the referenced graphic:

- **Alignment:** For monochrome graphics, the width must be divisible by 8. For 16-color graphics, it must be divisible by 2.
- **Order:** For dynamic changes, Fill pattern must be set first, followed by Fill type, to avoid errors in the VT.

21.26.4 Events (Table B.49)

The Fill Attributes object responds to the following events:

- **On Change Fill Attributes:** Triggered by the command Change Fill Attributes. The VT updates all objects that use this attribute.
- **On Change Attribute:** Responds to general attribute changes.

21.26.5 Implementation Implementation Importance

Fill attributes are essential for user interface design. They allow you to highlight areas (e.g., yellow fill for a warning zone in a bar chart) or create textured backgrounds. By changing the fill type at runtime, states (e.g., “tank empty” -> flashing red fill) can be visualized very effectively.

Further information and examples can be found in the [ISOBUS Wiki - Fill Attribute](#) by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.27 ID 26 – Input attributes – ISO 11783-6 – B.14.5

The **Input Attributes** object with **ID 26** is used to validate text input. It defines which characters an operator is allowed to enter into a linked *Input String* object.

21.27.1 Attribute and Record Format (Table B.52)

The following table describes the structure of the Input Attributes object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	26	3	Object type = Input Attributes.
[1]	Validation type	Integer	1	0 – 1	4	0 = Allowed characters (list), 1 = Forbidden characters (list).
-	Length	Integer	1	0 – 255	5	Length of the validation string in bytes.
-	Validation string	String	Length	-	6 ...	List of characters (8-bit string).

-	Number of macros to follow	Integer	1	0 – 255	var.	Number of following macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.27.2 Functionality and Validation

This object acts as a filter for keyboard input on the VT:

- **Referencing:** An *Input String* object (ID 8) references this object.
- **Filter Logic:** If *Validation type* is set to 0, the VT only allows the characters contained in *Validation string*. If it is set to 1, all characters except those listed are accepted.
- **Limitation:** This object only supports **8-bit strings**. If the linked input field uses a *WideString*, no validation takes place.

21.27.3 Events (Table B.51)

The Input Attributes object reacts to the following events:

- **On Change Value:** Triggered by the command *Change String Value*. The VT updates the validation string.

21.27.4 Implementation Implications

Input attributes are an important tool for preventing user errors.

- **Example Numeric:** A validation string “0123456789.” restricts a text field to purely numeric characters.
- **Example Special Characters:** Prohibits characters such as “;” or “”, which could cause problems in databases or file systems.

21.27.4.1 Note: Extended Input Attributes (ID 38)

For validating *WideStrings* (Unicode), the *Extended Input Attributes* object must be used, which allows the definition of entire code planes.

Note: For detailed specifications regarding data types and message formats, please refer to the official ISO 11783-6:2018 standard.

21.28 ID 27 – Object pointer – ISO 11783-6 – B.15

The **Object Pointer** object with ID 27 serves as a dynamic placeholder. It allows the referenced object to be replaced at runtime at a predefined location within a mask or container.

21.28.1 Attributes and Record Format (Table B.55)

The following table describes the structure of the Object Pointer object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	27	3	Object type = Object Pointer.

[1]	Value	Integer	2	0 – 65534, 65535	4 – 5	Object ID of the referenced object or NULL (65535).
-----	--------------	---------	---	------------------	-------	---

21.28.2 Functionality and Application

An object pointer is integrated into a mask like a normal child object. However, instead of drawing anything itself, it “redirects” the display to the object whose ID is stored in AID 1.

- **Dynamic Exchange:** The ECU can change Value (AID 1) at any time using the Change Numeric Value command. The VT then hides the old object and displays the new one in the same position.
- **Placeholder Function:** It is ideal for status icons (e.g., changing symbols for different machine states) without having to layer multiple objects and hide them individually.
- **NULL Pointer:** If the value is set to 65535, nothing is drawn at this location.

21.28.3 Events (Table B.54)

The Object Pointer object reacts to the following events:

- **On Change Value:** Triggered by the command Change Numeric Value. The VT hides the previous object and displays the new one. The parent screen is updated.

21.28.4 Implementation Implementation Implementation Implementation Implementation Implementation Implementation Implementation Implementation Implementation (IMA) significantly reduces the complexity of the screen control. Instead of manually managing many objects using Hide/Show, the ECU only needs to change a single ID in the pointer. This saves CAN bus bandwidth and simplifies the program logic on the machine control

- **On Change Value:** Triggered by the command Change Numeric Value. Further information and examples can be found in the [ISOBUS Wiki - Object Pointer](#) by Tobias Tenberg.

Note: For detailed specifications on data types and message formats, please refer to the official ISO 11783-6:2018.

21.29 ID 28 – Macro – ISO 11783-6 – B.16

The **Macro** object with **ID 28** allows you to store a sequence of commands in the Virtual Terminal and execute them automatically upon certain events. This reduces the necessary communication via ISOBUS, as simple UI logic runs directly in the terminal.

21.29.1 Attributes and Record Format (Table B.56)

The following table describes the structure of the Macro object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 255 (VT v4) 0 – 65534 (VT v5+)	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	28	3	Object type = Macro.
-	Number of bytes to follow	Integer	2	0 – 65535	4 – 5	Number of bytes for the command list.

-	Repeat: {Command}	Binary	6 – n	-	6 ...	List of command packages. Each command must be a multiple of 8 bytes long (padding with FFh).
---	--------------------------	--------	-------	---	-------	---

21.29.2 Functionality and Structure

A macro consists of a list of VT commands (see ISO 11783-6, Annex F).

- **Padding:** Each command within a macro must be padded to a length of **8 bytes** (using 0xFF) if the actual command is shorter (e.g., Change Numeric Value).
- **Execution:** Macros can be started by events (e.g., On Press of a button) or by the ECU command Execute Macro.
- **Consistency:** The ECU is responsible for ensuring that macros only reference objects that actually exist in the pool.

21.29.3 Available Macro Commands (Excerpt)

Macros can use almost all commanding VT functions:

- **Visibility:** Hide/Show Object (show/hide containers).
- **Interaction:** Enable/Disable Object (Locking buttons/input), Select Input Object (Setting focus).
- **Values:** Change Numeric Value (Changing variables or pointers), Change String Value.
- **Geometry:** Change Child Location/Position (Moving/scrolling objects), Change Size, Change End Point.
- **Display:** Change Background Color, Change Font/Line/Fill Attributes.
- **Navigation:** Change Active Mask (Screen change), Change Soft Key Mask.
- **Audio:** Control Audio Device (Emit beeps).
- **Lists:** Change List Item (Change the content of input lists).

21.29.4 Events

Macros do not trigger events themselves, but are started by events from other objects.

