

Overview

CoboldPC is designed as a convenient **data acquisition (DAq)** and **online/offline data analysis (DAn)** program for Windows NT based operating systems such as Windows Vista, XP, 2003, 2000, NT in their x86 flavor. x64 systems are only supported for Windows 7. All files for x64 do include "x64" in the filename.

Due to its modular structure it can potentially address almost any hardware and allows very elaborate data treatments as the analysis plug-in can be modified by the user. The interface between the main program and the hardware is done by a **dynamic link library named normally *DAq*.dll (called DAq module)** while the **analysis** is performed by another link library named normally ***DAn*.dll (called DAn module)**.

With this program package you have received a DAq module and at least one standard example analysis DAn module suitable for the purchased hardware. You also received a start-up batch file (CoboldCommandFile, filename.ccf) which will enable you to easily start your first **CoboldPC** session (e.g. DAq-hardware read-out with simultaneous on-line control via the example DAn module) and already learn about the most frequently used **CoboldPC** commands. On the installation CD you'll find a folder named *CoboldPC2011SampleFiles*. In there are some LMF (List Mode Files) taken by different hardware, i.e. TDC8HP, HM1 etc., as well as compressed (dcfz) document files (taken during an online or offline session).

If additional DAq modules are necessary due to changes of the hardware setup please contact **Roentdek**. You may change the DAn module using the MS-Visual C++ compiler. In Sources folder (if installed) there are sample projects for the Visual Studio 2010 Professional (or higher) compiler from Microsoft. The free Visual Studio Express versions are not sufficient to compile these sample projects!

The acquired data (events) received from the DAq hardware can be stored on disc in a list-mode format (ListModeFile, filename.lmf). The data are stored event by event (consisting of a row of so called coordinates simultaneously acquired) to re-run the analysis offline after the experiment. The size of the ListModeFile is roughly defined by the number of events acquired and the number of coordinates per event.

The online or offline calculation, performed by the DAn module, and sorting procedure, performed by the **CoboldPC** program and defined typically by a CoboldCommandFile-script, results in a number of one- and two-dimensional spectra showing the acquired data. These spectra can also be stored to disc in the so called (DumpCoboldFile, filename.dcf) or the compressed/deflated version filename.dcfz or filename.dcfld), printed, exported to other programs (text-editors, display programs or other data analysis programs) or exported as plain ASCII via *Cut and Paste*. You can also import ASCII-files in proper format to **CoboldPC** for data analysis in order to take advantage of the mathematical routines defined in **CoboldPC**.

All **CoboldPC** commands are listed and explained in the **CoboldPC** Help File section *Overview of commands*.

List-Mode

The program **CoboldPC** is a C++ program to analyse List-Mode-Data. **CoboldPC** stands for "Computer Based Online offline Listmode Dataanalyser".

List-Mode is a special technique used in collision-physics. In this mode all acquired information (named coordinates in **CoboldPC**) is stored event by event in a data-list (see Figure 1).

