ADwin

Testpoint-Driver

for ***ADwin***-Systems

Version 1.5

January 1999

1. Special features of *ADwin* boards.

The PC board ***ADwin*** includes its own processor for fast data acquisition, high-speed control and analysis. Combined with ***ADwin***, Testpoint gains real time abilities and a response task latency of less than 300 nanoseconds.

- The **A/D-Object** has been extended with a pretrigger function which can also acquire the signal before the trigger event. Up to **four A/D-Objects run simultaneous** on a single ***ADwin*** board.

- The **D/A-Object** enables the output of signals with an output frequency up to 100 kHz. **Two D/A-Objects run simultaneous** with different output frequencies.

- The **PID-Object** provides digital controllers with sampling rates up to 80 kHz. The control deviations and the actuator values can be read out by Testpoint during control. **Up to four controllers run parallel** with different sampling rates on a single ***ADwin*** board.

- The **DIO-Object** includes functions for digital in- and outputs via ***ADwin*** board.

- The **REALTIME-Object** controls the freely programmable and very fast realtime processes that have been developed with ***ADbasic***.

- The **SYSTEM-Object** allows remote controlling and watching of ***ADwin***-Systems via network (LAN, ISDN, Internet, ...).

The RISC processor used by the ***ADwin*** boards is especially designed to do fast parallel processing. All of the objects above can run parallel. **Several controllers, signal generators, counters and triggered measurement acquisitions run simultaneous** on a single ***ADwin*** board. Therefore it is possible to control and monitor complete test-rigs with one ***ADwin*** board.

2. *ADwin*-Software installation guide

2.1 Installing the *ADwin* Driver

Please insert the disk labeled ***ADwin*** in your CD-ROM-drive. The program SETUP.EXE on this disk starts automatically. (If the program don't start, please run the program **setup.exe** on the root directory. Press the button '***ADwin*** Driver Setup'. After choosing the language and the destination directory (it is recommended to confirm the default directory: C:\ADBASIC3), the following files are copied to the specified directory:

ADWIN4.BTL	Driver for the ***ADwin-4*** card
ADWIN5.BTL	Driver for the ***ADwin-5*** card
ADWIN8.BTL	Driver for the ***ADwin-8*** card
ADWIN9.BTL	Driver for the ***ADwin-9*** (ADSP) card
ISERVER.EXE	Load program for INMOS transputers
RUN.BAT	Help program for loading one of the *.BTL drivers
ADBLOAD.EXE	Program to load compiled ADbasic processes under DOS
TESTVE16.EXE	Displays all 16-Bit ADwin DLLs, installed on your system
TESTVE32.EXE	Displays all 32-Bit ADwin DLLs, installed on your system
ADTEST.EXE	Program to test the ***ADwin*** board
ADWINSET.EXE	Program to register the link address of the ***ADwin*** board (only for Windows NT)

Depending on your Windows version, the following files are copied to your Windows directory:

Windows 3.X	**Windows 95**	**Windows NT**
ADWIN.DLL	ADWIN.DLL	ADWIN.DLL
ADWIN32.DLL	ADWIN32.DLL	ADWIN32.DLL
ADPOINT.DLL	ADPOINT.DLL	ADPOINT.DLL
	ADWIN95.DLL	ADWINNT.DLL

When the setup program is finished correctly, you receive the message: The setup was successful.

If using *Windows NT*, you have to restart your computer after the driver installation is finished.

2.2 Load the example and copy the user defined objects to the stock

You need the file: **ADwin.tst .**

Start Testpoint and load the file: **ADwin.tst** . Copy the user defined objects **A/D, D/A , DIO, PID, REALTIME** and **SYSTEM** to the Testpoint stock window.

To do this, you have to grab the icons in the Testpoint object window and drag them to the stock window.

While loosing the left mouse button the control key on the PC keyboard has to be pressed.

2.3 Loading the *ADwin* driver

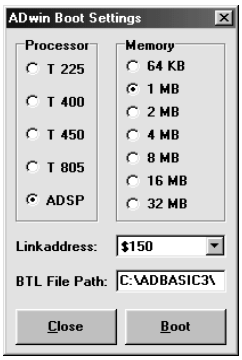
The appropriate driver must be loaded in order to access the ***ADwin*** card with your PC. To load the driver you can choose one of the following possibilities.

a) Using TestPoint

Use the action: ***Boot*** to be found in the ***ADwin***-System Object. Make sure the Settings of the System Object match with your ***ADwin*** card.

b) Using the program: ADtest

The program ADtest is mainly used to test and watch the ***ADwin*** board. It also offers the possibility to load the driver. To do this, start the program **ADtest.exe**. If you did not load the driver since your system was powered on, the following dialogbox appears:



Make sure that all settings match with your ***ADwin*** card and press the OK button. If no error message appears, the driver loading was successful. Otherwise press the Settings Button and try again with new settings. A detailled description of the program ADtest can be found in the next section.

ATTENTION !

The ***ADwin*** driver has to be loaded before trying to access the ***ADwin*** card. The driver builds the base for communication between PC and ***ADwin*** card.

