DIN EN ISO9001:2000
certified

ADDI-DATA GmbH
Dieselstraße 3
D-77833 OTTERSWEIER
+49 (0)7223 / 9493 – 0

Software description

ADDIPack

LabView driver

Edition: 01.02-09/2005

Product information

This manual contains the technical installation and important instructions for correct commissioning and usage, as well as production information according to the current status before printing.
The content of this manual and the technical product data may be changed without prior notice.
ADDI-DATA GmbH reserves the right to make changes to the technical data and the materials included herein.

Warranty and liability

The user is not permitted to make changes to the product beyond the intended use, or to interfere with the product in any other way.
ADDI-DATA shall not be liable for obvious printing and phrasing errors. In addition, ADDI DATA, if legally permissible, shall not be liable for personal injury or damage to materials caused by improper installation and/or commissioning of the board by the user or improper use, for example, if the board is operated despite faulty safety and protection devices, or if notes in the operating instructions regarding transport, storage, installation, commissioning, operation, thresholds, etc. are not taken into consideration. Liability is further excluded if the operator changes the board or the source code files without authorisation and/or if the operator is guilty of not monitoring the permanent operational capability of working parts and this has led to damage.

Copyright

ADDI-DATA software product licence

Please read this licence carefully before using the standard software. The customer is only granted the right to use this software if he/she agrees with the conditions of this licence.
The software must only be used to set up the ADDI-DATA boards.
Reproduction of the software is forbidden (except for back-up and for exchange of faulty data carriers). Disassembly, decompilation, decryption and reverse engineering of the software are forbidden. This licence and the software may be transferred to a third party if this party has acquired a board by purchase, has agreed to all the conditions in this licence contract and the original owner does not keep any copies of the software.

Trademarks

Update summary

Edition number Date (MM/YY)	Author	Update description
1st edition 18/05	RH	Creation

1 DRIVER INSTALLATION

1.1 LabView compatibility

The driver is compatible with LabView versions up to LabView 6i.

1.2 Installation

LabView has to be closed. Please copy the ADDI-DATA folder in the LabView folder called INSTR.LIB. You can now start LabView.
INSTR.LIB is generally located under the LabView main folder:

C:\Programme\National Instruments\LabVIEW\INSTR.LIB

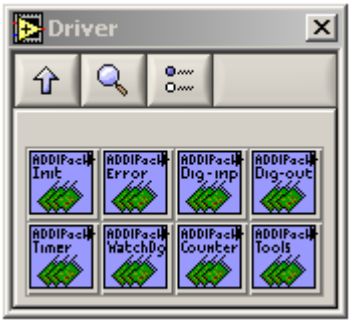
Remark: If you receive a message that proposed to overwrite the “dir.mnu” file, then accept it.

1.3 Driver location under LabView

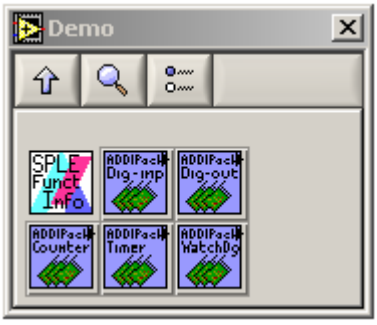
Under LabView driver are installed in the **Function Palette** under **Instrument Driver**  . You will see an **ADDI-DATA** icon  under Menu where drivers and sample are located.

General:

Driver:



Demos:



2 ADDIPACK FUNCTION DESCRIPTION

2.1 Initialisation driver

2.1.1  Open the driver.vi

C equivalent:

i_ADDIDATA_OpenWin32Driver

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Board identifier:	Handle of the board (not implemented).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
This function loads the ADDIDATA driver.
All information concerning the ADDI-DATA boards is read.

Return value:

0:	No error
Other value:	Error by calling the VI. Use the Error cluster to find the error source.

2.1.2  Close the driver.vi

C equivalent:

b_ADDIDATA_CloseWin32Driver

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Closes the ADDI-DATA driver and blocks the access to all boards.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.2 Information driver

2.2.1  Get localisation information.vi

C equivalent:
i_ADDIDATA_GetLocalisation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Entity virtual index: Virtual index of the selected channel or module.

Entity type: Type of the selected entity:
0: ADDIDATA_LOCALISATION_CHANNEL
1: ADDIDATA_LOCALISATION_MODULE

Functionality: Functionality of the selected entity:
0: ADDIDATA_DIGITAL_INPUT
1: ADDIDATA_DIGITAL_OUTPUT
2: ADDIDATA_ANALOG_INPUT
3: ADDIDATA_ANALOG_OUTPUT
4: ADDIDATA_TIMER
5: ADDIDATA_WATCHDOG
6: ADDIDATA_TEMPERATURE
7: ADDIDATA_COUNTER
9: ADDIDATA_RESISTANCE
11: ADDIDATA_PRESSURE
12: ADDIDATA_TRANSDUCER

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

Channel real index: Real index of the channel (as understood by the board)

Module real index: Real Index of the module (as understood by the board)

Board name: Board name of the device containing the entity

Board address[6]: Array of the device base address containing the entity

Board interrupt: Interrupt of the device containing the entity

Device number: Device number (BIOS) of the device containing the entity

PCI slot number information[MAX_PATH]: PCI slot of the device containing the entity

Device serial number: Serial number of the device containing the entity

Error out: Error cluster output.

Task:
Localise the entity (channel or module) of the virtual board in the real hardware configuration. Can be called up even if the driver is not open.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.3 Error driver

2.3.1  Manage the last error.vi

C equivalent:

No equivalent.

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Manages the last error and format it to the error cluster.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.3.2  Get the last error.vi

C equivalent:

i_ADDIDATA_GetLastError

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Function Number	Number of the function on which an error has occurred. See Table 2-2 of the error documentation
Error Code	Error number. See Table 2-3 of the error documentation
Error Level	0: An error has occurred in the application (ring 3) 1: The function which caused the error has been called up in the user interrupt routine. This parameter has only a meaning for 32-bit operating systems. (ring 0)
Error out:	Error cluster output.

Task:

Transmits the last error message.

Return value:

0:	No error
1:	Error message present
101:	Function cannot be called up in the user interrupt routine
102:	Error message not found

2.3.3  Format the error message.vi

C equivalent:

b_ADDIDATA_FormatErrorMessage

Parameters:

- Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error Number	Error code of the Get the Last Error.VI
Error String Size	Size of the Error String parameter.
Function Number	Function number. This information is transmitted through the Get the Last Error.VI
Function String Size	Size of the Function Name parameter.
Error in:	Error cluster input.

- Output:

Driver handle out:	Handle of the driver to use the next function.
Error String	Character string including the error
Function Name	Character string including the name of the function which has generated the error.
Error out:	Error cluster output.

Task:
Returns the character strings of the error and of the function name for the Error Number and Function Name parameters.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.4 Digital input driver

2.4.1  Get 1 digital input information.vi

C equivalent:

b_ADDIDATA_ GetDigitalInputInformation

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Digital input number:	Virtual number of the digital input
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Digital input type:	Type of the selected digital input.

0: Input type not defined.
1: 5V type
16: 12V type
32: 24V type
48: Level configurable type

Digital input interrupt:	Interrupt availability for the selected digital input.
Error out:	Error cluster out.

Task:
Returns information about the digital input channels.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.4.2  Get number of digital inputs.vi

C equivalent:

b_ADDIDATA_GetNumberOfDigitalInputs

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI)
Error in:	Error cluster input

Output:

Driver handle out:	Handle of the driver to use the next function.
Number of channels:	Number of digital input channels in the virtual board.
Error out:	Error cluster output.

Task:
Returns the number of digital input channels.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.4.3 Read 1 digital input.vi

C equivalent:

b_ADDIDATA_Read1DigitalInput

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI)
Channel number:	Virtual index of the selected channel.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver of the driver to use the next function.
Channel status:	Boolean status of the selected channel.
Channel value:	Digital value status of the selected channel.
Error out:	Error cluster output.

Task:
Indicates the status of a digital input. Channel number is the input to be read.
A value is returned through the variable channel status and channel value:
0 (low) or 1 (high).

Return value:
1: No error
0: Error by calling the VI. Use Manage the last error.VI to find the error source.

2.4.4 Read 32 digital inputs.vi

C equivalent:

b_ADDIDATA_Read32DigitalInputs

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
32 bit port index:	Virtual index of the selected 32 bit port to be read.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
32 bit port status:	Boolean status of the selected 32 bit port to be read (array of 32 boolean).
32 bit port value:	Digital status value of the selected 32 bit port to be read.
Error out:	Error cluster output.