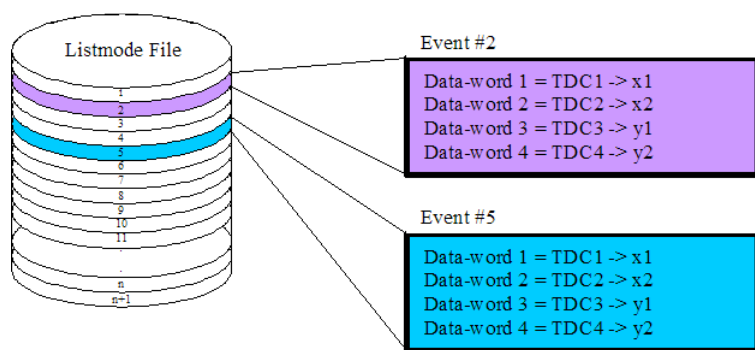
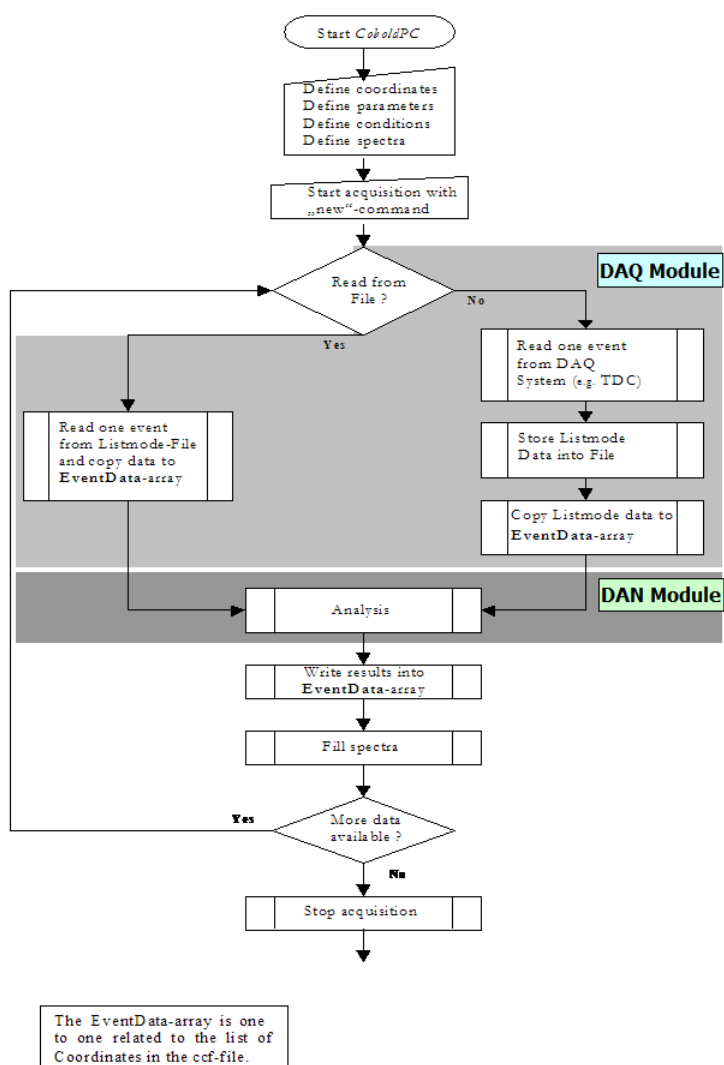


Figure 1: Schematic of a Listmode-File

Either during data acquisition (on-line mode) or after the experiment (off-line mode) the list is processed in **CoboldPC**. The program allows to sort and to display the information. To combine information of one event complex calculation and mathematics can be performed on the original event-data using the DAn module. During this process new coordinates are created and also sorted and displayed by the **CoboldPC** program.

Figure 2 shows the flow chart of the **CoboldPC** data taking and analysis.

Figure 2: Flowchart of a **CoboldPC** session

Organisation

CoboldPC is the main program that controls two sub-programs (DAn module and DAq module). For analysis and data acquisition the main program needs these two additional program parts. The DAn module (a DLL) is responsible for the calculations on the original List-Mode-Data. The DAq module (also a DLL) is handling all communications with the data acquisition hardware (e.g. a TDC). It will also handle the writing of the List-Mode-Data to a file if this was selected by the User. Please see chapter *ListMode File Data Format* for details of the LMF main header.

Quick Startup

Start the **CoboldPC** (either x86 [32Bit] or x64 [64Bit]) program out of the **CoboldPC 2011** program group as shown in Figure 3. On x86 Windows Systems the x64 executables and DLLs are not installed.