3. The program *ADtest* for testing and watching the *ADwin* board

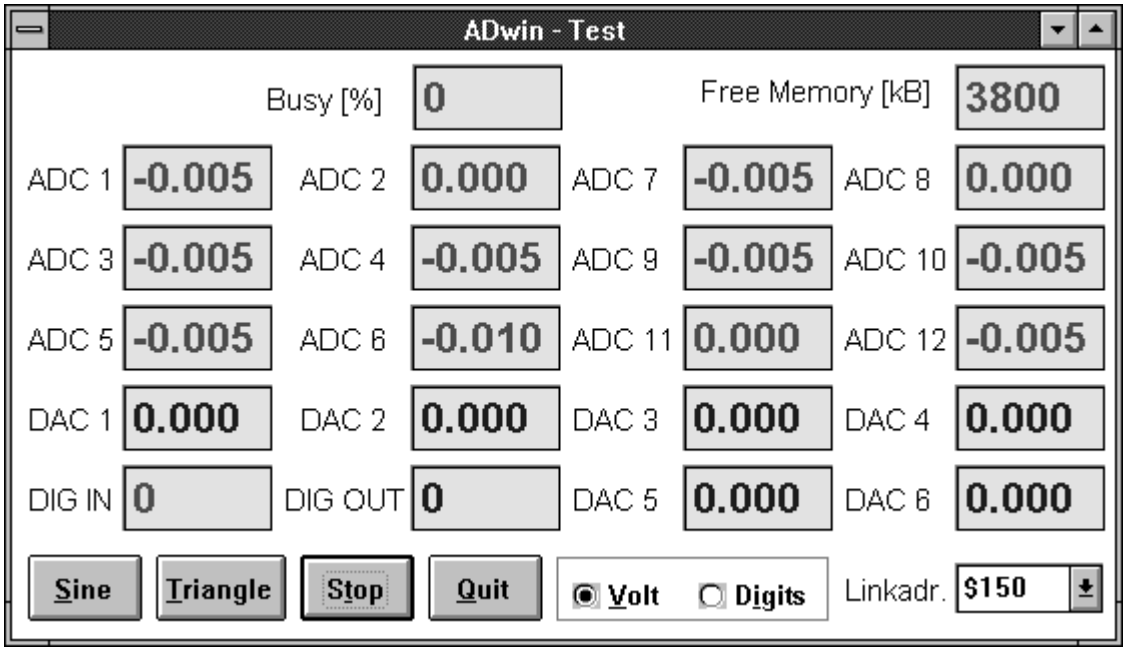
a) Test functions

The programm ADtest samples the 12 analog inputs of the ADwin board and displays the TTL levels of the digital inputs. The analog and digital outputs can be set by the user.
For testing, a 10 Hz sine or triangle signal can be given out via the analog outputs of the ADwin board.

b) Watch functions

The display ‘Busy [%]’ shows the transputer workload in percent. Workload rises linear with the frequency of the cyclic I/O processes. The window next to the ‘Busy [%]’ display shows the free memory on the ***ADwin*** board in kilo bytes. Memory is allocated each time a ***ADwin*** Testpoint object is started. Especially for applications that run simoultaneous several processes on the ***ADwin*** board this little program is very useful. It is easy to watch the need of memory and transputer workload required by each object.

ADtest runs simoultaneous with Testpoint under Windows.



A/D Object

The A/D object accesses the analog inputs of the ***ADwin*** board.
On the ***ADwin*** board up to four A/D processes can run simultaneous. Each process is controled by its own A/D Object. The numbers of the processes can be fixed at the A/D objects Settings-Menue.

Actions

Start A/D

Start A/D begins sampling one or more channels of analog input data at a specified rate. Sampling occurs in background, while processing continues with the next action line (that is, the sampling is not completed before the next line is executed). Therefore it is recommended to access data in the action list of the A/D object and not in the following action lines of the calling action list. **The A/D-Object can be used simoultanous with clocked analog outputs (by D/A Object) and the digital feedback controller.** The ***ADwin*** board is able to run several processes parallel with different clock rates.

Start A/D A/D1 #samples=1000, rate=1000 Hz, channel(s)=1, event after"ALL" sample(s)

#samples - the total number of samples to be acquired per channel.
Can be **from** 2 **to** 500000 or CONTINUOUS (to indicate sampling should not stop until the Stop A/D action is used).

rate - the sampling rate in Hertz.
Can be **from** 1 **to** 60000 (60kHz)

channel(s) - the channel(s) to be sampled.
A single number can be entered for an individual channel. Channel numbers start at 1.
If multiple channels are to be sampled, list the channels separated by commas.
Can be **from** 1 **to** 12

event after ... samples - how many samples should be gathered between A/D events (execution of the A/D action list). For slow sampling, a value of one allows each sample to be processed individually in the action list. In other cases, processing after the A/D run is complete is more appropriate. The special value **all** can be used to indicate an event after the total number of samples have been acquired.
Can be **from** 1 **to** 500000 or all

Stop A/D

Stop A/D stops any active background sampling.

Stop A/D A/D1

Acquire A/D

Acquire A/D takes a specified number of samples at a given rate from one or more A/D channels. **This action completes the entire acquisition before continuing to the next action line**, so sequences like Acquire A/D, then Draw Graph are OK. **The action acquire A/D can be used simoultanous with clocked analog outputs (by D/A Object) and the digital controller.** The ***ADwin*** board is able to run several processes parallel with different clock rates.

#samples - the total number of samples to be acquired per channel.
Can be **from** 2 **to** 4000

rate - the sampling rate in Hertz.
Can be **from** 31 **to** 30000 (30kHz)

channel(s) - the channel(s) to be sampled.
A single number can be entered for an individual channel. Channel numbers start at 1.
If multiple channels are to be sampled, list the channels separated by commas.
Can be **from** 1 **to** 12

Set A/D trigger analog (only process 1)

Set A/D trigger analog sets the triggering mode used for the next Start A/D action to be based on one of the analog input channels. The trigger level, polarity, edge/level mode and pretrigger time can all be specified. This function is valid only, if the process number in the Settings-Menue of the A/D-object is set to 1.
All other process numbers use immediate triggering.

channel - analog input channel that is used for triggering.
This channel must be included in the channel list of the Start A/D action.
Can be **from** 1 **to** 12

level - Trigger voltage that must be passed over or fall short of.
Can be **from** -10 V **to** 10 V

polarity - **Positive level:** Start sampling when measured value is greater than ***level***
Negative level: Start sampling when measured value is lower than ***level***
Rising Slope: Start sampling when measured value exceeds ***level***
Falling Slope: Start sampling when measured value falls below ***level***

pretrigger Time in milliseconds, that is displayed before triggering.
Can be **from** 0.0 **to** whole measurement time

Set A/D trigger immediate (only process 1)

Set A/D trigger immediate sets the triggering mode to be used for the next Start A/D action to immediate. Immediate mode means that sampling begins as soon as the Start A/D action line executes.