Task:
Indicates the status of a 32-bit port.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5 Digital output driver

2.5.1 Get 1 digital output information.vi

C equivalent:

b_ADDIDATA_ GetDigitalOutputInformation

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Digital output number:	Virtual channel index of the selected digital output.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Digital output type:	Type of the selected digital output.
Digital output interrupt:	Interrupt availability of the selected digital output.
Error out:	Error cluster output.

Task:
Returns information about the digital output channels.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.2 Get number of digital outputs.vi

C equivalent:

b_ADDIDATA_GetNumberOfDigitalOutputs

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Number of channels:	Number of digital output channels in the virtual board.
Error out:	Error cluster output.

Task:
Returns the number of available digital output channels.

Return value:
1: no error.
0: error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.3 Set 1 digital output on.vi

C equivalent:

b_ADDIDATA_Set1DigitalOutputOn

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Channel number:	Virtual index of the selected digital output.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Sets the output which was passed through the variable b_Channel.
Setting one output means setting the output to "High".
If you have switched on the digital ouput memory (ON), the selected outputs channels are set to "1". The other channels hold their state.
If you have switched off the digital output memory (OFF), the selected outputs are set to "1". The other channels are reset.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.4 Set 1 digital output off.vi

C equivalent:

b_ADDIDATA_Set1DigitalOutputOff

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Channel number:	Virtual index of the selected digital output.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Returns the number of available digital output channels.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.5 Get 1 digital output status.vi

C equivalent:

b_ADDIDATA_Get1DigitalOutputStatus

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Channel number:	Virtual index of the selected digital output.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Channel status:	Boolean status of the selected digital output.
Channel value:	Digital status value of the selected digital output.
Error out:	Error cluster output.

Task:
Return the status of a digital output. The variable w_Channel passes the output to be read (0 to 31). A value is returned through the variable pb_ChannelStatus: 0 (low) or 1 (high).

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.6 Set 32 digital outputs on.vi

C equivalent:

b_ADDIDATA_Set32DigitalOutputsOn

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
32 bit port index:	Virtual index of the 32 bit port.
32 bit port status:	Boolean status value of the 32 bit port (array of boolean).
32 bit port value:	Digital status value of the 32 bit port.
Boolean or digital value:	if TRUE the 32 bit port status will be used, if FALSE the 32 bit port value will be used.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Sets one or several outputs of a port.
Setting one output means setting the output to "High".
If you have switched on the digital ouput memory (ON), the selected outputs channels are set to "1". The other channels hold their state.
If you have switched off the digital output memory (OFF), the selected outputs are set to "1". The other channels are reset.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.7 Set 32 digital outputs off.vi

C equivalent:

b_ADDIDATA_Set32DigitalOutputsOff

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
32 bit port index:	Virtual index of the selected 32 bit port.
32 bit port status:	Boolean status of the selected 32 bit port (array of boolean).
32 bit port value:	Digital value of the selected 32 bit port.
Boolean or digital value:	if TRUE the 32 bit port status will be used, if FALSE the 32 bit port value will be used.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Resets one or several outputs of a port. Resetting one output means setting the output to "Low" (0). The other channels hold their state.

IMPORTANT!
You can use this function only if the digital output memory is ON.
See Set digital output memory on.VI.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.8 Get 32 digital output status.vi

C equivalent:

b_ADDIDATA_Get32DigitalOutputStatus

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
32 bit port number:	Virtual index of the selected 32 bit port.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
32 bit port status:	Boolean status of the selected 32 bit port (array of boolean).
32 bit port value:	Digital status value of the selected 32 bit port.
Error out:	Error cluster output.

Task:
Returns the status of a 32-bit port.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.9 Set digital output memory on.vi

C equivalent:

b_ADDIDATA_SetDigitalOutputMemoryOn

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Activates the digital output memory. After calling up this function, the outputs you have previously activated with the Set X digital output on.VIs are not reset. You can reset them with the function Set X digital output off.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.5.10 Set digital output memory off.vi

C equivalent:

b_ADDIDATA_SetDigitalOutputMemoryOff

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Deactivates the digital output memory.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6 Timer driver

2.6.1 Get 1 timer information.vi

C equivalent:

b_ADDIDATA_GetTimerInformation

Input:
Driver handle in: Handle of the driver to use the function (from Open the driver.VI).
Timer number: Virtual index of the selected Timer.
Error in: Error cluster input.

Output:
Driver handle out: Handle of the driver to use the next function.

Information cluster:

Timer mode: Modes available for the timer

Timer mode description

Selected mode	**Mode description**	***dw_ReloadValue* description**	**Hardware gate input action**
0	Mode 0 is typically used for event counting. After the initialisation, OUT is initially low, and remains low until the counter reaches 0. OUT goes high and remains high until a new count is written. See "b_ADDIDATA_WriteTimerValue" function.	Start counting value	Hardware gate
1	Mode 1 is similar to mode 0 except for the gate input action. The gate input is not used to enable or disable the timer (like in Mode 0), but to trigger it.	Start counting value	Hardware trigger
2	This mode functions like a divide-by-ul_ReloadValue counter. It is used to generate a real time clock interrupt. OUT is initially high after the initialisation. When the initial count has decremented to 1, OUT goes low for one CLK pulse. OUT then goes high again, the counter reloads the initial count (ul_ReloadValue) and the process is repeated. This action can generate an interrupt. See function "b_ADDIDATA_SetFunctionalityIntRoutine" and	Divider factor	Hardware gate

	"b_ADDIDATA_EnableDisableTimerInterrupt"		
3	Mode 3 is typically used for baud rate generation. This mode is similar to mode 2 except for the duty cycle of OUT. OUT will initially be high after the initialisation. When half the initial count (ul_ReloadValue) has expired, OUT goes low for the remainder of the count. The mode is periodic; the sequence above is repeated indefinitely.	Divider factor	Hardware gate
4	OUT is initially high after the initialisation. When the initial count expires, OUT will go low for one CLK pulse and then go high again. The counting sequence is triggered by writing a new value. See "i_ADDIDATA_WriteTimerValue" function. If a new count is written during counting, it will be loaded on the next CLK pulse.	Start counting value	Hardware gate
5	Mode 5 is similar to mode 4 except for the gate input action. The gate input is not used to enable or disable the timer, but to trigger it.	Start counting value	Hardware trigger

Value (Dec)	**Value (hex)**	**Selectable modes**
1	1	Mode 0 available
2	2	Mode 1 available
3	3	Mode 0, 1 available
4	4	Mode 2 available
5	5	Mode 0, 2 available
6	6	Mode 1, 2 available
7	7	Mode 0, 1, 2 available
8	8	Mode 3 available
9	9	Mode 0, 3 available
10	A	Mode 1, 3 available
11	B	Mode 0,1, 3 available
12	C	Mode 2, 3 available
13	D	Mode 0, 2, 3 available
14	E	Mode 1, 2, 3 available
15	F	Mode 0, 1, 2, 3 available
16	10	Mode 4 available
17	11	Mode 0, 4 available
18	12	Mode 1, 4 available
19	13	Mode 0, 1, 4 available
20	14	Mode 2, 4 available
21	15	Mode 0, 2, 4 available
22	16	Mode 1, 2, 4 available
23	17	Mode 0, 1, 2, 4 available

24	18	Mode 3, 4 available
25	19	Mode 0, 3, 4 available
26	1A	Mode 1, 3, 4 available
27	1B	Mode 0, 1, 3, 4 available
28	1C	Mode 2, 3, 4 available
29	1D	Mode 0, 2, 3, 4 available
30	1E	Mode 1, 2, 3, 4 available
31	1F	Mode 0, 1, 2, 3, 4 available
32	20	Mode 5 available
33	21	Mode 0, 5 available
34	22	Mode 1, 5 available
35	23	Mode 0, 1, 5 available
36	24	Mode 2, 5 available
37	25	Mode 0, 2, 5 available
38	26	Mode 1, 2, 5 available
39	27	Mode 0, 1, 2, 5 available
40	28	Mode 3, 5 available
41	29	Mode 0, 3, 5 available
42	2A	Mode 1, 3, 5 available
43	2B	Mode 0,1, 3, 5 available
44	2C	Mode 2, 3, 5 available
45	2D	Mode 0, 2, 3, 5 available
46	2E	Mode 1, 2, 3, 5 available
47	2F	Mode 0, 1, 2, 3, 5 available
48	30	Mode 4, 5 available
49	31	Mode 0, 4, 5 available
50	32	Mode 1, 4, 5 available
51	33	Mode 0, 1, 4, 5 available
52	34	Mode 2, 4, 5 available
53	35	Mode 0, 2, 4, 5 available
54	36	Mode 1, 2, 4, 5 available
55	37	Mode 0, 1, 2, 4, 5 available
56	38	Mode 3, 4, 5 available
57	39	Mode 0, 3, 4, 5 available
58	3A	Mode 1, 3, 4, 5 available
59	3B	Mode 0, 1, 3, 4, 5 available
60	3C	Mode 2, 3, 4, 5 available
61	3D	Mode 0, 2, 3, 4, 5 available
62	3E	Mode 1, 2, 3, 4, 5 available
63	3F	Mode 0, 1, 2, 3, 4, 5 available

Timer time unit: Time units available.