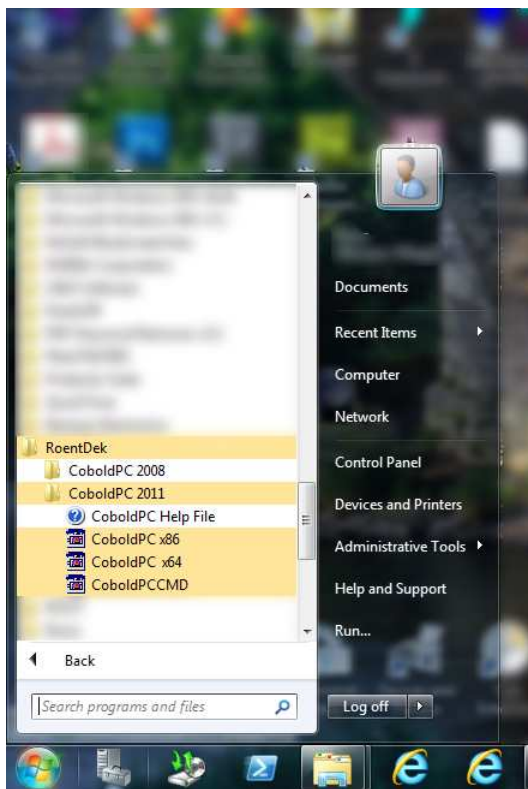


Figure 3: Start **CoboldPC 2011** program from Start-button under RoentDek-**CoboldPC 2011** group (here on an x64 Windows System)