Set A/D trigger immediate A/D1

Sample A/D

Sample A/D takes a single reading on one or more A/D channels. The A/D object data is set to the result of the sampling action (one number or a list of numbers). **Running measurements, analog outputs and digital feedback controllers are not influenced by this action.**

Sample A/D A/D1 once, channel(s)=

channel(s) - the channel(s) to be sampled.
A single number can be entered for an individual channel. Channel numbers start at 1.
If multiple channels are to be sampled, list the channels separated by commas.
Can be **from** 1 **to** 12

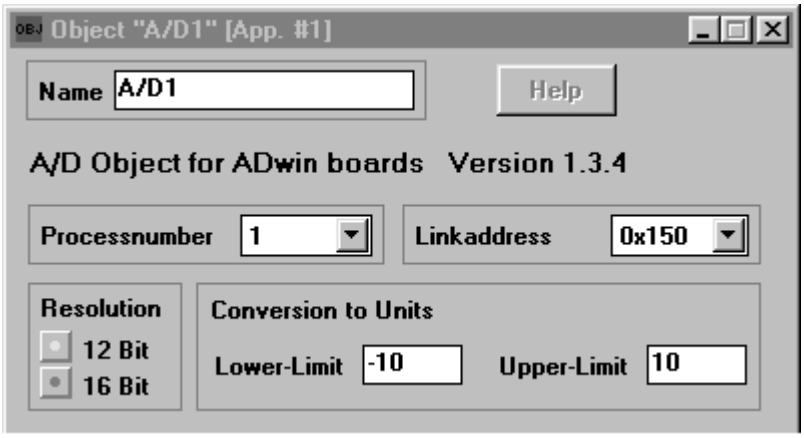
Data

The data value of the A/D object is the result of the last analog input operation.
In case of a 'Sample A/D' action, the data will be a number or list of numbers (if more than one channel is sampled).
In the case of a 'Start A/D' or 'Acquire A/D' action, the object's data is set when the A/D event occurs (the specified number of samples). The data is a vector of the specified number of samples, or a list of vectors when more than one channel is sampled.

Action List

The action list is executed when background A/D sampling is active, after the 'Start A/D' action. At each execution of the action list, the A/D object data is equal to the newly acquired samples.

Settings



Processnumber ***ADwin*** boards are able to run parallel up to four A/D processes without influencing each other. The processnumber indicates which process is addressed by the A/D object.

Linkaddress The linkaddress, that is switched on the ADDR-Dip-switch of the ***ADwin*** board. Because of the variable linkaddress several ***ADwin boards*** can be used in a single PC.

Resolution The resolution of the A/D-converters on your ***ADwin***-System.
Use 12 Bit for ***ADwin*** PC-boards, use 16 Bit for ***ADwin-GOLD***.

Lower-Limit, Upper-Limit

ADC values are transformed to the range, specified by Lower- and Upper-Limit. To get the analog input voltage, Lower- and Upper-Limit must be set to the range that is jumpered on the ***ADwin*** board.
The default voltage-range of the ***ADwin*** board is -10V ... +10V. In this case Lower-Limit should be set to -10 and Upper-Limit should be set to 10

D/A Object

The D/A object allows output on the analog outputs of the ***ADwin*** board.
On the ***ADwin*** board two D/A processes can run simultaneous. Each process is controled by its own D/A Object. The numbers of the processes can be fixed at the D/A objects Settings-Menue.

Actions

Start D/A

Start D/A begin sending output values to an analog output channel, at a given clock rate.
This function run simoultanous with clocked measures (by A/D Object) and digital feedback controlers (by PID Object). The ***ADwin*** board is able to run several processes parallel with different clock rates.

Start D/A D/A1 rate=10000 Hz,channel=1 ,values=rampe ,repeat=20

rate - the clock rate in Hertz.
Can be **from** 1 **to** 100000 (100kHz)

channel - the channel to send the values to.
Can be **from** 1 **to** 6

values - the voltages to be output.
Should be a vector of numbers, from some other object's data
Can be **from** -10V **to** 10V
Vector length can be **from** 1 **to** 1000000

repeat - how many times the whole output should be repeated
Can be **from** 1 **to** 1000000000

Stop D/A

Stops any active D/A timed output.

Stop D/A D/A1

Output D/A

Output D/A writes a single voltage value to a single D/A channel immediatly.
This action do not influence running measures, analog outputs and digital feedback controllers in any way.

```
Output D/A    D/A1          once,channel=1      ,value=0
```

channel - the channel to send the value to.
Can be **from** 1 **to** 6

value - the voltage to be output.
Can be **from** -10V **to** 10V

Settings

Processnumber ***ADwin*** boards are able to run parallel up to four D/A processes without influencing each other. The processnumber indicates which process is addressed by the D/A object.

Linkaddress The linkaddress, that is switched on the ADDR-Dip-switch of the ***ADwin*** board. Because of the variable linkaddress several ***ADwin boards*** can be used in a single PC.

Resolution The resolution of the D/A-converters on your ***ADwin***-System.
Use 12 Bit for ***ADwin***-PC boards, use 16 Bit for ***ADwin-GOLD***.

DIO Object

The DIO object allows digital input and output with ***ADwin*** boards.
The ***ADwin*** board is equipped with 16 digital in- and outputs, respectivley combined in a 16-bit word.

Actions

Output to

Output to sets the digital outputs

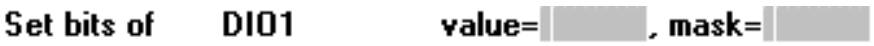
Value output word (0 = low, 1 = high)

Input from

Input from reads the digital inputs

Set bits of

Set bits of outputs a word to the digital outputs, affecting only the bits which are 1 in the mask parameter. Bits which are 0 in the mask parameter are unchanged.



Value output word (0 = low, 1 = high)
mask output mask (0 = affect, 1 = do not affect)

PID Object

The PID Object realizes fast digital controlling with ***ADwin*** boards.
On the ***ADwin*** board up to four controller processes can run simultaneous. Each process is controled by its own PID Object. The numbers of the processes can be fixed at the PID objects Settings-Menue.

Actions

Start PID

Start PID starts a PID-controller on the ***ADwin*** board. Control parameters are variable. The controller saves deviation and actuator values for the first cycles in a buffer. As soon as the buffer is filled (default is 256 values), the action list of the PID-controller is executed. The objects data is a list of two vectors, that contains the saved values.
The PID-controller run simoultanous with clocked measures (by A/D Object) and clocked analog outputs (by D/A Object). The ***ADwin*** board is able to run several processes parallel with different clock rates.

Start PID PID1 Setpoint=0 V, k=5 , t0=1 mS, td=0.2 mS, ti=50 mS

Setpoint - setpoint in Volts
Can be **from** -10V **to** 10V

k - gain factor
Can be **from** 0.1 **to** 1000

t0 - cycle time in milliseconds
Can be **from** 0.033ms **to** 1000ms

td - differential time in milliseconds
Can be **from** 0.001ms **to** 1000000ms

ti - integration time in milliseconds
Can be **from** 0.01ms **to** 1000000ms

Stop PID

Stop PID Stops the active PID-controller

Stop PID PID1

Get PID

Get-PID clears the controller save buffer to get space for the actual values. If the buffer is filled, the action list of the PID-controller is executed.