The Macro object supports the following commands:

- Execute Macro
- Execute Extended Macro
- Get Attribute Value

21.29.5 Implementation Implementation Significance

Macros are a powerful tool for **performance optimization**:

1. **Response Time:** A screen change immediately after a key press occurs via macro without CAN delay.
2. **Relief:** The ECU doesn't have to handle purely graphical tasks (e.g., toggling an icon when a button is pressed).
2. **Complexity:** Multiple actions can be combined into a single macro (e.g., "Set variable to 0" AND "Show success message" AND "Play sound").

Note: For detailed specifications of the individual command codes, refer to the official ISO 11783-6:2018, Annex F.

21.30 ID 29 – Auxiliary Function Type 1 – ISO 11783-6 – J.4.2

The **Auxiliary Function Type 1** object with ID 29 defines the attributes and design of an auxiliary function control. *Note: This object is considered obsolete as of VT version 3 and is replaced by Type 2 (ID 31). It should no longer be used in new projects.*

21.30.1 Attributes and Record Format (Table J.1)

The following table describes the structure of the Auxiliary Function Type 1 object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	29	3	Object type = Auxiliary Function Type 1.
[1]	Background color	Integer	1	0 – 255	4	Background color.
[2]	Function type	Integer	1	0 – 2	5	0 = Latching Boolean, 1 = Analog, 2 = Non-latching Boolean.
-	Number of objects to follow	Integer	1	1 – 255	6	Number of directly contained objects (Designator).
-	Repeat: {Object ID}	Integer	2	0 – 65534	7 + ...	Object ID of a contained object.
-	{X Location}	Signed Integer	2	-32768 to +32767	9 + ...	X position relative to the upper left corner.
-	{Y Location}	Signed Integer	2	-32768 to +32767	11 + ...	Y position relative to the upper left corner.

21.30.2 Functionality

The VT uses these attributes to enforce assignment to a compatible auxiliary input. The designator symbol must fit within the area of a softkey; any excess is clipped.

The object serves as a function definition that can be assigned to a physical auxiliary input device.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, Annex J.

21.31 ID 30 – Auxiliary Input Type 1 – ISO 11783-6 – J.4.4

The **Auxiliary Input Type 1** object with **ID 30** defines the properties of a physical control element (e.g., joystick, switch) of an auxiliary input device. *Note: This object is considered obsolete as of VT version 3 and is replaced by Type 2 (ID 32).*

21.31.1 Attributes and Record Format (Table J.3)

The following table describes the structure of the Auxiliary Input Type 1 object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-----	------	------	--------------	---------------	-------------	-------------

-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	30	3	Object type = Auxiliary Input Type 1.
[1]	Background color	Integer	1	0 – 255	4	Background color.
[2]	Function type	Integer	1	0 – 2	5	0 = Latching Boolean, 1 = Analog, 2 = Non-latching Boolean.
[3]	Input ID	Integer	1	0 – 255	6	ID of the physical input to which this object refers.
[-]	Number of objects to follow	Integer	1	1 – 255	7	Number of directly contained objects (designator).
-	Repeat: {Object ID}	Integer	2	0 – 65534	8 + ...	Object ID of a contained object.
-	{X Location}	Signed Integer	2	-32768 to +32767	10 + ...	X position relative to the upper left corner.
-	{Y Location}	Signed Integer	2	-32768 to +32767	12 + ...	Y position relative to the upper left corner.

21.31.2 Functionality

This object describes a physical input element (e.g., a button or a joystick axis) on an Auxiliary Input Device. It is provided by the Auxiliary Input Device so that the VT knows the available inputs and their properties. The Input ID attribute links this object to the device's status messages.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, Annex J.

21.32 ID 31 – Auxiliary Function Type 2 – ISO 11783-6 – J.4.3

The **Auxiliary Function Type 2** object with ID 31 is the modern definition for auxiliary functions in ISOBUS (from VT version 3 onwards). It offers extended possibilities for assigning controls.

21.32.1 Attributes and Record Format (Table J.2)

The following table describes the structure of the Auxiliary Function Type 2 object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-----	------	------	--------------	---------------	-------------	-------------

-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	31	3	Object type = Auxiliary Function Type 2.
[1]	Background color	Integer	1	0 – 255	4	Background color.
[2]	Function attributes	Bitmask	1	-	5	Bitmask for function type and control (see below).
[[[]]]	Number of objects to follow	Integer	1	1 – 255	6	Number of directly contained objects (designator).
[[[]]]	Repeat: {Object ID}	Integer	2	0 – 65534	7 + ...	Object ID of a contained object.
[[[]]]	{X Location}	Signed Integer	2	-32768 to +32767	9 + ...	X position relative to the upper left corner.
-	{Y Location}	Signed Integer	2	-32768 to +32767	11 + ...	Y position relative to the upper left corner.

21.32.1.1 Function Attributes (Bitmask AID 2)

- **Bits 0–3:** Auxiliary function type (see Table J.5)
- 0: Boolean Latching
- 1: Analogue
- 2: Boolean Momentary
- 3: Boolean Latching (Dual)
- 4: Analogue (Dual)
- 5: Boolean Momentary (Dual)
- ...
- **Bits 4–5:** Reserved
- **Bit 6:** Assignment Restriction (0 = Free, 1 = Restricted, see ISO 11783-6).
- **Bit 7:** Single Assignment (1 = May only be assigned to one input).

21.32.2 Special Features

This object is actively used for “Auxiliary Mapping” in the terminal. It allows the ECU to inform the terminal which functions (e.g., “Lift up/down”) are available so that the user can assign them to buttons or joysticks.

It replaces the obsolete Type 1 (ID 29).

Note: For detailed specifications, refer to the official ISO 11783-6:2018, Annex J.

21.33 ID 32 – Auxiliary Input Type 2 – ISO 11783-6 – J.4.5

The **Auxiliary Input Type 2** object with **ID 32** defines a physical control element of an auxiliary input device (e.g., a button on a joystick) from VT version 3 onwards.

21.33.1 Attributes and Record Format (Table J.4)

The following table describes the structure of the Auxiliary Input Type 2 object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	32	3	Object type = Auxiliary Input Type 2.
[1]	Back-ground color	Integer	1	0 – 255	4	Back-ground color.
[2]	Function attributes	Bitmask	1	-	5	Bitmask for function type and control (see below).
[3]	Input ID	Integer	1	0 – 255	6	ID of the physical input.
[-]	Number of objects to follow	Integer	1	1 – 255	7	Number of directly contained objects (designator).
[-]	Repeat: {Object ID}	Integer	2	0 – 65534	8 + ...	Object ID of a contained object.
[-] {X Location}	Signed Integer	2	–32768 to +32767	10 + ...	X position relative to the upper left corner.	
-	{Y Location}	Signed Integer	2	–32768 to +32767	12 + ...	Y position relative to the upper left corner.