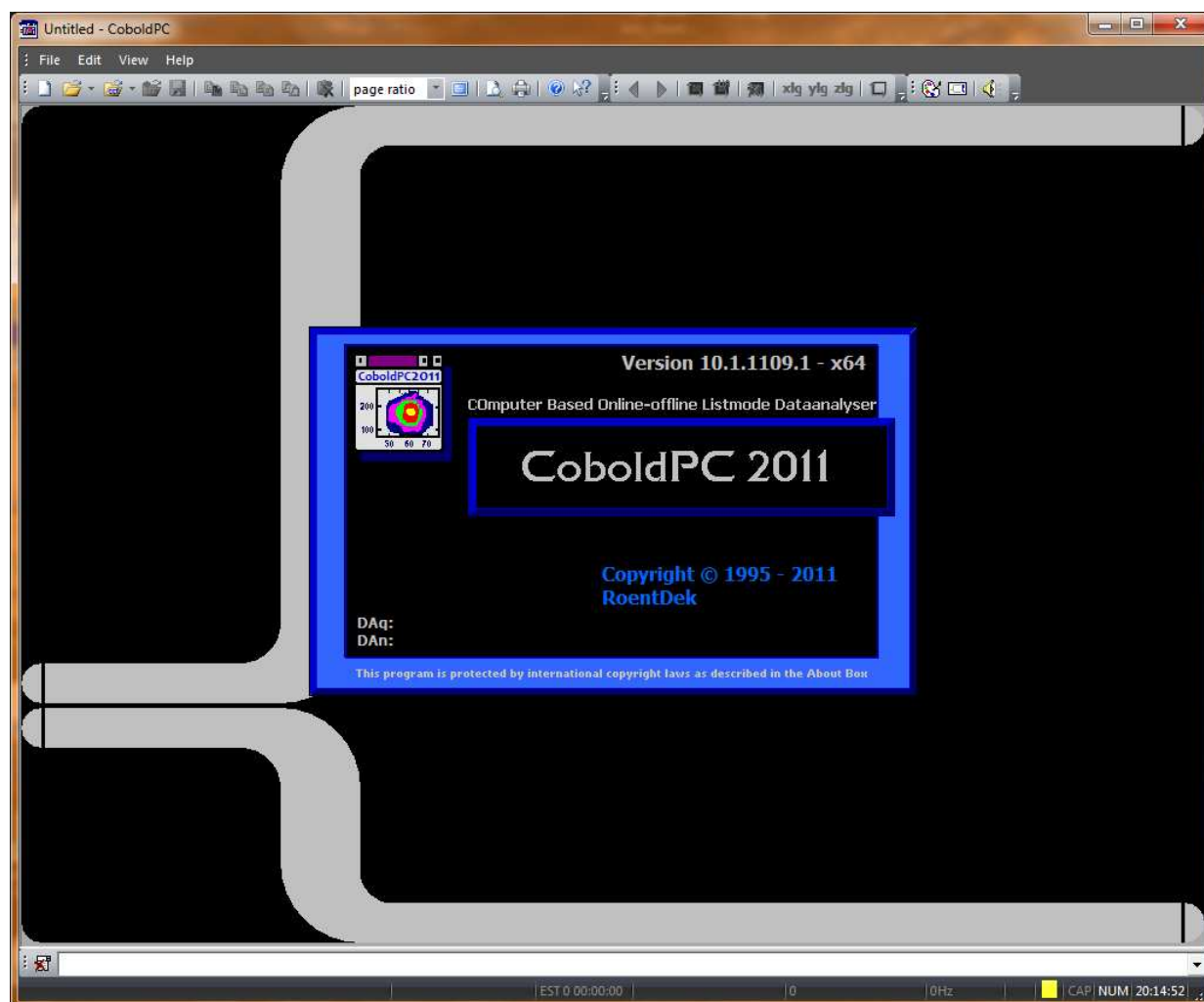


Figure 4: Startup windows of **CoboldPC 2011** (here the x64 version)

Getting Started

You may now start a **CoboldPC** session. At first start you'll get two information boxes showing that there is no DAq and DAn module loaded (Figure 5 and 6).

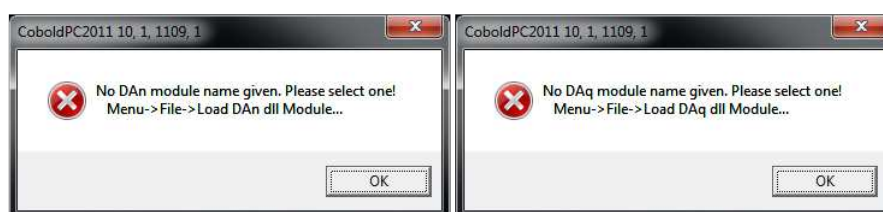


Figure 5 and 6: Showing missing DAn and DAq module

Please follow the given instructions in those boxes to load a DAq module, appropriate to your DAq hardware, and DAn module. Those modules are DLLs and can be found in the directory where you have installed **CoboldPC**.

For further testing make sure the hardware is working properly and data is coming in.

You may change, remove or add spectra or condition definitions and observe the effect of your changes of the ccf-file. If you change parameters make sure you don't supply parameter combinations that are not in accordance with the requirements of the DAn or DAq module routines.

The spectrum numbers will be attributed to the spectra in the order they have been defined.

Note that each spectrum definition will allocate a certain portion of the PC-RAM according to the number of channels and bin-size. Especially 2D spectra might require a large amount of RAM. As the RAM is limited you should choose spectra definitions with minimal size and tolerable bin-size (i.e. spectral resolution) for your respective information needs from this spectrum. The save command will store all spectra information (definition and actual content).

You may store graphics or ASCII of single spectra by clicking with the right mouse button on the displayed spectra (follow pop-up menu instructions). Similar you may import ASCII in proper format to predefined spectra. You may also print a displayed spectrum, colour or gray mode are optional (see *mode2D* command)

Note again the difference between storing the acquired data in CoboldListMode format (*.lmf) and CoboldDocumentFile format (*.dcf, *.dcfz, or *.dcfd)

The *.lmf will store the hardware data in a list if you have defined a filename as parameter in the *new* command. This file contains the basic information content of your hardware acquired data. To extract the information you run a **CoboldPC** session with defined spectra and condition, on-line (while acquiring and storing the data in *.lmf) or off-line (when analyzing the data in the list-mode file afterwards). Then you eventually store the **CoboldPC** session with the spectra content, etc. to disc. This is a snapshot of the data treatment as performed so far. From this save you will be able to do fit routines or spectra calculus later, but there is no way to do changes of conditions for spectra or parameters for the analysis from this CoboldDocumentFile (*.dcf, *.dcfz or *.dcfd).