Get PID PID1

Set PID

Set PID sends new control parameters to a running controller. After changing, the controller save deviation and actuator values for the first cycles in a buffer. As soon as the buffer is filled (default is 256 values), the action list of the PID-controller is executed. The objects data is a list of two vectors, that contain the saved values.

Set PID PID1 Setpoint=0 V, k=-1 , t0=-1 mS, td=-1 mS, ti=-1 mS

Setpoint - setpoint in Volts
Can be **from** -10V **to** 10V

k - gain factor
Can be **from** 0.1 **to** 1000

t0 - cycle time in milliseconds
Can be **from** 0.033ms **to** 1000ms

td - differential time in milliseconds
Can be **from** 0.001ms **to** 1000000ms

ti - integration time in milliseconds
Can be **from** 0.01ms **to** 1000000ms

Config PID

Config PID fixes measure input, regulate output and buffer size.
This action is only executed, if the controller is not running.

Config PID PID1 Measurement input=1 , Actuating output=1 , Buffersize=256 samples

Measurement input - number of controller input channel
Can be **from** 1 **to** 12

Actuating output - number of analog output for controller setpoints
Can be **from** 1 **to** 6

Buffersize - size of buffer which is used to save controller behaviour
Can be **from** 8 **to** 2048

Data

The PID-Object data is a list of two vectors, that contain the saved deviation and regulate values since the last Start-, Set- or Get-action. Vector length is fixed by the buffer size.

Action List

The action list is executed after every Start-, Set- or Get function, as soon as the buffer is filled.

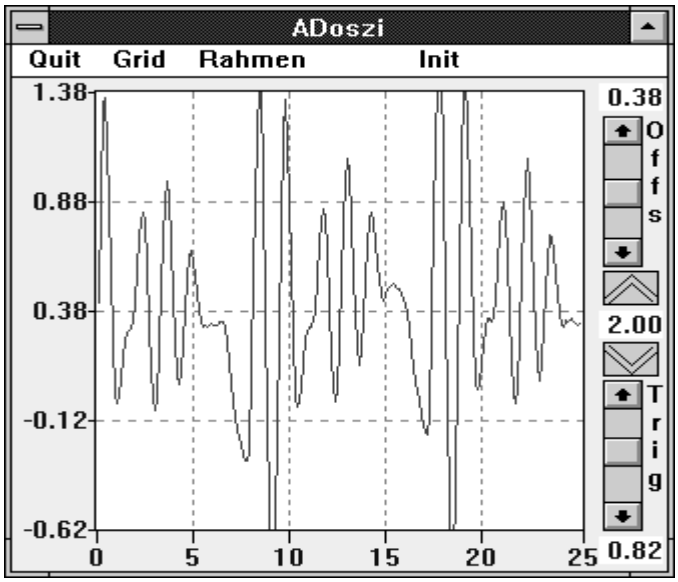
Settings

Processnumber ***ADwin*** boards are able to run parallel up to four PID processes without influencing each other. The processnumber indicates which process is addressed by the PID object.

Linkaddress The linkaddress, that is switched on the ADDR-Dip-switch of the ***ADwin*** board. Because of the variable linkaddress several ***ADwin boards*** can be used in a single PC.

Oszi Object

The Oszi Object includes a fast graphical display for ADwin-4 and ADwin-8 cards.



The Oszi object has the ability to display continuous up to two analog inputs without any delay or flickering.

The Oszi object include edge controled trigger functions with variable pretrigger time.

The Oszi-object in use with ***ADwin***-8 cards can carry out a fast online FFT from signals up to 15kHz and show the results in the graphical display with an update rate of 10 Hz.

The Oszi-object run simoultanous with other ***ADwin*** Testpoint Objects on one ***ADwin*** board.
For example:

- Signal generation with D/A-Object and measurement with Oszi-object
- slow measurement with AD-Object and fast control via display with Oszi-object
- Controlling with PID-object and display of deviation with Oszi-object

Actions

Start Oszi starts the fast graphical display

Stop Oszi stops and erease the fast graphical display

Remark: The Oszi-Object is not included in the standard Testpoint Driver Software. It can be ordered optional for the price of 600,- DM + MWSt.

ADoszi-Driver install guide

The following files are needed:

- **ADwin8.btl** Version 1.1
- **ADoszi.dll**
- **ADoszis.exe**
- **ADpoint.dll**
- **iserver.exe**
- **run.bat**

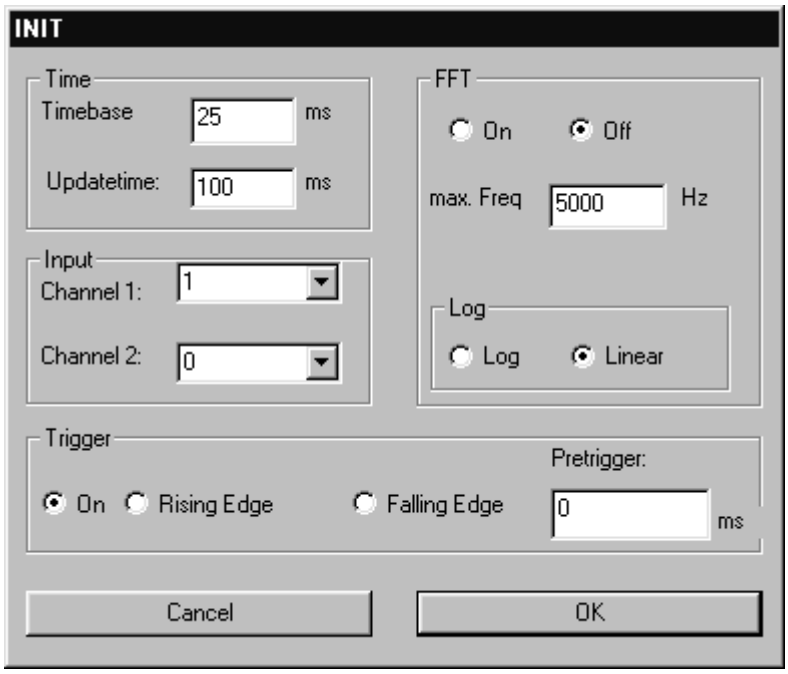
Before starting Testpoint, the programm **ADwin8.btl** has to be loaded to the ***ADwin*** board by typing ***run ADwin8*** on the DOS command line. After the programm is loaded, the message ***ADwin8 V1.4.1 started !*** appears on the screen.