21.33.1.1 Function Attributes (Bitmask AID 2)

- **Bits 0–3:** Auxiliary function type (see Table J.5)
- 0: Boolean Latching
- 1: Analogue
- 2: Boolean Momentary
- 3: Boolean Latching

- ...
- **Bits 4–5:** Reserved
- **Bit 6:** Reserved
- **Bit 7:** Single Assignment (1 = May only be assigned to one function).

21.33.2 Functionality

The terminal uses this information to display the available keys and their physical properties to the user. When the user presses a key, the input device sends a status message with the current value (Boolean or Analog) and the Input ID to the ISOBUS.

It replaces the obsolete Type 1 (ID 30).

Note: For detailed specifications, refer to the official ISO 11783-6:2018, Annex J.

21.34 ID 33 – Auxiliary Control Designator Type 2 Object Pointer – ISO 11783-6 – J.4.7

The **Auxiliary Control Designator Type 2 Object Pointer** object with ID 33 allows a working set to graphically display the currently assigned auxiliary functions and their inputs in a dialog box.

21.34.1 Attributes and Record Format (Table J.6)

The following table describes the structure of the Auxiliary Control Designator Type 2 object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	33	3	Object type = Auxiliary Control Designator Type 2 Object Pointer.
[1]	Pointer Type	Integer	1	0 – 3	4	Reference type (see below).
[2]	Auxiliary Object ID	Integer	2	0 – 65534, 65535	5 – 6	Object ID of an Auxiliary Function (Type 2) or an Auxiliary Input (Type 2).

21.34.1.1 Pointer Types (AID 1)

- **0:** The VT displays the designator of the object specified by Auxiliary Object ID.
- **1:** The VT displays the designator of the object assigned to the object referenced by Auxiliary Object ID (e.g., if AID 2 is a function, the assigned input is shown).
- **2:** The VT displays the designator of the working set that owns the object specified by Auxiliary Object ID.
- **3:** The VT displays the designator of the working set that owns the object assigned to the object referenced by Auxiliary Object ID.

21.34.2 Events (Events - Table J.7)

The object responds to the following events:

- **On Change Value:** Triggered when the mapping is changed or the referenced object is modified. The VT updates the display.

21.34.3 Benefits for the Developer

This object allows you to create a graphical overview that shows the user: “This function on my machine is currently controlled by button Y on joystick Z.” Since the mapping is often done by the user at the terminal, this pointer allows for a dynamic display without the ECU needing prior knowledge of the actual assignment.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, Annex J.

21.35 ID 34 – Window Mask – ISO 11783-6 – B.19

The **Window Mask** object with **ID 34** (introduced with VT version 4) allows you to define a portion of the screen that can be updated independently of the main data mask or populated with content from other working sets.

21.35.1 Attributes and Record Format (Table B.61)

The following table describes the structure of the Window Mask object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	34	3	Object type = Window Mask.
-	Width	Integer	1	1 – 2	4	Width in user layout columns (only relevant for Type 0 Free Form, otherwise ignored).
-	Height	Integer	1	1 – 6	5	Height in user layout rows (only relevant for Type 0 Free Form, otherwise ignored).
-	Window Type	Integer	1	0 – 18	6	Window type (see below).
[1]	Background colour	Integer	1	0 – 255	7	Background color (only relevant for Type 0).
[2]	Options	Bitmask	1	0 – 3	8	Bit 0: Available (0=Not available/blanked, 1=Available) Bit 1: Transparent (1=Transparent background)

						ground, type 0 only).
[3]	Name	Integer	2	0 – 65534	9 – 10	Object ID of an output string (name for mapping screen).
-	Window Title	Integer	2	0 – 65534, 65535	11 – 12	Object ID of an output string (title in the window).
-	Window Icon	Integer	2	0 – 65534, 65535	13 – 14	Object ID of an output object (icon for mapping screen).
-	Number of object references to follow	Integer	1	0 – 2	15	Number of referenced objects (depending on the window type).
-	Number of objects to follow	Integer	1	0 – 255	16	Number of directly contained objects (only for Type 0 Free Form).
-	Number of macros to follow	Integer	1	0 – 255	17	Number of following macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65535	18 + ...	Referenced objects (for predefined types).
-	Repeat: {Object ID}	Integer	2	0 – 65534	var.	Contained objects (for Type 0 Free Form).
-	{X Location}	Signed Integer	2	-32768 to +32767	var.	X-position relative to the window.
-	{Y Location}	Signed Integer	2	-32768 to +32767	var.	Y-position relative to the window.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.35.1.1 Window Types (Excerpt from B.19.2)

- **0:** Free Form (Free design, working set defines content).
- **1:** 1x1 Numeric Output with units.
- **2:** 1x1 Numeric Output without units.
- **3:** 1x1 String Output.
- **4:** 1x1 Numeric Input with Units.
- ...
- **7:** 1x1 Horizontal Linear Bargraph.
- **8:** 1x1 Single Button.
- ...
- **10:** 2x1 Numeric Output with Units.

21.35.2 Events (Events - Table B.60)

The Window Mask object responds to the following events:

- **On Show:** When the window becomes visible as part of a user layout mask.
- **On Hide:** When the window is hidden.
- **On Refresh:** When child objects are changed.
- **On Change Background Colour:** Responds to a color change.
- **On Change Child Location / Position:** Updates child objects.
- **On Change Attribute:** Responds to general attribute changes.
- **On Pointing Event:** Touch events (only for Free Form Windows).

Further information and examples can be found in the [ISOBUS Wiki - Window Mask](#)] by Tobias Tenberg.

Note: For detailed specifications, please refer to the official ISO 11783-6:2018, B.19.

21.36 ID 35 – Key Group – ISO 11783-6 – B.20

The **Key Group** object with **ID 35** is used to group softkeys. This is primarily used in conjunction with **User Layout Soft Key Masks** (from VT version 4 onwards).

21.36.1 Attributes and Record Format (Table B.63)

The following table describes the structure of the Key Group object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	35	3	Object type = Key Group.
[1]	Options	Bitmask	1	0 – 3	4	Bit 0: Available (0=Not available/blanked, 1=Available)Bit 1: Transparent (1=Background color of keys is ignored).
[2]	Name	Integer	2	0 – 65534	5 – 6	Object ID of an output string or object pointer (name for mapping screen).
-	Key Group Icon	Integer	2	0 – 65534, 65535	7 – 8	Object ID of an output object

						(icon for mapping screen).
-	Number of objects to follow	Integer	1	1 – 4	9	Number of key objects in this group. Max. 4.
-	Number of macros to follow	Integer	1	0 – 255	10	Number of following macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65534	11 + ...	Object ID of a key object or object pointer to key.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After objects) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.36.2 Events (Table B.62)

The Key Group object reacts to the following events:

- **On Change Attribute:** Reacts to general attribute changes.