1: ns
2: micro s
3: ns and micro s
4: ms
5: ns and ms
6: micro s and ms
7: ns, micro s and ms
8: s
9: ns and s

10: micro s and s
11: ns, ms and s
12: ms and s
13: ns, ms and s
14: micro s, ms and s
15: ns, micro s, ms and s

Timer time step: Possible time steps for the timer

Resolution: Resolution for the timer.

Hardware gate available:
0: Hardware gate not available.
1: Hardware gate available.

Hardware trigger available
0: Hardware trigger not available.
1: Hardware trigger available.

Output available
0: Hardware output not available
1: Hardware output available

Error out: Error cluster output.

Task:
Returns the timer modes, the time units, the time steps, the hardware gate and trigger available and the resolution which can be used for the selected timer.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.2 Get number of timers.vi

C equivalent:

b_ADDIDATA_GetNumberOfTimers

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Number of timers:	Number of timers in the virtual board.
Error out:	Error cluster output.

Task:
Returns the number of timers available on the board.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.3 Init 1 timer.vi

C equivalent:

b_ADDIDATA_InitTimer

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Timer mode: Selection of the timer mode.

Timer mode description

Selected mode	**Mode description**	***dw_ReloadValue* description**	**Hardware gate input action**
0	Mode 0 is typically used for event counting. After the initialisation, OUT is initially low, and remains low until the counter reaches 0. OUT goes high and remains high until a new count is written. See "b_ADDIDATA_WriteTimerValue" function.	Start counting value	Hardware gate
1	Mode 1 is similar to mode 0 except for the gate input action. The gate input is not used to enable or disable the timer (like in Mode 0), but to trigger it.	Start counting value	Hardware trigger
2	This mode functions like a divide-by-ul_ReloadValue counter. It is used to generate a real time clock interrupt. OUT is initially high after the initialisation. When the initial count has decremented to 1, OUT goes low for one CLK pulse. OUT then goes high again, the counter reloads the initial count (ul_ReloadValue) and the process is repeated. This action can generate an interrupt. See function "b_ADDIDATA_SetFunctionalityIntRoutine" and "b_ADDIDATA_EnableDisableTimerInterrupt"	Divider factor	Hardware gate
3	Mode 3 is typically used for baud rate generation. This mode is similar to mode 2 except for the duty cycle of OUT. OUT will initially be high after the initialisation. When half the initial count (ul_ReloadValue) has expired, OUT goes	Divider factor	Hardware gate

	low for the remainder of the count. The mode is periodic; the sequence above is repeated indefinitely.		
4	OUT is initially high after the initialisation. When the initial count expires, OUT will go low for one CLK pulse and then go high again. The counting sequence is triggered by writing a new value. See "i_ADDIDATA_WriteTimerValue" function. If a new count is written during counting, it will be loaded on the next CLK pulse.	Start counting value	Hardware gate
5	Mode 5 is similar to mode 4 except for the gate input action. The gate input is not used to enable or disable the timer, but to trigger it.	Start counting value	Hardware trigger

Timer time unit: Selection of the time unit
0: ns
1: micros
2: ms
3: s
4: min

Reload value: Start value or time interval (depends from the used ADDI-DATA board). See Get 1 timer information.VI

Error in: Error cluster input.

Output:
Driver handle out: Handle of the driver to use the next function.
Error out: Error cluster output.

Task:
Initialises the timer.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.4 Start 1 timer.vi

C equivalent:

b_ADDIDATA_StartTimer

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Timer number:	Virtual index of the selected timer.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Starts the timer.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.5 Stop 1 timer.vi

C equivalent:

b_ADDIDATA_StopTimer

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Timer number:	Virtual index of the selected timer.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Stops the timer.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.6 Trigger 1 timer.vi

C equivalent:

b_ADDIDATA_TriggerTimer

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Timer number:	Virtual index of the selected timer.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Triggers the timer.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.7 Release 1 timer.vi

C equivalent:

b_ADDIDATA_ReleaseTimer

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Timer number:	Virtual index of the selected timer.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Releases the timer for a new initialisation.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.8 Read 1 timer status.vi

C equivalent:

b_ADDIDATA_ReadTimerStatus

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Timer number:	Virtual index of the selected timer.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.

Timer status
- 0: Timer ran down or did not start
- 1: Timer is running

Timer status LEDs lights when value = 1.

Software trigger status
- 0: Software trigger did not occur
- 1: Software trigger occurred

When the status of the software trigger is read, it is automatically reset. By the next calling of the parameter, 0 is returned if no trigger occurred during this period.

Timer software trigger status LEDs lights when value = 1.

Hardware trigger status
- 0: Hardware trigger did not occur
- 1: Hardware trigger occurred

Timer hardware trigger status LEDs lights when value = 1.

Error out: Error cluster output.

Task:
Returns the status of the timer, the software trigger and the hardware trigger.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.6.9 Read 1 timer value.vi

C equivalent:

b_ADDIDATA_ReadTimerValue

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Timer number:	Virtual index of the selected timer.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Timer value:	Value of the selected timer.
Error out:	Error cluster output.

Task:
Reads the timer value.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7 Counter driver

2.7.1 Get 1 counter information.vi

C equivalent:

b_ADDIDATA_GetCounterInformation

Input:
Driver handle in: Handle of the driver to use the function (from Open the driver.VI).
Counter number: Virtual index of the selected counter.
Error in: Error cluster input.

Output:
Driver handle out: Handle of the driver to use the next function.
Infomation cluster:

Resolution: Resolution of the selected counter.

Interrupt availablity:
0: Counter cannot generate an interrupt
1: Counter can generate an interrupt

Input level selection
1: Counter only counts low pulses
2: Counter only counts high pulses
3: Counter can count low and high pulses

Hardware gate available
0: Hardware gate not available.
1: Hardware gate available.

Hardware trigger available
0: Hardware trigger not available.
1: Hardware trigger available.

Output available
0: Hardware output not available
1: Hardware output available

Up down selection
0: Up/Down selection not available and counter counts down
1: Up/Down selection not available and counter counts upwards
2: Up/Down selection available

Error out: Error cluster output.

Task:
Returns the hardware gate availablity, the hardware trigger availablity, the up/down selection and the resolution which can be used for the selected counter.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.2 Get number of counters.vi

C equivalent:

b_ADDIDATA_GetNumberOfCounters

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Number of counters:	Number of counter in the virtual board.
Error out:	Error cluster output.

Task:
Returns the number of counters available on the virtual board.

Return value:
1: No error.
0: error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.3 Init 1 counter.vi

C equivalent:

b_ADDIDATA_InitCounter

Input:
Driver handle in: Handle of the driver to use the function (from Open the driver.VI).
Counter number: Virtual index of the selected counter
The first counter begins from 0.
Level selection:
1: Counter counts low levels
2: Counter counts high levels
3: Counter counts high and low levels
This parameter has no influence if the selection of the counter direction is not possible. See the Get counter information.VI

Reload value: Reload or overflow value (depends from the used ADDI-DATA board). See Set counter direction.VI

Error in: Error cluster input.

Output:
Driver handle out: Handle of the driver to use the next function.
Error out: Error cluster output.

Task:
Initialises the counter.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.4 Start 1 counter.vi

C equivalent:

b_ADDIDATA_StartCounter

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Counter number:	Virtual index of the selected counter.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Starts the counter.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.5 Stop 1 counter.vi

C equivalent:

b_ADDIDATA_StopCounter

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Counter number:	Virtual index of the selected counter.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Stops the counter.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.6 Trigger 1 counter.vi

C equivalent:

b_ADDIDATA_TriggerCounter

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Counter number:	Virtual index of the selected counter.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Triggers the counter.
If you have selected 0 for the Set counter direction.VI, the counter reloads the start value.
If you have selected 1 for the Set counter direction, the counter is cleared.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.7 Release 1 counter.vi

C equivalent:

b_ADDIDATA_ReleaseCounter

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Counter number:	Virtual index of the selected counter
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Releases the counter for a new initialisation.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.8 Read 1 counter status.vi

C equivalent:

b_ADDIDATA_ReadCounterStatus

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Counter number:	Virtual index of the selected counter.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.