Copy the files **ADoszi.dll, ADoszis.exe** and **ADpoint.dll** to your Windows-Subdirectory.

Start Testpoint, load the file **ADwin.tst** and copy the user defined object **Oszi,** to the stock.

To test the correct function, load and start the programm udo_oszi.tst.

.



REALTIME Object

The REALTIME-Object handles very fast- and free programable real-time processes created with ***ADbasic*** On the ***ADwin*** board up to 10 *ADbasic* processes can run simultaneous. Each process is controled by its own REALTIME Object. The numbers of the processes can be fixed at the REALTIME objects Settings-Menue.
ADbasic is able to save compiled processes that can be loaded to the ***ADwin*** board by the action Load process.

Actions

Start Process

Start Process starts a process on the ***ADwin*** board created with ***ADbasic***. On the ***ADwin*** board up to 10 *ADbasic* processes can run simultaneous. Each process is started by its own REALTIME Object.

Start Prozess ADwin1

Stop Process

Stop Process stops an ***ADbasic*** process running on the ***ADwin*** board.

Stop Prozess ADwin_Pr_1

Set Parameter

Set Parameter sets a parameter, used by ***ADbasic,*** to the desired value.
A series of 80 parameters are available, numbered from 1 to 80. Each parameter represents a long-Integer varible. In ***ADbasic*** these variables are named Par_1 ... Par_80.
On the ***ADwin-8*** card there are additional 80 floating-point parameters, named Fpar_1 bis Fpar_80. These parameters can be addressed by switching the Type field of the action Set Parameter to: “Float“

Set Parameter ADwin_Pr_1 Parameter no. = 1 , value = 1 , type = "Integer"

Parameter no.	Number of the parameter that should be set. By using the Pull-Down List, the delay between 2 processor timer generated events (**Delay**) and the max number of events can be set.
Value	32-Bit integer or float to be send to the ***ADwin*** board
Type	parameter-type Integer = Par, Float = Fpar

Read Parameter

Read Parameter reads a parameter, that has been set by a ***ADbasic*** process.
A series of 80 parameters are available, numbered from 1 to 80. Each parameter represents a long-Integer varible. In ***ADbasic*** these variables are named Par_1 ... Par_80.
On the ***ADwin-8*** card there are additional 80 floating-point parameters, named Fpar_1 ... Fpar_80. These parameters can be addressed by switching the Type field of the action Read Parameter to: “Float“

Read Parameter ADwin_Pr_1 Parameter no. = 1 , type = "Integer"

Parameter no. Number of the parameter that should be read

Type parameter-type Integer = Par, Float = Fpar

Set Data

Set Data sends a set of values (data) to the ***ADwin*** board. There are 200 sets available, numbered from 1 to 200. Before using this action the desired set has to be defined in ***ADbasic*** with at least the count of elements that should be send.
In ***ADbasic*** the data sets are arrays, named data_1 .. data_200.

Set Data ADwin_Pr_1 Dataset no. = 1 , length = 100 , values = Container1

Dataset no. Number of data set

length - Count of elements to be transfered

Value - Elements to be transfered (e.g. container)

Read Data

Read Data gets a data set from the ***ADwin*** board that was set by an ***ADbasic***-process.
There are 200 sets available, numbered from 1 to 200.
In ***ADbasic*** the values can be set by addressing arrays, named data_1 .. data_200.

Read Data ADwin_Pr_1 Dataset no. = 1 , lenght = 100 ,start at 1

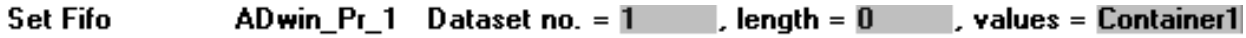
Dataset no. Number of data set

length - Count of elements to be transfered

Start at Start index

Set Fifo

To ensure a continuous data flow, the data set structure can be organized as fifo memory. In this case the elements send, are automaticly appended to the previous send data. Before using this action the desired set must be defined in ***ADbasic*** as fifo. The return value of this action is the count of free elements in the fifo.



Dataset no. Number of data set

length - Lenght of data set that should be appended. If this field is set to 0, the action returns the count of unused (free) fifo elements.

Values - The elements that should be appended.

Read Fifo

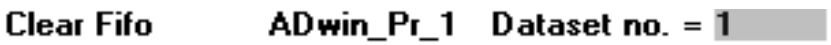
To ensure a continuous data flow, the data set structure can be organized as fifo memory. This action reads out the data of the used ***ADbasic*** fifo in the same sequenze in which it were written. The used Data sets must be defined in ***ADbasic*** as fifo.

Dataset no. Number of data set to read

length - Count of elements that should be read. If set to **all,** all stored elements are returned. If demanding more elements than available, all existing elements are returned.

Clear Fifo

Clear Fifo clears the contents of the desired fifo. This action must be used before starting a measurement task to ensure that no data from previous tasks is present.



Dataset no. Number of data set that should be cleared

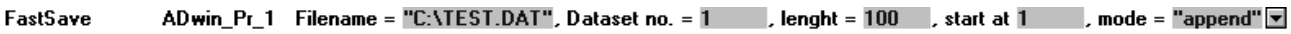
Load Process

Load Process loads a compiled **ADbasic**-process to the ***ADwin*** board. That means it is possible to load compiled ***ADbasic*** processes without using the ***ADbasic*** development tool.

Process name The name of the file you want to load.
The name must include path and extension „.T81“ or „.T82“ , “.T83“ for process No. 2 or No. 3 .

FastSave

This function is used to write the values of an ADbasic data set directly to disk. Depending on your system transfer rates up to 200 kbyte/second can be reached.



Filename Data is written to this file.
The name must include path and extension.

Dataset no. Number of data set to write.

length - Count of elements that should be saved.