21.36.3 Meaning and Functionality

The Key Group object is used to define a logically related group of softkeys (e.g., “Hydraulic Functions”). This group is primarily used in **User Layout Soft Key Masks**.

- **User Mapping:** The user can decide at the terminal where they want this group of keys to be placed in their softkey bar. The VT forces the user to place the group *as a whole* to maintain logical coherence.
- **Transparency:** It is recommended to make Key Groups transparent so that the VT can set a uniform background color for the keys.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.20.

21.37 ID 36 – Graphics Context Object – ISO 11783-6 – B.18

The **Graphics Context Object (GCO)** with **ID 36** (from VT version 4 onwards) provides a dynamic graphics buffer (canvas) into which the ECU can draw at runtime. It is comparable to a bitmap whose content can be modified pixel-perfectly.

21.37.1 Attributes and Record Format (Table B.59)

The following table describes the structure of the Graphics Context Object in the object pool.

AID	Name	Type	Size (Bytes)	Range Value /	Record Byte	Description	
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.	
[0]	Type	Integer	1	36	3	Object type =	

						Graphics Context.	
[1]	Viewport Width	Integer	2	0 – 32767	4 – 5	Width of the visible window (display in the canvas).	
[2]	Viewport Height	Integer	2	0 – 32767	6 – 7	Height of the visible window.	
[3]	Viewport X	Signed Integer	2	–32768 to +32767	8 – 9	X-position of the viewport relative to the canvas (0 = Left).	
[4]	Viewport Y	Signed Integer	2	–32768 to +32767	10 – 11	Y-position of the viewport relative to the canvas (0 = Top).	
[5]	Canvas Width	Integer	2	0 – 32767	12 – 13	Total width of graphics memory.	
[6]	Canvas Height	Integer	2	0 – 32767	14 – 15	Total height of graphics memory.	
[7]	Viewport Zoom	Float	4	–32.0 to +32.0	16 – 19	Viewport zoom factor (see Table F.1).	
[8]	Graphics Cursor X	Signed Integer	2	–32768 to +32767	20 – 21	Current X position of the drawing cursor.	
[9]	Graphics Cursor Y	Signed Integer	2	–32768 to +32767	22 – 23	Current Y position of the drawing cursor.	
[10]	Fore-ground Colour	Integer	1	0 – 255	24	Fore-ground color for drawing operations.	
[11]	Back-ground Colour	Integer	1	0 – 255	25	Back-ground	

						color. Used for filling during parsing.	
[12]	Font Attributes	Integer	2	0 – 65534, 65535	26 – 27	Reference to the Font Attributes object (for text commands).	
[13]	Line Attributes	Integer	2	0 – 65534, 65535	28 – 29	Reference to the Line Attributes object (for line commands).	
[14]	Fill Attributes	Integer	2	0 – 65534, 65535	30 – 31	Reference to Fill Attributes object (for fill commands).	
[15]	Format	Integer	1	0 – 2	32	Canvas color depth: 0=1-bit, 1=4-bit, 2=8-bit.	
[16]	Options	Bitmask	1	0 – 3	33	Bit 0: Transparency	Bit 1: Color (0=FG/BG Attribute, 1=Line/Font/Fill Attribute).
[17]	Transparency Color	Integer	1	0 – 255	34	Transparency color (when Options Bit 0 is set).	

21.37.2 Functionality and Structure

The GCO consists of two main components:

1. **Canvas:** A persistent graphics memory (bitmap) with a defined size (Canvas Width/Height). The content is retained even when the object is not displayed.
2. **Viewport:** A “window” that displays a portion of the canvas. The viewport defines the size of the object on the canvas. By changing Viewport X/Y, the content can be scrolled (panned).

21.37.3 Graphics Context Commands

The canvas is manipulated using special commands (see ISO 11783-6, Annex F), such as:

- **Set Graphics Cursor:** Sets the writing position.
- **Draw Point / Line / Rectangle / Polygon / Ellipse:** Draws geometric shapes.
- **Draw Text:** Writes text to the cursor position.
- **Copy Canvas:** Copies areas within the canvas.

21.37.4 Events (Table B.58)

The Graphics Context object reacts to the following events:

- **On Change Attribute:** Triggered when attributes change (e.g., viewport position, zoom). The VT updates the display.
- **On Change Background Colour:** Fills the object (the canvas) with the new background color.
Caution: Deletes the previous content!

21.37.5 Application Example

Typical applications include **GPS maps** (swath logging), where the ECU continuously plots the driven route as a line in the GCO.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.18.

21.38 ID 37 – Output List – ISO 11783-6 – B.9.4

The **Output List** object with **ID 37** (from VT version 4 onwards) is used to display one of several objects from a list. Which object is currently visible is controlled by an index (value).

21.38.1 Attributes and Record Format (Table B.25)

The following table describes the structure of the Output List object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	37	3	Object type = Output List.
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the display area in pixels. Clipping occurs outside.
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the display area in pixels. Clipping occurs outside.
[3]	Variable reference	Integer	2	0 – 65534, 65535	8 – 9	Reference to a Number Variable object for the index.
[4]	Value	Integer	1	0 – 255	10	Current index (0-254). 255 = no display. Only if variable Ref == NULL.
-	Number of list items	Integer	1	0 – 255	11	Number of items in the list.

-	Number of macros to follow	Integer	1	0 – 255	12	Number of following macro references.
-	Repeat: {Object ID}	Integer	2	0 – 65534, 65535	13 + ...	Object ID of a list item (what to display). NULL = Empty placeholder.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	(After objects) Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.38.2 Functionality

The Output List behaves similarly to an animation, but is manually controlled via the index.

- **Index:** The displayed content is determined by the value (Value or Variable). Index 0 displays the first object in the list.
- **Special Value 255:** With a value of 255, nothing is displayed (the object is invisible).
- **NULL Pointer:** If a list entry has the ID NULL (65535), nothing is displayed for that index.

21.38.3 Events (Table B.24)

The Output List object reacts to the following events:

- **On Change Value:** Triggered when the index changes. The VT updates the display.
- **On Change Attribute:** Reacts to general attribute changes.
- **On Change Size:** Reacts to size changes.
- **On Refresh:** Triggered when the VT needs to redraw the object.

Further information and examples can be found in the [ISOBUS Wiki - List \(Output\)](#) by Tobias Tenberg.

Note: For detailed specifications, please refer to the official ISO 11783-6:2018, B.9.4.

21.39 ID 38 – Extended Input Attributes – ISO 11783-6 – B.14.6

The **Extended Input Attributes** object with **ID 38** (from VT version 4 onwards) is used to validate text input (input string) when using **WideStrings** (Unicode).