Status information cluster:

Counter status
0: Counter ran down or did not start
1: Counter is running

Counter status LEDs lights when value = 1.

Software trigger status
0: Software trigger did not occur
1: Software trigger occurred

Software trigger status LEDs lights when = 1.

When the status of the software trigger is read, it is automatically reset. By the next calling of the parameter, 0 is returned if no trigger occurred during this period.

Hardware trigger status
0: Hardware trigger did not occur
1: Hardware trigger occurred

Hardware trigger status LEDs lights when = 1.

Software clear status
0: Software clear did not occur
1: Software clear occurred

Software clear status LEDs lights when = 1.

Error out: Error cluster output.

Task:

Returns the status of the counter, of the software trigger, of the hardware trigger and of the software clear.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.9 Read 1 counter value.vi

C equivalent:

b_ADDIDATA_ReadCounterValue

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Counter number:	Virtual index of the selected counter.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Counter value:	Current Value of the selected counter.
Error out:	Error cluster output.

Task:
Reads the counter value.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.7.10 Set 1 counter direction.vi

C equivalent:

b_ADDIDATA_SetCounterDirection

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Counter number: Virtual index of the selected counter
The first counter begins from 0.

Direction selection:
1: Selects the counter for up-counting
0: Selects the counter for down-counting

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.
Error out: Error cluster output.

Task:

Sets the counting direction (upward or downward)

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8 Watchdog driver

2.8.1 Get 1 watchdog information.vi

C equivalent:

b_ADDIDATA_GetWatchdogInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).
Watchdog number: Virtual index of the selected watchdog.
Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.
Timer time unit: Time units available.

1: ns
2: micro s
3: ns and micro s
4: ms
5: ns and ms
6: micro s and ms
7: ns and micro s and ms
8: s
9: ns and s
10: micro s and s
11: ns and ms and s
12: ms and s
13: ns and ms and s
14: micro s and ms and s
15: ns and micro s and ms and s

Watchdog time step: Possible time steps for the watchdog

Resolution: Resolution for the watchdog in bits.

Hardware gate available:
0: Hardware gate not available.
1: Hardware gate available.

Hardware trigger available
0: Hardware trigger not available.
1: Hardware trigger available.

Warning relay available
0: Warning relay not available
1: Warning relay available

Reset relay available
- 0: Reset relay not available
- 1: Reset relay available

Warning delay available
- 0: Waiting time between the warning relay and the activation of the reset relay is not available
- 1: Waiting time between the warning relay and the activation of the reset relay is available.

Error out: Error cluster output.

Task:
Returns the time units (Watchdog time unit), the time steps (Watchdog time step) and the resolution (Resolution) which can be used for the selected timer (Timer number).

Return value:
1: no error.
0: error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8.2 Get number of watchdogs.vi

C equivalent:

b_ADDIDATA_GetNumberOfWatchdogs

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Number of watchdogs: Number of watchdogs in the virtual board.

Watchdog type: Array of the watchdog type.
Give the type of each watchdog. See table 2-1
Watchdog type [0]: Type of the first watchdog
Watchdog type [1]: Type of the 2nd watchdog
...
Watchdog type [Number of watchdog - 1]: Type of the last watchdog

Value Description:
1: Watchdog for the digital outputs
After running-down of the watchdog, the outputs are reset.
2: Watchdog for the analog outputs
After running-down of the watchdog, the outputs are reset.
3: Watchdog for the digital/analog outputs.
After running-down of the watchdog, the digital/analog outputs are reset.
4: System watchdog. This watchdog can be used for the system control.

Error out: Error cluster output.

Task:

Returns the number of watchdogs and the type of each watchdog.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8.3 Init 1 watchdog.vi

C equivalent:

b_ADDIDATA_InitWatchdog

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Watchdog number: Virtual index of the selected watchdog. The first watchdog begins from 0.

Delay time unit: Selection of the time unit

0: ns
1: micro s
2: ms
3: s
4: min

Delay value: Start value or watchdog time (depends from the used ADDI-DATA board). See Get watchdog information.VI.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Initialises the watchdog.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8.4 Start 1 watchdog.vi

C equivalent:

b_ADDIDATA_StartWatchdog

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Watchdog number:	Virtual index of the selected watchdog.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Starts the watchdog.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8.5 Stop 1 watchdog.vi

C equivalent:

b_ADDIDATA_StopWatchdog

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Watchdog number:	Virtual index of the selected watchdog.
Error in:	Error cluster input.

Output:
Driver handle out: Handle of the driver to use the next function.
Error out: Error cluster output.

Task:
Stops the watchdog.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8.6 Trigger 1 watchdog.vi

C equivalent:

b_ADDIDATA_TriggerWatchdog

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Watchdog number:	Virtual index of the selected watchdog.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Triggers the watchdog.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8.7 Release 1 watchdog.vi

C equivalent:

b_ADDIDATA_ReleaseWatchdog

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Watchdog number:	Virtual index of the selected watchdog.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:
Releases the watchdog for a new initialisation.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.8.8 Read 1 watchdog status.vi

C equivalent:

b_ADDIDATA_ReadWatchdogStatus

Input:
Driver handle in: Handle of the driver to use the function (from Open the driver.VI).
Watchdog number: Virtual index of the selected watchdog.
Error in: Error cluster input.

Output:
Driver handle out: Handle of the driver to use the next function.

Watchdog status
0: Watchdog has run down or did not start
1: Watchdog is running

Watchdog status LEDs lights when value = 1.

Software trigger status
0: Software trigger did not occur
1: Software trigger occurred
When the status of the software trigger is read, it is automatically reset. By the next calling of the parameter, 0 is returned if no trigger occurred during this period.

Watchdog software trigger status LEDs lights when value = 1.

Hardware trigger status
0: Hardware trigger did not occur
1: Hardware trigger occurred

Watchdog hardware trigger status LEDs lights when value = 1.

Error out: Error cluster output.

Task:
Returns the status of the watchdog, the software trigger and the hardware trigger.

Return value:
1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9 Analog input driver

2.9.1  Get number of analog input module.vi

C equivalent:

b_ADDIDATA_GetNumberOfAnalogInputModules

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Number of modules: Returns the number of analog input modules.

Error out: Error cluster output.

Task:

Returns the number of analog input modules.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.2  Get 1 analog input module number.vi

C equivalent:

b_ADDIDATA_GetAnalogInputModuleNumber

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Channel number: Virtual index of the selected channel.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Module number: Virtual index of the corresponding module.

Error out: Error cluster output.

Task:

Returns the analog input module virtual index (Module number) from the selected channel (Channel number)

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.3  Get 1 analog input module general information.vi

C equivalent:

b_ADDIDATA_GetAnalogInputModuleGeneralInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected Module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

Inputs resolution: Returns the input resolution
8: 8-bit resolution
16: 16-bit resolution, ...

Unipolar bipolar configurable:
0: Unipolar/Bipolar hardware configurable
1: Unipolar/Bipolar software configurable

Unipolar available:
0: Unipolar mode not available
1: Unipolar mode available

Bipolar available:
0: Bipolar mode not available
1: Bipolar mode available

Single difference selected:
0: Single mode selected
1: Difference mode selected

AC available:
0: AC available
1: AC not available

DC available:
0: DC available

1: DC not available

Auto calibration:
0: Auto calibration not available
1: Auto calibration available

Max input voltage:
Return the maximal input voltage value in V or A

Conversion calc type:
0: Binary type
(XX000,XX001,XX010,XX011)
1: Multiple type (60,120,240,480, ...)

Conversion unit type:
0: Time unity (ns,s,ms,s, ...)
1: Frequency unity (MHz, KHz, Hz, ...)

Available conversion unit:

For time unity:

D0: 0: ns not available
1: ns available
D1: 0: s not available
1: s available
D2: 0: ms not available
1: ms available
D3: 0: s not available
1: s available

For frequency unity:

D0: 0: MHz not available
1: MHz available
D1: 0: KHz not available
1: KHz available
D2: 0: Hz not available
1: Hz available
D3: 0: mHz not available
1: mHz available

Conversion resolution: Convert time resolution
8: 8-bit resolution
16: 16-bit resolution, ...

Min conversion time: Minimum converting time.
7000: 7000 (ns)
10000: 10000 (ns), ...