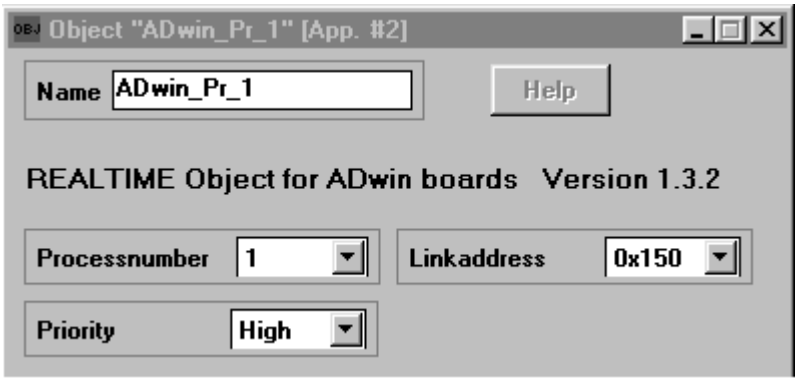
Start at First Element to save

mode Save mode: append = If file exists, data is appended to the end of the file.
overwrite = Existing file is erased before writing.

Action List

The ***ADbasic*** function **"activate_pc"** executes the action list of the REALTIME object.

Settings



Processnumber The number of the ***ADbasic***-processes, that should be addressed (1 - 10).
If running two or more ***ADbasic***-processes on the ***ADwin*** board, two or more REALTIME objects are needed. The actions Load Process, Start- and Stop Process are only valid for the ***ADbasic***-process, specified by this field.

Linkaddress The linkaddress, that is switched on the ADDR-Dip-switch of the ***ADwin*** board. Because of the variable linkaddress several ***ADwin boards*** can be used in a single PC.

Priority The priority of the ***ADbasic***-process. Fast measurement processes should be run with " High " priority, slow evaluation processes with " Low " priority.

ADwin-System Object 

The ***ADwin***-System Object is used to initialize the processor on the ***ADwin*** board and loads the driver program to the card. It also includes actions that dertermine the processor busyness and free memory of the ***ADwin*** board.

Actions

Boot

Boot initializes the ***ADwin*** board and loads the driver for communication between Testpoint and ***ADwin*** board.

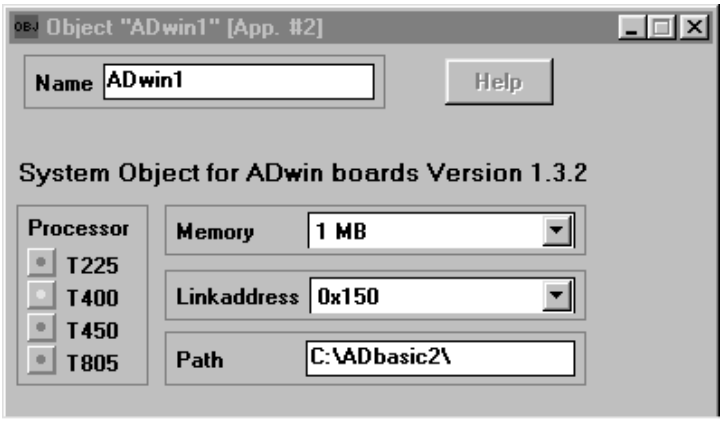
Workload

Workload delivers th actual processor busyness of the ***ADwin*** board . The return value is in the range of 0 to 100 per cent.

Memory

Memory returns the free memory of the ***ADwin*** board.

Settings



Processor Processor of the used ***ADwin*** board.

Memory Available memory of the used ***ADwin*** board.

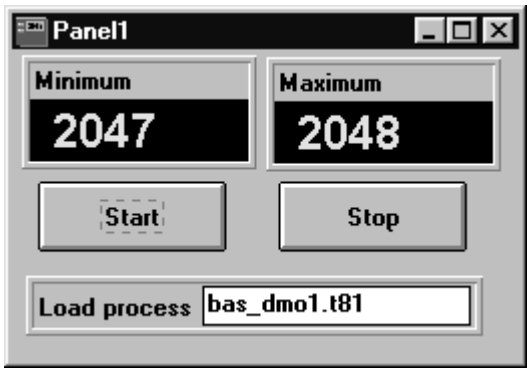
Linkaddress The linkaddress, that is switched on the ADDR-Dip-switch of the ***ADwin*** board. Because of the variable linkaddress several ***ADwin boards*** can be used in a single PC.

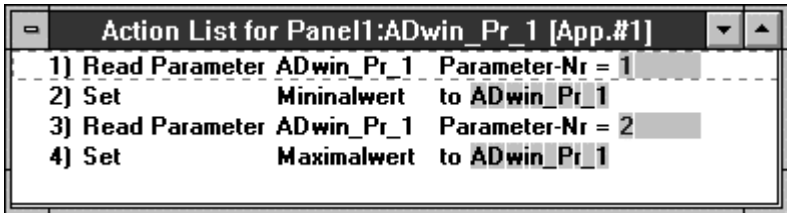
Path The path where the driver program (according to the used processor: adwin2.btl, adwin4.btl or adwin8.btl) can be found.

ADbasic Sample Programs :

1. On-line evaluation of measurements

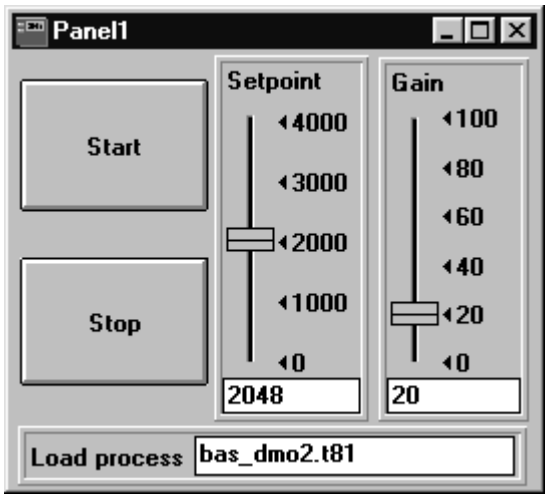
```
REM The program BAS_DMO1.BAS finds the maximum and minimum values out of
REM 1000 measurements from ADC1 and writes the result in the variables
REM PAR_1 and PAR_2.

DIM i1, iw, max, min AS INTEGER

INIT:
i1 = 1
max = 0
min = 4095

EVENT:
iw = ADC(1)
IF (iw > max) THEN max = iw
IF (iw < min) THEN min = iw
i1 = i1 + 1
IF (i1 > 1000) THEN
  i1 = 1

  PAR_1 = min          'Write minimum value in  'Parameter 1
  PAR_2 = max          'Write maximum value in  'Parameter 2

  max = 0
  min = 4095
  ACTIVATE_PC          'Inform PC (only for 'TestPoint)
ENDIF
```