21.39.1 Attribute and Record Format (Table B.53)

The following table describes the structure of the Extended Input Attributes object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	38	3	Object type = Extended Input Attributes.
[1]	Validation type	Integer	1	0 – 1	4	0 = Allowed characters (whitelist), 1 = Forbid-

						den characters (blacklist).
-	Number of code planes to follow	Integer	1	1 – 17	5	Number of defined Unicode planes.
-	Repeat: {Code plane number}	Integer	1	0 – 16	6 ...	Unicode plane number (0 = BMP).
-	{Number of character ranges to follow}	Integer	1	1 – 255	7 ...	Number of ranges in this plane.
-	Repeat: {First character}	Integer	2	0 – 65535	8 ...	Start character of the range (WideChar).
-	{Last character}	Integer	2	0 – 65535	10 ...	End character of the range (WideChar).

21.39.2 Functionality

While the simple *Input Attributes* object (ID 26) only supports 8-bit characters, this object allows for fine-grained control of allowed Unicode characters (WideString). This is necessary for languages with more than 256 characters (e.g., Asian, Cyrillic).

Validation is performed by defining ranges within Unicode layers.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.14.6.

21.40 ID 39 – Colour Map – ISO 11783-6 – B.17

The **Colour Map** object with **ID 39** (optional from VT version 4/5, mandatory from VT version 6) allows a working set to redefine the terminal's color table at runtime.

21.40.1 Attributes and Record Format (Table B.57)

The following table describes the structure of the Colour Map object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	39	3	Object type = Colour Map.
-	Number of colour indexes to follow	Integer	2	2, 16, 256	4 – 5	Number of following entries. Must match the VT's color depth (see Get Hardware).
-	Repeat: {Colour Map}	Integer	var.	0 – 255	6 ...	List of colour indexes.

21.40.1.1 Structure of the entries

The entries define a redirection of the colour indexes.

- **Example:** If the entry at index 0 has the value 1, colour 1 (white) will now be displayed wherever colour 0 (black) was used in the design.
- The length of the entries depends on the VT type:
- **Type 0 (Monochrome):** 2 entries.
- **Type 1 (16 colours):** 16 entries.
- – **Type 2 (256 colors):** 256 entries.

21.40.2 Functionality

Normally, a terminal uses a standard color palette (e.g., 256 colors according to ISO). With the Colour Map, a working set can define, for example, that color index 1 (default: red) should be displayed as blue instead.

- **Indirection:** This provides a level of indirection. Instead of changing each object individually, the entire application's color scheme can be changed (e.g., day/night mode) simply by toggling the Colour Map.
- **Activation:** A Colour Map is activated using the command `Select Colour Map`.

21.40.3 Note on VT Version 6

Starting with version 6, the Colour Palette object (ID 45) is also supported, which allows the RGB values of the colors themselves to be changed. The Colour Map maps indices to indices, while the Colour Palette maps indices to RGB values.

Note: For detailed specifications, please refer to the official ISO 11783-6:2018, B.17.

21.41 ID 40 – Object Label Reference List – ISO 11783-6 – B.21

The **Object Label Reference List** object with **ID 40** (from VT version 5 onwards) is used to assign a list of label objects to objects (such as variables or input fields).

21.41.1 Attributes and Record Format (Table B.64)

The following table describes the structure of the Object Label Reference List object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	40	3	Object type = Object Label Reference List.
[1]	Number of Labelled objects	Integer	2	0 – 65535	4 – 5	Number of subsequent label assignments.
-	Repeat: {Object ID}	Integer	2	0 – 65534	6 – 7 ...	Object ID of the object to be labeled (e.g., Input Number).
-	{String Variable reference}	Integer	2	0 – 65535	8 – 9 ...	Object ID of a string variable containing the label text (or FFFFh = no text).
-	{Font type}	Integer	1	0 – 255	10 ...	Font type (see Annex K). Ignored for

						WideString or NULL.
-	{Object Label graphic representation}	Integer	2	0 – 65535	11 – 12 ...	Object ID of a graphic (icon) for the label (or FFFFh = no graphic).

21.41.2 Meaning and Functionality

This object is used to assign a name (text) and an icon (graphic) to other objects (e.g., working sets, input fields). These “labels” are used by the VT:

- **Working Set Label:** The label for the working set object is displayed in the list of active workgroups.
- **Input Labels:** For input fields, the VT displays the label (text/graphic) in the popup editor so the user knows which value they are currently editing (e.g., “seed quantity” instead of just “120”).

Only **one** Object Label Reference List is allowed per object pool.

Note: For detailed specifications, please refer to the official ISO 11783-6:2018, B.21.

21.42 ID 41 – External Object Definition – ISO 11783-6 – B.22

The **External Object Definition** object with **ID 41** (from VT version 5 onwards) is part of the mechanism for **cross-working-set object references**. It defines which objects in a user’s own pool may be referenced by other working sets.

21.42.1 Attributes and Record Format (Table B.66)

The following table describes the structure of the External Object Definition object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	41	3	Object type = External Object Definition.
[1]	Options	Bitmask	1	0 – 1	4	Bit 0: Enabled (1=Share active, 0=Disabled).
[2]	NAME 0	Integer	4	0 – 2 ³² -1	5 – 8	Bytes 1–4 of the NAME (ISO 11783-5) of the authorized working set.
[3]	NAME 1	Integer	4	0 – 2 ³² -1	9 – 12	Bytes 5–8 of the NAME of the authorized working set.
-	Number of objects to follow	Integer	1	0 – 255	13	Number of shared objects.
-	Repeat: {Object ID}	Integer	2	0 – 65534, 65535	14 + ...	Object ID of an object from the local pool that is shared

						with the external work set.
--	--	--	--	--	--	-----------------------------

21.42.2 Functionality

For Working Set A to display an object from Working Set B (via an *External Object Pointer*), Working Set B must explicitly share this object with Working Set A in an *External Object Definition*. This ensures security and control over the local pool resources.

Recommendation: The Enabled bit should initially be set to 0 when the pool is loaded and only activated at runtime when the partner's NAME is known and up-to-date.

21.42.3 Events (Table B.65)

The External Object Definition object responds to the following events:

- **On Change Attribute:** Triggered by the command Change Attribute (e.g., enabling/disabling). The VT re-evaluates all currently displayed External Object Pointers that reference objects in this WS.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.22.

21.43 ID 42 – External Reference NAME – ISO 11783-6 – B.23

The **External Reference NAME** object with **ID 42** (from VT version 5 onwards) identifies another working set on the bus via its unique ISO NAME.