Conversion step: Conversion time steps

20: 20 steps
50: 50 steps, ...

Single acquisition:
0: Single acquisition not available
1: Single acquisition available

Auto refresh acquisition:
0: Auto refresh acquisition not available
1: Auto refresh acquisition available

Scan acquisition:
0: Scan acquisition not available
1: Scan acquisition available

Sequence acquisition:
0: Sequence acquisition not available
1: Sequence acquisition available

First channel number: Returns the virtual index of the first channel number
Last channel number: Returns the virtual index of the last channel number

Error out: Error cluster output.

Task:

Returns general information about the analog input module (input resolution, convert time, available acquisition modes)

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.4  Init 1 analog input channel.vi

C equivalent:

b_ADDIDATA_InitAnalogInput

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Channel number: Virtual index of the selected analog input to initialise.

Initialisation cluster:

Gain: Gain of the channel.

Polarity: Polarity of the selected channel
1: ADDIDATA_UNIPOLAR for unipolar and
2 : ADDIDATA_BIPOLAR for bipolar

Offset range: Offset range of the channel.
Not used for the moment. Set it to 0.

Coupling: Select the coupling:
1 : ADDIDATA_AC_COUPLING : Alternative coupling
0 : ADDIDATA_DC_COUPLING : Direct coupling

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Initialises the selected analog input channel.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.5  Release 1 analog input channel.vi

C equivalent:

b_ADDIDATA_ReleaseAnalogInput

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Channel number: Virtual index of the selected analog input to release.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Releases the analog input logic for the selected channel number to allow a new initialisation.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.6  Read more analog input channels.vi

C equivalent:

b_ADDIDATA_ReadMoreAnalogInputs

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Initialisation cluster:

First channel: Virtual index of the first channel to be read

Last channel: Virtual index of the last channel to be read

Converting time: Converting time.
(Refer to table)

Converting time unit: Determines the unit of the converting time.
(Refer to table)

ConvertingTimeUnit	Unit selection	Converting time	Conversion time
0	ns / MHz	10	10 ns / MHz
		20	20 ns / MHz
		300	300 ns / MHz
		1000	1000 ns / MHz
		22222	22222 ns / MHz
1	s / KHz	10	10 s /KHz
		20	20 s / KHz
		300	300 s / KHz
		1000	1000 s / KHz
		22222	22222 s / KHz
2	ms / Hz	10	10 ms/ Hz
		20	20 ms / Hz
		300	300 ms / Hz
		1000	1000 ms / Hz
		22222	22222 ms / Hz
3	s / mHz	10	10 s / mHz
		20	20 s / mHz
		300	300 s / mHz
		1000	1000 s / mHz
		22222	22222 s / mHz

Interrupt flag:

ADDIDATA_DISABLE: No interrupt is generated after the last conversion and the variable Channel array values and calibration values returns the digital value.

ADDIDATA_ENABLE: An interrupt is generated after the last conversion. The digital value of all selected channels is returned through the interrupt function. Refer to the chapter "Interrupt"

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Channel array values and calibration values:

Returns the digital value of all selected channels

Channel array values and calibration values[0] = digital value of the first channel
Channel array values and calibration values[1] = digital value of the calibration offset (if
available)
Channel array values and calibration values[2] = digital value of the calibration gain (if
available)
Channel array values and calibration values[3] = digital value of the next channel
Channel array values and calibration values[4] = digital value of the calibration offset (if
available)
Channel array values and calibration values[5] = digital value of the calibration gain (if
available)

Error out: Error cluster output.

Task:

Returns the digital value (Channel array values and calibration values) of all selected channels (First channel number, Last channel number).
To obtain the real analog input value, you must call up the
Convert more digital to real analog value VI.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.7  Get 1 analog input module single acquisition information.vi

C equivalent:

b_ADDIDATA_GetAnalogInputModuleSingleAcquisitionInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected Module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

Interrupt:
0: No interrupt can by generated
1: Interrupt can by generated

Software trigger:
0: Software trigger not available
1: Software trigger available

Hardware trigger:
0: Hardware trigger not available
1: Hardware trigger available

Hardware gate:
0: Hardware gate not available
1: Hardware gate available

Number of gain:
Returns the number of gain values available

Gain available:
Array which defines the available gain values

Error out: Error cluster output.

Task:

Returns information about the analog input module for the single acquisition mode (Read more analog inputs.VI)

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.8  Convert more digital to real analog values.vi

C equivalent:

b_ADDIDATA_ConvertMoreDigitalToRealAnalogValues

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

First channel: Virtual index of the first channel to be convert

Last channel: Virtual index of the first channel to be convert

Digital values array:

Digital value of the analog input (composed of the digital value of the channel, of the calibration offset and of the calibration gain if available)

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Real values array:
Returns the real analog value of the input channel

Error out: Error cluster output.

Task:

Converts the digital value array into a real analog value array for the selected channels.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.9  Init analog input SCAN acquisition.vi

C equivalent:

b_ADDIDATA_InitAnalogInputSCANAcquisition

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Initialisation cluster:

First channel: Virtual index of the first channel to be read

Last channel: Virtual index of the first channel to be read

Converting time: Converting time.
Refer to table

Converting time unit: Determines the unit of the converting time.
Refer to table

ConvertingTimeUnit	Unit selection	Converting time	Conversion time
0	ns / MHz	10	10 ns / MHz
		20	20 ns / MHz
		300	300 ns / MHz
		1000	1000 ns / MHz
		22222	22222 ns / MHz
1	s / KHz	10	10 s /KHz
		20	20 s / KHz
		300	300 s / KHz
		1000	1000 s / KHz
		22222	22222 s / KHz
2	ms / Hz	10	10 ms/ Hz
		20	20 ms / Hz
		300	300 ms / Hz
		1000	1000 ms / Hz
		22222	22222 ms / Hz
3	s / mHz	10	10 s / mHz
		20	20 s / mHz
		300	300 s / mHz
		1000	1000 s / mHz
		22222	22222 s / mHz

Delay time mode: Delay mode selection
Refer to table

Delay time mode	Mode description
0	The delay between two SCANs is not used
1	The delay between two SCAN cycles is used. The delay is started after the first acquisition. After this delay a new SCAN cycle is started
2	The delay between two SCAN cycles is used. The delay is started after the last SCAN channel acquisition. After this delay a new SCAN cycle is started

Delay time unit:

Delay time unit selection

For time unit:
0: ns selected
1: s selected
2: ms selected
3: s selected

For frequency unity:
0: MHz selected
1: KHz selected
2: Hz selected
3: mHz selected

Delay time: Delay time

SCAN counter: Define the number of SCAN cycles if the SCAN mode is setting to 1.
Refer to table

SCAN mode:
Refer to table

SCAN mode	SCAN counter	SCAN description
0	Not used	Only one SCAN is started after calling up the "StartAnalogInputSCAN.VI"
1	Determines the number of SCAN	A predefined number of SCANs is started after up the function "StartAnalogInputSCAN.VI"
2	Not used	An undefined number of SCANs is started after up the function

"StartAnalogInputSCAN.VI"

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

SCAN handle: Handle of the initialised SCAN. This handle is used for all SCAN functions.

Error out: Error cluster output.

Task:

Initialises the analogue input acquisition SCAN.
This can be combined with the hardware or software trigger.
See the Enable disable analog input hardware trigger.VI and Enable disable analog input software trigger.VI

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.10  Start 1 analog input SCAN.vi

C equivalent:

b_ADDIDATA_StartAnalogInputSCAN

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

SCAN handle: Handle of the selected SCAN to start. This handle is returned by "InitAnalogInputSCANAcquisition.VI"

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Starts the selected SCAN

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.11  Stop analog input SCAN.vi

C equivalent:

b_ADDIDATA_StopAnalogInputSCAN

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

SCAN handle: Handle of the selected SCAN to stop. This handle is returned by "InitAnalogInputSCANAcquisition.VI"

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Stops the selected SCAN

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.12  Close analog input SCAN.vi

C equivalent:

b_ADDIDATA_CloseAnalogInputSCAN

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

SCAN handle: Handle of the selected SCAN to stop. This handle is returned by "InitAnalogInputSCANAcquisition.VI"

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Close the selected SCAN and releases the SCAN handle.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.13  Get analog input SCAN status.vi

C equivalent:

b_ADDIDATA_GetAnalogInputSCANStatus

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

SCAN handle: Handle of the selected SCAN to start. This handle is returned by "InitAnalogInputSCANAcquisition.VI"

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

SCAN status: Status of the SCAN.
0: SCAN not started
1: SCAN started
2: SCAN completed
3: SCAN completed and new SCAN started

Error out: Error cluster output.