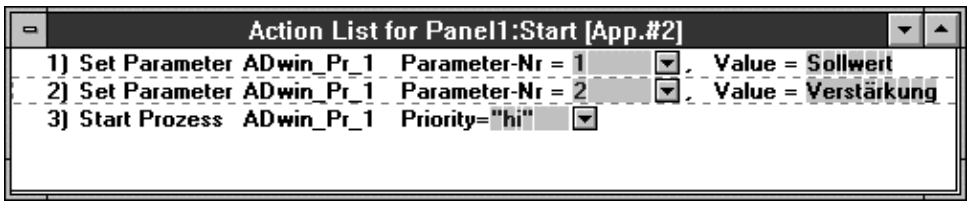


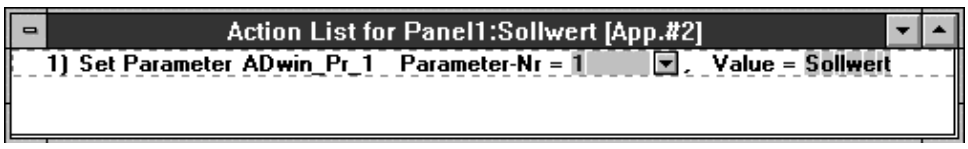


2. Digital proportional controller

```
REM The program BAS_DMO2.BAS is a digital proportional controller. The
REM setpoint is specified with PAR_1, the gain with PAR_2. This can run
REM at up to 50kHz.

DIM deviation, act AS INTEGER

INIT:
PAR_1 = 2048           'Specify initial setpoint

EVENT:
REM Compute the control deviation from the
REM difference between the specified setpoint PAR_1
REM and the momentary voltage applied to Input 1.
deviation = PAR_1 - ADC(1)

REM Compute actuating value
act = deviation * PAR_2 + 2048

REM Output actuating value on Analog Output 1
DAC(1, act)
```

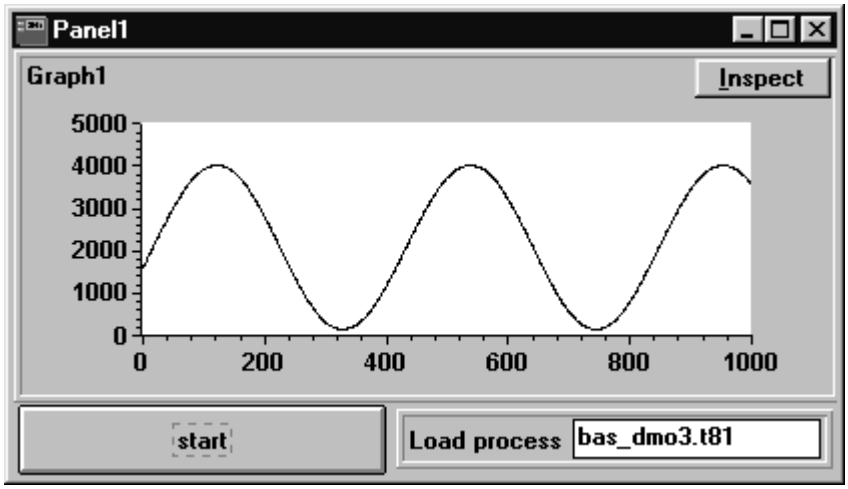


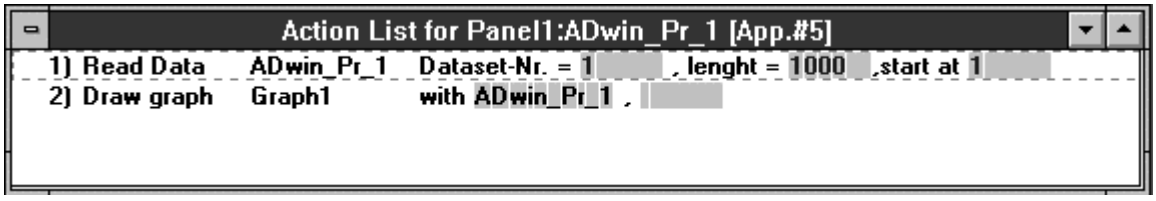




3. Data interchange with DATA

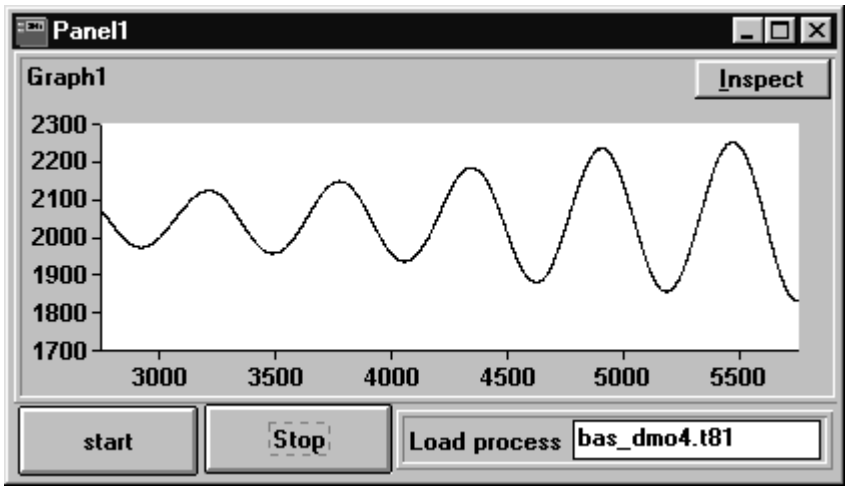
```
REM The program BAS_DMO3.BAS measures Analog Input 1 and after 1000
REM measurements informs the PC which can then fetch the measurements.
REM The data is transferred using a DATA array.

DIM DATA_1[1000] AS INTEGER
DIM index AS INTEGER

INIT:
index = 0

EVENT:
index = index+1
IF (index > 1000) THEN
  ACTIVATE_PC          'Inform PC (only for 'TestPoint)
  END                  'Terminate process
ENDIF

DATA_1[index] = ADC(1) 'Acquire measurement and  'save it in array
```