21.43.1 Attributes and Record Format (Table B.68)

The following table describes the structure of the External Reference NAME object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	42	3	Object Type = External Reference NAME.
[1]	Options	Bitmask	1	0 – 1	4	Bit 0: Enabled (1=Reference active, 0=Disabled).
[2]	NAME 0	Integer	4	0 – 2 ³² -1	5 – 8	Bytes 1–4 of the NAME (ISO 11783-5) of the referenced working set.
[3]	NAME 1	Integer	4	0 – 2 ³² -1	9 – 12	Bytes 5–8 of the NAME of the referenced working set.

21.43.2 Functionality

This object functions as an “address book entry.” When an ECU wants to load an object from another ECU, it uses this entry to tell the terminal which ECU (identified by the NAME) the object originates from.

- **Activation:** Should be initially disabled. When the referenced working set is online, the ECU can activate this object.

21.43.3 Events (Table B.67)

The External Reference NAME object responds to the following events:

- **On Change Attribute:** Triggered by the command Change Attribute. The VT re-evaluates all currently displayed external object pointers that reference this NAME object.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.23.

21.44 ID 43 – External Object Pointer – ISO 11783-6 – B.24

The **External Object Pointer** object with ID 43 (from VT version 5 onwards) allows a working set to display objects that are physically located in the object pool of **another** working set.

21.44.1 Attributes and Record Format (Table B.70)

The following table describes the structure of the External Object Pointer object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	43	3	Object type = External Object Pointer.
[1]	Default Object ID	Integer	2	0 – 65534, 65535	4 – 5	ID of a local fallback object if external access fails.
[2]	External Reference NAME ID	Integer	2	0 – 65534, 65535	6 – 7	ID of an External Reference NAME object that identifies the target worksheet.
[3]	External Object ID	Integer	2	0 – 65534, 65535	8 – 9	ID of the object in the external pool.

21.44.2 Functionality and Rules

This object allows the display of objects from external object pools.

- **Display:** The terminal searches for the object in the pool of the ECU identified by the NAME and draws it at the pointer's location.
- **Context:** Events (e.g., Button Press) and macros are executed in the context of the **original working set** (owner of the object). Messages (e.g., Button Activation) are sent to the owner.
- **Fallback:** If the external object is not found, is not released, or is invalid, **Default Object** is displayed.
- **Security:** The target object must have been released by the owning ECU via *External Object Definition* (ID 41).

21.44.3 Events (Events - Table B.69)

The External Object Pointer object reacts to the following events:

- **On Change Attribute:** Triggered by the command Change Attribute. The VT re-evaluates the pointer and redraws it if necessary.

21.44.4 Application Example

A tractor (Working Set A) wants to display the fill level of a trailed fertilizer spreader (Working Set B) in its main interface. It uses an *External Object Pointer* that points to the fill level indicator in the spreader's reservoir.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.24.

21.45 ID 44 – Animation – ISO 11783-6 – B.25

The **Animation** object with **ID 44** (from VT version 5 onwards) enables the automatic or manual playback of a sequence of objects (frames).

21.45.1 Attributes and Record Format (Table B.72)

The following table describes the structure of the Animation object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description	
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.	
[0]	Type	Integer	1	44	3	Object type = Animation.	
[1]	Width	Integer	2	0 – 65535	4 – 5	Width of the animation area in pixels.	
[2]	Height	Integer	2	0 – 65535	6 – 7	Height of the animation area in pixels.	
[3]	Refresh Interval	Integer	2	0 – 65535	8 – 9	Time in ms between frames (0 = Timer off).	
[4]	Value	Integer	1	0 – 255	10	Current list index of the displayed object (0-254).	
[5]	Enabled	Integer	1	0 or 1	11	0 = Stopped (Disabled), 1 = Animating (Enabled).	
[6]	First Child Index	Integer	1	0 – 254	12	Start index of the animation sequence.	

[7]	Last Child Index	Integer	1	0 – 254	13	End index of the animation sequence.	
[8]	Default Child Index	Integer	1	0 – 254	14	Index of the object displayed when the animation is disabled (depending on Options).	
[9]	Options	Bitmask	1	0 – 7	15	Bit 0: Sequence (0=Single Shot, 1=Loop)	Bits 1–2: Disabled Behavior (0=Pause, 1=Reset to First, 2=Default Object, 3=Blank).

- **Number of objects to follow** | Integer | 1 | 0 – 255 | 16 | Number of frames (objects) included. | | - |
- **Number of macros to follow** | Integer | 1 | 0 – 255 | 17 | Number of following macro references. | | - |
- **Repeat:** {Object ID} | Integer | 2 | 0 – 65534 | 18 + ... | Object ID of a frame (child object). | | - | {X Location} | Signed Integer | 2 | –32768 to +32767 | 20 + ... | X position relative to the animation. | | - | {Y Location} | Signed Integer | 2 | –32768 to +32767 | 22 + ... | Y position relative to the animation. | | - | **Repeat:** {Event ID} | Integer | 1 | 0 – 255 | var. | Event ID that triggers the macro. | | - | {Macro ID} | Integer | 1 | 0 – 255 | var. | Macro ID of the macro to be executed. |

21.45.2 Functionality

The animation object manages a list of child objects. When Enabled is 1, the terminal automatically increments Value (index) at the same rate as Refresh Interval.

- **Loop:** After reaching Last Child Index, the loop starts again at First Child Index.
- **Single Shot:** The animation stops at the last frame.
- **Deactivation:** The behavior when stopping (pause, reset to frame 1, default image, or blank) is controlled via the options.

21.45.3 Events (Events - Table B.71)

The animation object reacts to the following events:

- **On Enable / On Disable:** State change.
- **On Change Value:** When the index is changed (manually or automatically). The VT draws the new object.
- **On Change Attribute:** Reacts to general attribute changes.
- **On Change Size:** Reacts to size changes.
- **On Refresh:** Triggered by the timer (refresh interval) or other refresh conditions.

21.45.4 Recommendation

To avoid overloading the terminal's performance, individual objects should be kept small. A refresh interval of at least 200 ms is recommended.

Note: For detailed specifications, please refer to the official ISO 11783-6:2018, B.25.

21.46 ID 45 – Colour Palette – ISO 11783-6 – B.26

The **Colour Palette** object with **ID 45** (from VT version 6 onwards) allows a working set to completely replace the terminal's default color palette with its own ARGB color definitions.

21.46.1 Attributes and Record Format (Table B.73)

The following table describes the structure of the Colour Palette object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	45	3	Object type = Color Palette.
[1]	Options	Bitmask	1	0	4	Reserved (should be 0).
-	Number of ARGB values to follow	Integer	2	0 – 256	5 – 6	Number of following color definitions.
-	Repeat: {B}	Integer	1	0 – 255	7 + ...	Blue value.
-	{G}	Integer	1	0 – 255	8 + ...	Green value.
-	{R}	Integer	1	0 – 255	9 + ...	Red value.
-	{A}	Integer	1	0 – 255	10 + ...	Alpha value (0 = Transparent, 255 = Opaque).