Task:

Returns the status (SCAN status) of the selected SCAN (SCAN handle).

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.14  Convert digital to real analog value SCAN.vi

C equivalent:

b_ADDIDATA_ConvertDigitalToRealAnalogValueSCAN

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

SCAN handle: This handle is returned by the "InitAnalogInputSCAN.VI"

Digital values array:

Digital value of the analog input (composed of the digital value of the channel, of the calibration offset and of the calibration gain if available)

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Real values array:
Returns the real analog value of the input channel

Error out: Error cluster output.

Task:

Converts the digital value array of the analog input into a real analog value array.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.15  Get analog input module SCAN information.vi

C equivalent:

b_ADDIDATA_GetAnalogInputModuleSCANInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected Module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

SCAN configurable:
0: SCAN fixed. User must pass the first and last channel from the selected module
1: SCAN configurable. User is free to define the first and last channel

Software trigger:
0: Software trigger not available
1: Software trigger available

Hardware trigger:
0: Hardware trigger not available
1: Hardware trigger available

Hardware gate:
0: Hardware gate not available
1: Hardware gate available

Common gain:
Defines if the gain can be different for each input
0: Gain is common on each input
1: Gain can be different for each input.

Common polarity:
Defines if the polarity can be different for each input
0: Polarity is common on each input
1: Polarity can be different for each input.

Number of gain:
Returns the number of gain values available

Gain available[255]:
Array which defines the available gain value

Delay time configurable:
0: Delay after each sequence not available
1: Delay after each sequence available

Delay available mode:
D0: 0: Mode 1 not available
1: Mode 1 available
D1: 0: Mode 2 not available
1: Mode 2 available

Delay calc type:
0: Binary type (XX000, XX001, XX010)
1: Multiple type (60, 120, 240, 480, 960, ...)

Delay time unit type:
0: Time unit (ns, s, ms, s, ...)
1: Frequency unit (MHz, kHz, Hz, mHz, ...)

Delay time unit:

For time unit:
D0: 0: ns not available
1: ns available
D1: 0: s not available
1: s available
D2: 0: ms not available
1: ms available
D3: 0: s not available
1 : s available

For frequency unity:

D0: 0: MHz not available
1: MHz available
D1: 0: KHz not available
1: KHz available
D2: 0: Hz not available
1: Hz available
D3: 0: mHz not available
1: mHz available

Delay time resolution: Delay time resolution
8: 8-bit resolution
16: 16-bit resolution, ...

Min delay time: Minimum delay time.
  7000: 7000(ns)
  10000: 10000(ns), ...

Delay time step: Conversion delay time steps
  20: 20 steps
  50: 50 steps, ...

Acquisition mode:
  XX1: The acquisition can work in Single mode
  X11: The acquisition can work in continuous mode
  1XX: The acquisition can work in counting mode

Max number of acquisition:
  Returns the max number of acquisition for the counting acquisition SCAN mode

Error out: Error cluster output.

Task:

Returns information about the analog input module for the SCAN acquisition mode (Init analog input SCAN acquisition.VI and Start analog input SCAN.VI). Into the SCAN mode the user define the first and last channel to acquire. After each SCAN an interrupt is generated and the user receive the analog input values.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.16  Start analog input auto refresh.vi

C equivalent:

b_ADDIDATA_StartAnalogInputAutoRefresh

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected analog input module to start in auto refresh mode.

Converting time: Converting time.
Refer to table

Converting time unit: Determines the unit of the converting time.
Refer to table

ConvertingTimeUnit	Unit selection	Converting time	Conversion time
0	ns / MHz	10	10 ns / MHz
		20	20 ns / MHz
		300	300 ns / MHz
		1000	1000 ns / MHz
		22222	22222 ns / MHz
1	s / KHz	10	10 s /KHz
		20	20 s / KHz
		300	300 s / KHz
		1000	1000 s / KHz
		22222	22222 s / KHz
2	ms / Hz	10	10 ms/ Hz
		20	20 ms / Hz
		300	300 ms / Hz
		1000	1000 ms / Hz
		22222	22222 ms / Hz
3	s / mHz	10	10 s / mHz
		20	20 s / mHz
		300	300 s / mHz
		1000	1000 s / mHz
		22222	22222 s / mHz

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Starts the auto refresh acquisition on the selected module.

Converting time and Converting time unit determines the converting time.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.17  Stop analog input auto refresh.vi

C equivalent:

b_ADDIDATA_StopAnalogInputAutoRefresh

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected analog input module to start in auto refresh mode.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Stops the auto refresh acquisition on the selected module.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.18  Read 1 analog input auto refresh value.vi

C equivalent:

b_ADDIDATA_Read1AnalogInputAutoRefreshValue

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Channel number: Virtual index of the selected channel to be read

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Channel value and calibration value: Returns the digital value of all selected channel

Error out: Error cluster output.

Task:

Returns the digital value (Channel value and calibration value) of the selected channel (Channel number).

To obtain the real analog input value, you must call up the

Convert more digital to real analog value VI.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.19  Read analog input auto refresh counter value.vi

C equivalent:

b_ADDIDATA_ReadAnalogInputAutoRefreshCounterValue

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected module of the counter which has to be read.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Counter value: Returns the value of the counter of the selected module.

Error out: Error cluster output.

Task:

Returns the counter value (Counter value) of the counter of the selected module (Module number).

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.20  Get 1 analog input module auto refresh information.vi

C equivalent:

b_ADDIDATA_GetAnalogInputModuleAutoRefreshInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected Module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

Interrupt:
0: No interrupt can by generated
1: Interrupt can by generated

Software trigger:
0: Software trigger not available
1: Software trigger available

Hardware trigger:
0: Hardware trigger not available
1: Hardware trigger available

Hardware gate:
0: Hardware gate not available
1: Hardware gate available

Access mode: Auto refresh buffer access mode
8: 8-bit access mode
16: 16-bit access mode
32: 32-bit access mode

Common gain: Defines if the gain can be different for each input
0: Gain is common on each input
1: Gain can be different for each input.

Common polarity: Defines if the polarity can be different for each input

0: Polarity is common on each input
1: Polarity can be different for each input.

Number of gain: Returns the number of gain values available

Gain available[255]: Array which defines the available gain value

Error out: Error cluster output.

Task:

Returns information about the analog input module for the auto refresh acquisition mode (Start analog input auto refresh.VI). The auto refresh mode acquires continually the analog inputs and gives the values via a memory space.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.21  Init analog input sequence acquisition.vi

C equivalent:

b_ADDIDATA_InitAnalogInputSequenceAcquisition

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Number of channel: Number of channel in the sequence.

Sequence channel array: Array of the virtual index of the channel to be converted.

Initialisation cluster:

Converting time: Converting time.
Refer to table

Converting time unit: Determines the unit of the converting time.
Refer to table

ConvertingTimeUnit	Unit selection	Converting time	Conversion time
0	ns / MHz	10	10 ns / MHz
		20	20 ns / MHz
		300	300 ns / MHz
		1000	1000 ns / MHz
		22222	22222 ns / MHz
1	s / KHz	10	10 s /KHz
		20	20 s / KHz
		300	300 s / KHz
		1000	1000 s / KHz
		22222	22222 s / KHz
2	ms / Hz	10	10 ms/ Hz
		20	20 ms / Hz
		300	300 ms / Hz
		1000	1000 ms / Hz
		22222	22222 ms / Hz
3	s / mHz	10	10 s / mHz
		20	20 s / mHz
		300	300 s / mHz
		1000	1000 s / mHz
		22222	22222 s / mHz

Delay time mode: Delay mode selection
Refer to table

Delay time mode	Mode description

0	The delay between two SCANs is not used
1	The delay between two SCAN cycles is used. The delay is started after the first acquisition. After this delay a new SCAN cycle is started
2	The delay between two SCAN cycles is used. The delay is started after the last SCAN channel acquisition. After this delay a new SCAN cycle is started

Delay time unit:

Delay time unit selection

For time unit:
0: ns selected
1: s selected
2: ms selected
3: s selected

For frequency unit:
0: MHz selected
1: KHz selected
2: Hz selected
3: mHz selected

Delay time: Delay time

Sequence counter: Number of sequence cycles.

Interrupt sequence counter: Number of sequences before an interrupt is generated.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Sequence handle: Handle of the initialised sequence. This handle is used for all sequence functions.

Error out: Error cluster output.