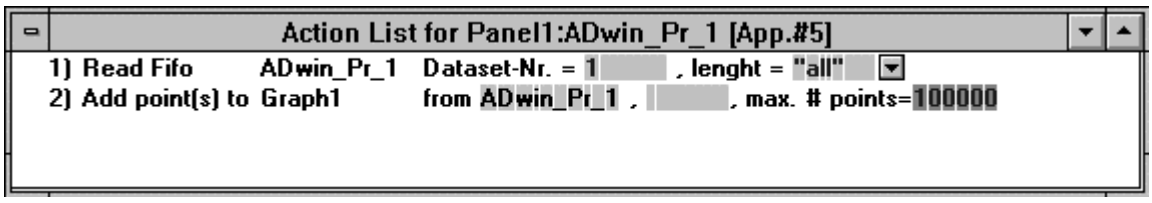


4. Continuous data transfer with a Fifo

```
REM The program BAS_DMO4.BAS continuously measures Analog Output 1 and
REM after 1000 measurements informs the PC which can then fetch the
REM measurements. The data is transferred using a FIFO.

DIM DATA_1[3000] AS LONG AS FIFO
DIM index AS INTEGER

INIT:
index = 0

EVENT:
index = index+1
IF (index > 1000) THEN
  ACTIVATE_PC          'Inform PC (only for 'TestPoint)
  index = 0
ENDIF

REM Acquire measurement and save in FIFO
DATA_1 = ADC(1)
```



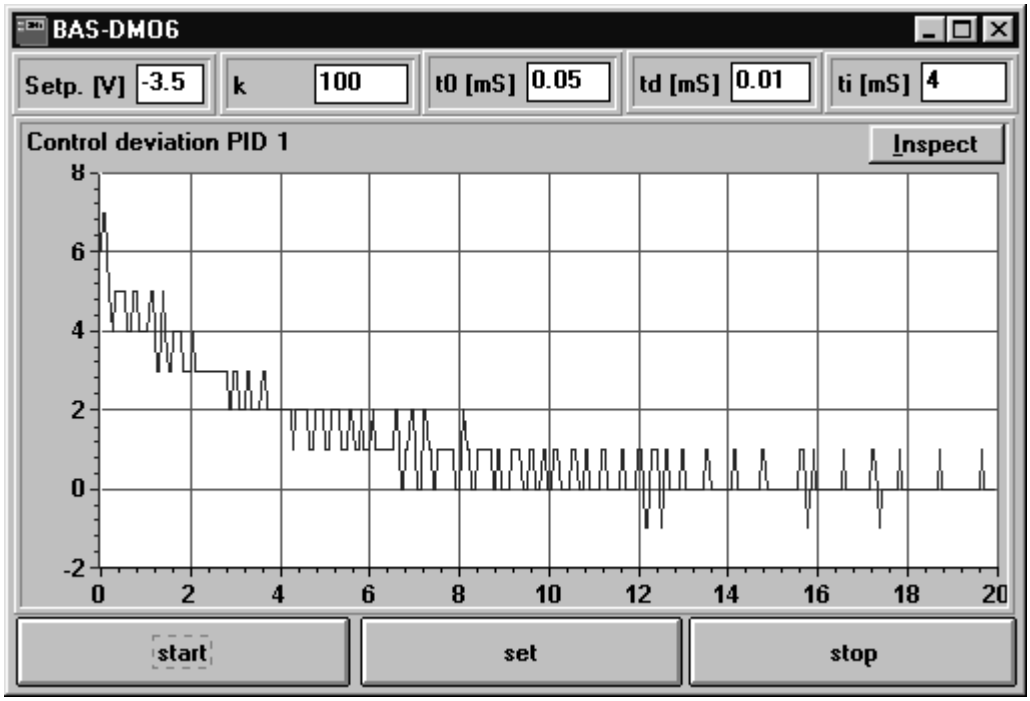
The PC is informed after every 1000 measurements. It can then read out the last 1000 measurements while the *ADwin* card continues undisturbed with the measurement of the next 1000 values. Note that the FIFO has been set to 3000 elements to allow the PC time to read out all 1000 values.



5. Digital PID-controller

```
REM The program BAS_DMO6.BAS is a digital PID-controller. The controller
REM only operates with integer variables so that the program can also be
REM used on the ADwin2 and ADwin4 cards. The controller coefficients are
REM computed on the PC, multiplied by 1000 and then transferred as
REM integer parameters to the ADwin card.

REM PAR_1 = Setpoint in digits
REM PAR_2 = q0 * 1000 = k *(1 + t0/(2 * ti) + td/t0) * 1000
REM PAR_3 = q1 * 1000 = -k*(1 + 2 * td/t0 - t0/(2 * ti)) * 1000
REM PAR_4 = q2 * 1000 = k * td/t0 * 1000
REM PAR_5 = Buffer index for the control deviation
REM PAR_9 = Flag for new setpoint definition by PC


DIM iw, deviation AS LONG
DIM setpoint, gain AS LONG
DIM ek, ek1, ek2, uk, uk1 AS LONG
DIM q0, q1, q2 AS LONG
DIM DATA_1[600] AS SHORT

INIT:
setpoint = PAR_1
q0=PAR_2
q1=PAR_3
q2=PAR_4
PAR_5 = 0
SET_MUX(0)

EVENT:
SET_MUX(0)            'Set multiplexer to Input 1
uk1 = uk + q0 * ek
START_CONV(1)         'Start measurement
dac 1, (uk1/1000)     'Output actuating value
                      'Since all parameters are multiplied by 1000,
                      'it must be devided by 1000.
ek2 = ek1
ek1 = ek
uk = uk1 + q1 * ek1 + q2 * ek2
WAIT_EOC(1)           'Wait until the ADC can be 'read out.

ek = setpoint - READADC(1)
                     'Read out and compute 'control deviation.

REM Write control deviation in a buffer
IF (PAR_5 < 500) THEN
  DATA_1[PAR_5] = ek
  PAR_5 = PAR_5 + 1
  IF (PAR_5 = 499) THEN ACTIVATE_PC
ENDIF
```

```
REM If the parameters have been changed, the
REM new parameters must be passed.
IF (PAR_9 = 1) THEN
  setpoint = PAR_1
  q0 = PAR_2
  q1 = PAR_3
  q2 = PAR_4
  PAR_9 = 0
  PAR_5 = 0
ENDIF
```



The Testpoint program use the controller parameters to calculate q0, q1 and q2.
These values are multiplied by 1000 to receive integer values that are transfered to the ***ADwin*** board as Parameter 2, 3 and 4.