21.46.2 Functionality

When a Colour Palette object is active, it replaces the default colors of the terminal for this working set. The colors are populated starting at index 0.

Activation: A Colour Palette can be set as the default via the *Working Set Special Controls* object (ID 47) or activated at runtime using the *Select Colour Map or Palette* command.

Transparency: The alpha channel enables semi-transparent colors. However, this should be used with caution, especially with overlapping objects.

21.46.3 Difference to the Colour Map (ID 39)

While the *Colour Map* only changes the assignment (indexing) of existing terminal colors, the *Colour Palette* redefines the physical colors (ARGB values). This enables true branding in company colors.

21.46.4 Events

The object itself does not trigger any events, but it affects the display of all objects that use colors.

- **On Change Attribute:** Triggered when attributes are changed. The virtual machine may need to redraw the entire screen.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.26.

21.47 ID 46 – Graphic Data (PNG) – ISO 11783-6 – B.27

The **Graphic Data** object with **ID 46** (from VT version 6 onwards) is used to store raw data for graphics, especially in **PNG format**.

21.47.1 Attributes and Record Format (Table B.74)

The following table describes the structure of the Graphic Data object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	46	3	Object type = Graphic Data.
[1]	Format	Integer	1	0	4	Graphic format: 0 = PNG (max. 32-bit RGBA).
-	Number of bytes in raw data	Integer	4	0 – 2 ³² -1	5 – 8	Number of bytes in the raw data.
-	Repeat: {raw data}	Integer	1	0 – 255	9 ...	Raw data of the graphic (bytes).

21.47.2 Special Features

The **Graphic Data** object (from VT version 6 onwards) is used to store raw data for graphics, especially in **PNG format**. Unlike the classic *Picture Graphic* object (ID 20), which was based on simple bitmaps, this object uses the industry standard PNG.

Independence: The object contains its own color palette (within the PNG data) and is therefore **not** affected by the *Color Map* (ID 39) or *Color Palette* (ID 45) of the Working Set.

21.47.3 Usage

This object is not normally placed directly in a mask, but is referenced by a **Scaled Graphic** object (ID 48) to display and scale it to the desired size. It can also be referenced in an *External Object Pointer* (ID 43).

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.27.

21.48 ID 47 – Working Set Special Controls – ISO 11783-6 – B.29

The **Working Set Special Controls** object with ID 47 (from VT version 6 onwards) is used for the central control of pool-wide settings such as colors and languages.

21.48.1 Attributes and Record Format (Table B.78)

The following table describes the structure of the Working Set Special Controls object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.
[0]	Type	Integer	1	47	3	Object type = Working Set Special Controls.
[1]	Number of Bytes to follow	Integer	2	5 – 65535	4 – 5	Number of bytes in this object after this attribute. Enables parsing compatibility.

[2]	Object ID of Colour Map object	Integer	2	0 – 65534, 65535	6 – 7	ID of the initial Colour Map to be used (ID 39) or NULL.
[3]	Object ID of Colour Palette object	Integer	2	0 – 65534, 65535	8 – 9	ID of the initial Colour Palette to be used (ID 45) or NULL.
-	Number of languages pairs to follow	Integer	1	0 – 255	10	Number of the following language/country code pairs.
-	Repeat: {Language Code}	String	2	-	11 – 12 ...	2-letter code according to ISO 639-1 (e.g., “de”).
-	{Country Code}	String	2	-	13 – 14 ...	2-letter code according to ISO 3166-1 (e.g., “DE”) or “20 20” (Hex) if not applicable.

21.48.2 Meaning and Functionality

This object is the central point of contact for the terminal when loading the pool (from VT version 6 onwards).

- **Colors:** It defines which *Color Map* and *Color Palette* should be used **initially** when activating the pool.
- **Languages:** This defines a list of supported languages that **replaces** the list in the *Working Set* object (ID 0). Combining language and country codes (e.g., pt + BR vs. pt + PT) allows for more precise selection.
- **Extensibility:** The attribute Number of Bytes to follow allows the object to be extended with new attributes in the future without confusing older VTs (they simply skip the unknown bytes).

There can be a maximum of **one** Working Set Special Controls object per object pool.

21.48.3 Events (Table B.77)

The object responds to the following events:

- **On Refresh:** Triggered when settings change that require a complete rebuild of the display (e.g., color palette).

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.29.

21.49 ID 48 – Scaled Graphic – ISO 11783-6 – B.28

The **Scaled Graphic** object with **ID 48** (from VT version 6 onwards) is used to display and scale graphic objects.

21.49.1 Attributes and Record Format (Table B.76)

The following table describes the structure of the Scaled Graphic object in the object pool.

AID	Name	Type	Size (Bytes)	Range / Value	Record Byte	Description
-	Object ID	Integer	2	0 – 65534	1 – 2	Unique ID in the object pool.

[0]	Type	Integer	1	48	3	Object type = Scaled Graphic.
[1]	Width	Integer	2	0 – 65535	4 – 5	Target width in pixels.
[2]	Height	Integer	2	0 – 65535	6 – 7	Target height in pixels.
[3]	ScaleType	Integer	1	0 – 127	8	Scaling mode and adjustment (see below).
[4]	Options	Bitmask	1	0 – 1	9	Bit 0: Flashing (0=Normal, 1=Blinking).
[5]	Value	Integer	2	0 – 65535	10 – 11	Object ID of the graphic object to be displayed (Graphic Data ID 46 or Picture Graphic ID 20) or pointer.
-	Number of macros to follow	Integer	1	0 – 255	12	Number of following macro references.
-	Repeat: {Event ID}	Integer	1	0 – 255	var.	Event ID that triggers the macro.
-	{Macro ID}	Integer	1	0 – 255	var.	Macro ID of the macro to be executed.

21.49.1.1 Scaling Types (Bits 0-2 of ScaleType)

- **0:** No scaling (use original size from the raw data).
- **1:** Scale to width (maintain aspect ratio).
- **2:** Scale to height (maintain aspect ratio).
- **3:** Scale to both width and height (distortion possible).
- **4:** Fit to area (Best Fit, maintain aspect ratio, graphic becomes as large as possible).

21.49.1.2 Adjustment (Bits 3-6 of ScaleType)

Defines the position within the area defined by Width and Height:

- **Horizontal (Bits 3-4):** 0=Left, 1=Center, 2=Right.
- **Vertical (Bits 5-6):** 0=Top, 1=Center, 2=Bottom.

21.49.2 Events (Events - Table B.75)

The Scaled Graphic object responds to the following events:

- **On Refresh:** Triggered on a mask refresh or option change.
- **On Change Attribute:** Responds to general attribute changes.
- **On Change Value:** Triggered when the referenced graphic object (value) is changed. The VT loads and scales the new image.