To re-run the data analysis offline with all options you need the .lmf-file which was written during data acquisition body> File is roughly defined by the number of events acquired and the number of coordinates per event.

The online or offline calculation, performed by the DAn module, and sorting procedure, performed by the **CoboldPC** program and defined typically by a CoboldCommandFile-script, results in a number of one- and two-dimensional spectra showing the acquired data. These spectra can also be stored to disc in the so called (DumpCoboldFile, filename.dcf or the compressed/deflated version filename.dcfz or filename.dcf), printed, exported to other programs (text-editors, display programs or other data analysis programs) or exported as plain ASCII via *Cut and Paste*. You can also import ASCII-files in proper format to **CoboldPC** for data analysis in order to take advantage of the mathematical routines defined in **CoboldPC**.

All **CoboldPC** commands are listed and explained in the **CoboldPC** Help File section *Overview of commands*.

System Requirements

The minimum system requirements to run **CoboldPC** are:

- Standard Intel (Pentium) based Computer
- installed memory of 512MB on x86 systems, 1GB on x64 systems
- WindowsNT x86 based operating system (Windows 7, Windows Vista, Windows XP, Windows Server 2008, Windows Server 2003, Windows 2000, WindowsNT)
- WindowsNT x64 based operating system (Windows 7, Windows Server 2008 R2)
so far only TDC8HP is supported for x64 operating systems
- Internet Explorer 8.0 or higher
- latest service releases/service pack and updates for the operating system (see. windows update at the microsoft web site).

We recommend the following system

- >= Standard Intel Pentium 4 based Computer with >2,5GHz
- >= 1GByte memory
- Windows XP or Windows 7 with all available service packs

It is possible that **CoboldPC** will not run on Intel compatible PCs. (For example special japanese PCs)

After Installation Requirements

Precompiled versions of DAq and DAn modules (as DLLs) can be found in the folder where you have installed **CoboldPC**. When you first start **CoboldPC** no DAq or DAn module is defined for loading. Please specify a DAq and DAn module as described in the *Getting Started* section. In the menu File->Preferences you can specify whether the last known DAq and DAn modules should loaded at startime or not. If you have ordered special versions of these modules you will find these in the *AddOn* directory of the installation CD or they will be sent to you by eMail.

COM Installation

Cobold PC uses some COM (Component Object Model) applications for i.e. save **CoboldPC** documents compressed (*.dcfz) or deflated (*.dcfd) using the .Net Framework (tm) from Microsoft.

Normally all needed COM applications are installed during the normal **CoboldPC** installation. Only in case of an unplanned update directly from RoentDek, it is necessary to register these COMs manually. If you have received such an Update you should also have got informations about "how to register COMs". If not please contact RoentDek for further instructions

If COMs are not installed correctly (or with the wrong version) you'll get a Message telling you about that problem. When you come to use the not installed COM you'll get an additional warning about the missing COM.

Introduction

The program package of **CoboldPC** consists of a main routine where the general data I/O infrastructure in Windows is embedded. A large number of data-treatment, storing and display routines are accessible. It contains exchangeable subroutines that can address different data I/O hardware (DAq module, DAq = **Data Acquisition**) and allows the advanced user to compile custom DAn module (DAn = **Data Analysis**) routines. When you start **CoboldPC** it will, is selected in the preferences, automatically load the DAq and DAn module subroutines. Now your analysis session may begin. The main constituents of a **CoboldPC** analysis session are:

- Parameters (definition is independent)
- Coordinates (definition is independent)
- Conditions (definition depends on coordinates or other conditions)
- Spectra (definition depends on coordinates, conditions as well as parameters)

These objects have to be defined now after starting the main program by typing the proper commands in the command bar, which is located at the bottom of the **CoboldPC** window. This list of commands is usually written in a script file. The script files of **CoboldPC** have the extension .ccf (Cobold Command File).