Task:

Initialises the analogue input acquisition sequence.
Sequence counter defines the number of sequences.
A sequence is composed of the channels that are defined in the sequence channel array.
Interrupt sequence counter defines the number of sequences before an interrupt is generated.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.Q8

2.9.22  Start analog input sequence.vi

C equivalent:

b_ADDIDATA_StartAnalogInputSequenceAcquisition

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Sequence handle:	This handle is returned by the "Init analog input sequence acquisition.VI".
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:

Starts the selected sequence.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.23  Stop analog input sequence.vi

C equivalent:

b_ADDIDATA_StopAnalogInputSequenceAcquisition

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Sequence handle:	This handle is returned by the "Init analog input sequence acquisition.VI".
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:

Stop the selected sequence.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.24  Release 1 analog input sequence.vi

C equivalent:

b_ADDIDATA_ReleaseAnalogInputSequenceAcquisition

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Sequence handle:	This handle is returned by the "Init analog input sequence acquisition.VI".
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.
Error out:	Error cluster output.

Task:

Releases the selected sequence (Sequence handle) and lets the resources free for another process.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.25  Get analog input sequence handle status.vi

C equivalent:

b_ADDIDATA_GetAnalogInputSequenceAcquisitionHandleStatus

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Initialisation status:
0: No sequence initialised
>0: Number of sequence initialised

Last initialised sequence handle: Last initialised sequence handle

Current sequence status: Current sequence status.
0: No sequence started
1: Sequence started
2: Sequence pause
3: Sequence ended

Current sequence handle: Current sequence handle

Error out: Error cluster output.

Task:

Gets the sequence handle status.
Returns the last initialised sequence status, the selected acquisition sequence handle and status.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.26  Convert digital to real analog value sequence.vi

C equivalent:

b_ADDIDATA_ConvertDigitalToRealAnalogValueSequence

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Sequence handle: This handle is returned by the "InitAnalogInputSequenceAcquisition.VI"

Digital values array:

Digital value of the analog input (composed of the digital value of the channel, of the calibration offset and of the calibration gain if available)

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Real values array: Returns the real analog value of the input channel

Error out: Error cluster output.

Task:

Converts the digital value array of the analog input into a real analog value array.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.27  Get 1 analog input module sequence information.vi

C equivalent:

b_ADDIDATA_GetAnalogInputModuleSequenceInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected Module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

Sequence configurable:
0: Sequence fixed. User must pass the first and last channel from the selected module
1: Sequence configurable User is free to define the first and last channel

Software trigger:
0: Software trigger not available
1: Software trigger available

Hardware trigger:
0: Hardware trigger not available
1: Hardware trigger available

Hardware gate:
0: Hardware gate not available
1: Hardware gate available

Common gain:
Defines if the gain can be different for each input
0: Gain is common on each input
1: Gain can be different for each input.

Common polarity:
Defines if the polarity can be different for each input
0: Polarity is common on each input
1: Polarity can be different for each input.

Number of gain:
Returns the number of gain values available

Gain available[255]:
Array which defines the available gain value

Delay time configurable:
0: Delay after each sequence not available
1: Delay after each sequence available

Delay available mode:
D0: 0: Mode 1 not available
1: Mode 1 available
D1: 0: Mode 2 not available
1: Mode 2 available

Delay calc type:
0: Binary type (XX000, XX001, XX010)
1: Multiple type (60, 120, 240, 480, 960, ...)

Delay time unit type:
0: Time unit (ns,s,ms,s, ...)
1: Frequency unit (MHz, kHz, Hz, mHz, ...)

Delay time unit:

For time unit:
D0: 0: ns not available
1: ns available
D1: 0: s not available
1: s available
D2: 0: ms not available
1: ms available
D3: 0: s not available
1: s available

For frequency unity:

D0: 0: MHz not available
1: MHz available
D1: 0: KHz not available
1: KHz available
D2: 0: Hz not available
1: Hz available
D3: 0: mHz not available
1: mHz available

Delay time resolution: Delay time resolution
8: 8-bit resolution
16: 16-bit resolution, ...

Min delay time: Minimum delay time.
7000: 7000 (ns)
10000: 10000 (ns), ...

Delay time step: Conversion delay time steps
20: 20 steps
50: 50 steps, ...

Acquisition mode:
XX1: The acquisition can work in Single mode
X11: The acquisition can work in continuous mode
1XX: The acquisition can work in counting mode

Max number of acquisition:
Return the max number of acquisition for the counting acquisition SCAN mode

Error out: Error cluster output.

Task:

Returns information about the analog input module for the sequence acquisition mode
(Init analog input sequence acquisition.VI and Start analog input sequence acquisition). The sequence mode uses the DMA. This allows the acquisition in the background.
Into the sequence mode the user defines a channel sequence to acquire. After each X sequence an interrupt is generated and the user receives the analog input values via a buffer.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.28  Get 1 analog input module hardware trigger status.vi

C equivalent:

b_ADDIDATA_GetAnalogInputHardwareTriggerStatus

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected Module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Hardware trigger flag:
ADDIDATA_ENABLE: The hardware trigger is enabled.
ADDIDATA_DISABLE: The hardware trigger is disabled

Hardware trigger status
0: Hardware trigger did not occur
1: Hardware trigger occurred

Hardware trigger count: Number of pulses that fail before the next trigger occurs

Hardware trigger state
0: Hardware trigger is not active (Low state)
1: Hardware trigger is active (High state)

Error out: Error cluster output.

Task:

Returns the status (occur or not), the state from input (active or not) and the number of that fail before the next trigger occur.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.29  Get analog input hardware trigger information.vi

C equivalent:

b_ADDIDATA_GetAnalogInputHardwareTriggerInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Module number: Virtual index of the selected Module.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

Low level trigger:
0: Hardware low trigger level not available
1: Hardware low trigger level available

High level trigger:
0: Hardware high trigger level not available
1: Hardware high trigger level available

Hardware trigger count:
0: Hardware trigger counter not available
1: Hardware trigger counter available BYTE

Hardware trigger auto refresh available mode:
D3 to D0 XXX1: One shot trigger available
XX1X: Single auto refresh trigger available
X1XX: X auto refresh trigger available

Hardware trigger SCAN available mode:
XXX1: One shot trigger available
XX1X: Single scan trigger available
X1XX: X scan trigger available

Hardware trigger sequence available mode:
XXX1: One shot trigger available
XX1X: Single sequence trigger available
X1XX: X sequence trigger available

Max trigger count value:
Max number of trigger pulses before the hardware trigger action occurs

Error out: Error cluster output.

Task:

Returns the hardware trigger available trigger actions and the available configuration mode

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.9.30  Enable disable analog input hardware trigger.vi

C equivalent:

b_ADDIDATA_EnableDisableAnalogInputHardwareTrigger

Input:

Driver handle in:	Handle of the driver to use the function (from Open the driver.VI).
Module number:	Virtual index of the selected Module.
Error in:	Error cluster input.

Output:

Driver handle out:	Handle of the driver to use the next function.

Hardware trigger flag:
1: ADDIDATA_ENABLE: Enables the hardware trigger.
0: ADDIDATA_DISABLE: Hardware trigger disabled by triggering the analog input module

Hardware trigger level:
1: ADDIDATA_LOW: If the hardware trigger is ued, it triggers from "1" to "0"
2: ADDIDATA_HIGH: If the hardware trigger is used, it triggers from "0" to "1"
3: ADDIDATA_HIGH_LOW: If the hardware trigger is used, it triggers from "0" to "1" or
from "1" to "0"

Hardware trigger action: Trigger action selection
(refer to table)

Hardware trigger cycle count: Define the number of sequences, auto refresh cycles or SCAN cycles to trigger
Refer to table

Hardware trigger action	Mode description
0	After each Hardware trigger count trigger a single conversion is started
1	After the first Hardware trigger count trigger the conversion are started. All next trigger have not effect.

	The trigger are rearmed after the next call from the function Start analog input sequence acquisition.VI orStart analog input SCAN.VI Start analog input auto refresh acquisition.VI
11	After Hardware trigger count trigger a single auto refresh cycle is started
10	After each Hardware trigger count trigger a series of Hardware trigger cycle count auto refresh cycles is started.
7	After Hardware trigger count trigger a single SCAN is started
6	After each Hardware trigger count trigger a series of Hardware trigger cycle count SCAN is started.
3	After Hardware trigger count trigger a single sequence is started
2	After each Hardware trigger count trigger a series of Hardware trigger cycle count sequences is started

Hardware trigger count: Hardware trigger counter. Defines the number of trigger events before the action occur (> 0)

Time out: Define the time out for the ADDIDATA_TRIGGER_START_A_SINGLE_CONVERSION mode.
Unity is ms.
0: No time out used

Error out: Error cluster output.