Note: For detailed specifications, refer to the official ISO 11783-6:2018, B.28.

21.50 ISOBUS VT Objects

This section describes the different object types of the ISOBUS Virtual Terminal according to ISO 11783-6.

21.50.1 List of objects

- ID 0 - Working Set
- ID 1 - Data Mask
- ID 2 - Alarm Mask
- ID 3 - Container
- ID 4 - Soft Key Mask
- ID 5 - Key (Soft Key)
- ID 6 - Button
- ID 7 - Input Boolean
- ID 8 - Input String
- ID 9 - Input Number
- ID 10 - Input List
- ID 11 - Output String
- ID 12 - Output Number
- ID 13 - Output Line
- ID 14 - Output Rectangle
- ID 15 - Output Ellipse
- ID 16 - Output Polygon
- ID 17 - Output Meter
- ID 18 - Output Linear Bar Graph
- ID 19 - Output Arched Bar Graph
- ID 20 - Picture Graphic
- ID 21 - Number Variable
- ID 22 - String Variable
- ID 23 - Font Attributes
- ID 24 - Line Attributes
- ID 25 - Fill Attributes
- ID 26 - Input Attributes
- ID 27 - Object Pointer
- ID 28 - Macro
- ID 29 - Auxiliary Function Type 1
- ID 30 - Auxiliary Input Type 1
- ID 31 - Auxiliary Function Type 2
- ID 32 - Auxiliary Input Type 2
- ID 33 - Auxiliary Control Designator Type 2 Object Pointer
- ID 34 - Window Mask
- ID 35 - Key Group
- ID 36 - Graphics Context Object
- ID 37 - Output List
- ID 38 - Extended Input Attributes
- ID 39 - Color Map
- ID 40 - Object Label Reference List
- ID 41 - External Object Definition
- ID 42 - External reference NAME
- ID 43 - External Object Pointer
- ID 44 - Animation
- ID 45 - Color Palette
- ID 46 - Graphic Data (PNG)
- ID 47 - Working Set Special Controls
- ID 48 - Scaled Graphic

22. Lehrplan praxis

22.1 Input Attributes, Key Events & Interaction Diagnostics

 **Source & Legal Basis:** **Source:** *Framework Curriculum for the Preparation for the Master Craftsman Examination in Agricultural and Construction Machinery Mechatronics* **Publisher:** German Association of Agricultural and Construction Machinery Mechatronics Technicians (LandBauTechnik-Bundesverband e. V.), Alfredstraße 102, 45131 Essen (as of February 25, 2025) **Regulation:** *Master Craftsman Examination Regulation (LandBauMechMstrV)* of September 9, 2024 (Federal Law Gazette 2024 I No. 277, effective August 1, 2025) **Framework Curriculum Category:** Part I LE 2.2 (40 teaching units) | Part II LE 1.1 & 1.3 **Relevance:** Input processing, button events, and diagnostics of HMI communication errors.

22.1.1 1. Interactive Objects & Input Fields

- **Input Number / Input String:** Input of target parameters (e.g., application rate in l/ha, working width in m).
 - **Button Objects:** Buttons and switches for triggering machine actions (e.g., start folding, turn on pump).
 - **Key Events:** Processing of VT_KEY_EVENT (Button Pressed, Button Released, Value Changed).
-

22.1.2 2. Error Diagnostics on VT Interfaces

- **Object Pool Upload Errors:** Diagnostics for transmission interruptions (Memory Overflow, Version Incompatibility).
- **CAN Bus Messages:** Analysis of VT-to-ECU and ECU-to-VT messages using CAN trace tools.

Canal bus messages:** Analysis of VT-to-ECU and ECU-to-VT messages using CAN trace tools.

22.2 ISOBUS VT Object Pools & HMI (ISO 11783-6)

 **Source & Legal Basis:** **Source:** *Framework Curriculum for the Preparation for the Master Craftsman Examination in Agricultural and Construction Machinery Mechatronics* **Publisher:** German Association of Agricultural and Construction Machinery Mechatronics Engineers (LandBauTechnik-Bundesverband e. V.), Alfredstraße 102, 45131 Essen (as of February 25, 2025) **Regulation:** *Master Craftsman Examination Regulation (LandBauMechMstrV)* of September 9, 2024 (Federal Law Gazette 2024 I No. 277, effective August 1, 2025) **Framework Curriculum Category:** Part I LE 1.4 (80 teaching units) | Part II LE 1.1 (120 hours) **Relevance:** Object-based user interfaces for ISOBUS implements according to ISO 11783-6.

22.2.1 1. Fundamentals of the Virtual Terminal (VT / UT)

- **ISO 11783 Part 6:** Specification of the communication between the implement (ECU) and the Universal Terminal (UT).
 - **Object Pool Concept:** The implement transmits a binary object pool to the terminal at system startup. The terminal renders the graphical elements independently.
 - **ISO Designer:** Visual development tool for creating and managing VT object pools (XML / IOP files).
-

22.2.2 2. Core Objects in the Object Pool

- **Working Set Object:** Represents the implement in the terminal menu.
- **Data Mask Object:** The actual display mask / main screen on the terminal.
- **SoftKey Mask Object:** Sidebar with function keys (softkeys) that can be dynamically adjusted.
- **Alarm Mask Object:** Prioritized warning and emergency messages for security events.

SoftKey Mask Object:

22.3 Mask Layout, Softkeys & Ergonomics

 **Source & Legal Basis:** **Source:** *Framework Curriculum for the Preparation for the Master Craftsman Examination in Agricultural and Construction Machinery Mechatronics* **Publisher:** German Association of Agricultural and Construction Machinery Mechatronics Technicians (LandBauTechnik-Bundesverband e. V.), Alfredstraße 102, 45131 Essen (as of February 25, 2025) **Regulation:** *Master Craftsman Examination Regulation (LandBauMechMstrV)* of September 9, 2024 (Federal Law Gazette 2024 I No. 277, effective August 1, 2025) **Framework Curriculum Category:** Part I LE 1.4 | Part II LE 1.4 (80 teaching units) **Relevance:** Ergonomic design of machine displays and intuitive softkey hierarchies.

22.3.1 1. Data Mask Design

- **Object Placement:** Structuring of output values (numerical values, bar graphs, graphics, icons).
 - **Color Tables & Scaling:** Use of standard color codes according to ISO 11783-6 for status indicators (Green = Normal Operation, Yellow = Warning, Red = Error).
 - **Dynamic Hide/Show:** Hiding inactive controls to prevent driver distraction.
-

22.3.2 2. Softkey Structures

- **Physical vs. Virtual Keys:** Mapping of on-screen buttons to physical buttons on the terminal frame or joystick (AUX-N / AUX-O).
- **Softkey Levels:** Hierarchical navigation between the main menu, settings, and diagnostic screens.