Task:

Releases or blocks the action of the hardware trigger.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.10 Analog output driver

2.10.1  Get number of analog outputs.vi

C equivalent:

b_ADDIDATA_GetNumberOfAnalogOutputs

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Number of analog outputs: Number of the analogue output channels

Analog output type:
Type Array.
Gives the type of each analogue output.
See Table

Value	Definition
0	Analog output is immediately written
1	Analog output can be written later with synchronisation

Analog output type [0]: Type of the first analog output.
Analog output type [1]: Type of the second analog output.
...
Analog output type [Number of analog output - 1]: Type of the last analog output

Error out: Error cluster output.

Task:

Returns the number of analogue outputs and the type of each analogue output.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.10.2  Get 1 analog output information.vi

C equivalent:

b_ADDIDATA_GetAnalogOutputInformation

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Analog output number: Virtual index of the selected analog output.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Information cluster:

Number of voltage mode: Number of voltage modes which can be used for the channel

High voltage range array: Possible voltage range for the selected analogue output in V.

Low voltage range array: Possible voltage range for the selected analogue output in hundredths of V.

SW polarity array: Possible software polarity for the selected analogue output. See Table

HW polarity array: Possible hardware polarity for the selected analogue output. See Table.

Value	Definition
0	No polarity available.
1	0 V to Max Voltage Range
2	- Max Voltage Range to + Voltage Range
3	Both are available

Resolution array: Possible resolution for the selected analogue output. (see table)

Value
Resolution
10 10-bit
11 11-bit
12 12-bit
13 13-bit
14 14-bit
15 15-bit
16 16-bit
24 24-bit
32 32-bit

Synchronisation: If = 1, synchronisation is available for the analogue output.

Error out: Error cluster output.

Task:

Gives the number of modes (Number of voltage mode), the voltage range (High voltage range and High voltage range arrays), the polarity (SW polarity and HW polarity arrays), the synchronisation (Synchronisation) and the resolution (Resolution arrays) which can be set for the selected analog output (Analog output number).

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.10.3  Init more analog outputs.vi

C equivalent:

b_ADDIDATA_InitMoreAnalogOutputs

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Number of analog outputs: Number of the analogue outputs to be initialised.

Initialisation cluster:

Channel index: Analog output virtual index array.
The first analogue output begins from 0.

Voltage mode: Voltage or current range array.

See Table

Value	Definition
0	First output range supported by the board
1	Second output range supported by the board
2	Third output range supported by the board
3	Fourth output range supported by the board

Polarity: Polarity array.
See Table

Value	Definition
1	0 V to Max Voltage Range
2	- Max Voltage Range to + Voltage Range

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Initialises the selected analog input channel.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.10.4  Write more analog outputs.vi

C equivalent:

b_ADDIDATA_WriteMoreAnalogOutputs

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Number of analog outputs: Number of the analogue outputs to be written.

Initialisation cluster:

Channel index: Analog output virtual index array.
The first analogue output begins from 0.

Digital value to write: Array of the digital values to be written.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Write a value on each of the selected channels.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.10.5  Release more analog outputs.vi

C equivalent:

b_ADDIDATA_ReleaseMoreAnalogOutputs

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Number of analog outputs: Number of the analogue outputs to be released.

Channel index: Analog output virtual index array.
The first analogue output begins from 0.

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

Releases several analogue output channels.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.11 Interrupt driver

2.11.1  Set the interrupt functionality.vi

C equivalent:

b_ADDIDATA_SetFunctionalityIntRoutineWin32

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Functionality: Determines the functionality on which the interrupt management is to be stopped.
0: ADDIDATA_DIGITAL_INPUT
1: ADDIDATA_DIGITAL_OUTPUT
2: ADDIDATA_ANALOG_INPUT
3: ADDIDATA_ANALOG_OUTPUT
4: ADDIDATA_TIMER
5: ADDIDATA_WATCHDOG
6: ADDIDATA_TEMPERATURE
7: ADDIDATA_COUNTER
8: ADDIDATA_BI_DIRECTIONAL
9: ADDIDATA_RESISTANCE
10: ADDIDATA_TIMER_COUNTER_WATCHDOG
11: ADDIDATA_PRESSURE
12: ADDIDATA_TRANSDUCER
13: ADDIDATA_DIG_IO

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Error out: Error cluster output.

Task:

This function must be called up for each functionality for which an interrupt is to be enabled. It installs one user interrupt function for all functionalities of the same type.

Return value:

1: No error.

0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.11.2  Test if interrupt.vi

C equivalent:

b_ADDIDATA_TestInterrupt

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Error in: Error cluster input.

Output:

Driver handle out: Handle of the driver to use the next function.

Functionality: Mask of the functionality which has generated the interrupt.
See table.

Interrupt mask: Mask of the events which have generated the interrupt.
See table.

Byte array: The value passed to this variable depends from the used functionality.
See table.

Word array: The value passed to this variable depends from the used functionality.
See table.

Dword array: The value passed to this variable depends from the used functionality.

See table.Table: INTERRUPTSOURCE.HLP

Error out: Error cluster output.

Task:

Checks if functionality has generated an interrupt. If yes, the function returns the functionality type and the interrupt source in the different arrays.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

2.11.3  Reset the interrupt functionality.vi

C equivalent:

b_ADDIDATA_ResetFunctionalityIntRoutine

Input:

Driver handle in: Handle of the driver to use the function (from Open the driver.VI).

Functionality: Determines the functionality on which the interrupt management is to be stopped.

0: ADDIDATA_DIGITAL_INPUT
1: ADDIDATA_DIGITAL_OUTPUT
2: ADDIDATA_ANALOG_INPUT
3: ADDIDATA_ANALOG_OUTPUT
4: ADDIDATA_TIMER
5: ADDIDATA_WATCHDOG
6: ADDIDATA_TEMPERATURE
7: ADDIDATA_COUNTER
8: ADDIDATA_BI_DIRECTIONAL
9: ADDIDATA_RESISTANCE
10: ADDIDATA_TIMER_COUNTER_WATCHDOG
11: ADDIDATA_PRESSURE
12: ADDIDATA_TRANSDUCER
13: ADDIDATA_DIG_IO

Error in: Error cluster input.

Output:
Driver handle out: Handle of the driver to use the next function.
Error out: Error cluster output.

Task:
Stops the interrupt management of the selected functionality (Functionality).
Deinstalls the user interrupt routine for this functionality.

Return value:

1: No error.
0: Error by calling the VI. Use the Manage the last error.VI to find the error source.

3 ADDIPACK SAMPLES

3.1 ADDIPack DEMO Get available functionality.vi

Task: This sample demonstrates how to get the functionality avalaibality.

3.2 Digital input demo

3.2.1 ADDIPack DEMO Test 1 digital input.vi

Task: This sample demonstrates how to use 1 digital input.

3.2.2 ADDIPack DEMO Test 32 digital inputs.vi

Task: This sample demonstrates how to use a 32 digital input port.

3.3 Digital output demo

3.3.1 ADDIPack DEMO Test 1 digital output.vi

Task: This sample demonstrates how to use 1 digital output.

3.3.2 ADDIPack DEMO Test 32 digital outputs.vi

Task: This sample demonstrates how to use a 32 digital output port.

3.4 ADDIPack DEMO Test 1 timer.vi

Task: This sample demonstrates how to use 1 timer.

3.5 ADDIPack DEMO Test 1 counter.vi

Task: This sample demonstrates how to use 1 counter.

3.6 ADDIPack DEMO Test 1 watchdog.vi

Task: This sample demonstrates how to use 1 watchdog.

3.7 Analog input demo

3.7.1 ADDIPack DEMO Test more analog inputs in polling mode.vi

Task: This sample demonstrates how to use more analog input in polling mode.

3.7.2 ADDIPack DEMO Test more analog inputs in SCAN mode.vi

Task: This sample demonstrates how to use more analog input in SCAN mode.

3.7.3 ADDIPack DEMO Test more analog inputs in sequence mode.vi

Task: This sample demonstrates how to use more analog input in sequence mode.

3.7.4 ADDIPack DEMO Test more analog inputs in auto refresh mode.vi

Task: This sample demonstrates how to use more analog input in auto refresh mode.

3.8 Analog output demo

3.8.1 ADDIPack DEMO Test more analog outputs.vi

Task: This sample demonstrates how to use more analog outputs.