To ease the startup and the operation of **CoboldPC** at least one CCF is predefined according to your specified requirements and you received an appropriate startup set of files.

In the *CoboldPC2011SampleFiles* Sample folder of your installation CD you'll find a LMF (LMF = List Mode File) appropriate for your purchased hardware. Sample startup files (CCF) are located in your installation directory of **CoboldPC**.

Parameters

A Parameter is a floating point number that can be defined by the *parameter* command in the command line. Parameters can also be set/changed during running the analysis session. These parameters will be used within the DAq and DAn modules. This feature enables for example changing a calibration factor or something similar in the analysis without compiling the DAn module again.

Normally parameters 0-999 are partially used by the standard DAq module and parameters 1000-2999 are partially used by the standard DAn modules. To not interfere with *standard* parameter settings, please use for your custom parameters indexes above 3000.

The syntax is:

```
parameter 7,389.56
```

which gives the parameter 7 the value 389.56.

The definition of the parameters are interlinked with the *Parameters* array in the DAq and DAn modules subroutines. See the example source code for more explanation.

Coordinates

A *coordinate* is simply a variable. **CoboldPC** uses a list of these variables for the data transfer between the main program and the DAq and DAn modules. The first part of this list is filled by the DAq module. The DAq module reads out the hardware (e.g. a TDC) and writes this data into the first part of the list of coordinates. Thus you have to define at least as many coordinates as hardware channels are read out by the DAq module. The lower part of the list will be filled by the DAn module. Here all the results of the calculations performed inside the DAn module are stored. All coordinates can be displayed in spectra.

The syntax for defining the coordinates named e.g. x,y,z is:

```
coordinate x
coordinate y
coordinate z
```

or

```
coordinate x,y,z
```

This list of coordinates is one to one related to the *EventData* array that you will notice if you look at the source code of a DAn module. Actually, this array is used inside the DAq module and DAn module whenever the list of coordinates must be accessed. E.g. this is done twice inside the DAn module. First, when the raw data is read inside the DAn module and second when the results of the calculations in the DAn are written back into the Coordinate list.

Conditions

A *condition* is a one- or multidimensional restriction which is preventing data to be filled into a certain spectrum (defined with that condition as command parameter) if the condition is not fulfilled. Conditions are either single restrictions on a specific coordinate or boolean combinations of two such conditions.

The syntax is:

```
condition x,100,200,x-cond      (defining a condition on coordinate x with the name x-cond)
condition y,10,50,y-cond      (defining a condition on coordinate y with the name y-cond)
```

The syntax for boolean combinations is:

```
condition x-cond,and,y-cond,xy-cond  (defining a combined condition using logical and between x-cond and y-cond with the name xy-cond)
```

for an *and* combination. Also *or*, *not* and *xor* are allowed combinations.

Spectra

A spectrum is a 1- or 2-dimensional histogram. The required memory is allocated in the RAM of the PC. You need to define each spectrum with a set of command parameters (attributes for this spectrum) that define for example the size and the coordinate you want to visualise. In each hardware read-out cycle the spectra will be filled with new data. This happens in the background. The spectra must be manually updated in order to see the new entries.

The syntax for a 1-dimensional spectrum is:

```
define1 0,1000,5,x,,x-Axis,always,X-Spectrum  (the double-comma between x and x-Axis is correct)
```

which defines a spectrum ranging from channel 0 to channel 1000 with a bin size of 5 sorting coordinate x without any condition (always) (see next subsection), with the Spectrum name *X-Spectrum*. The title of the x-axis is *x-Axis*. It is possible to define a weight-parameter which can be controlled by the DAn.dll. However, if the standard weigh-parameter of 1.0 is used then this option can be left out (,,).

The syntax for a 2-dimensional spectrum is:

```
define2 0,1000,5,x,x-Axis,0,1000,5,y,y-Axis,,always,X-Y-Spectrum
```

sorting the x-coordinate on the x-axis and the y-coordinate on the y-axis, both ranging from 0 to 1000 with a bin size of 5 without condition.

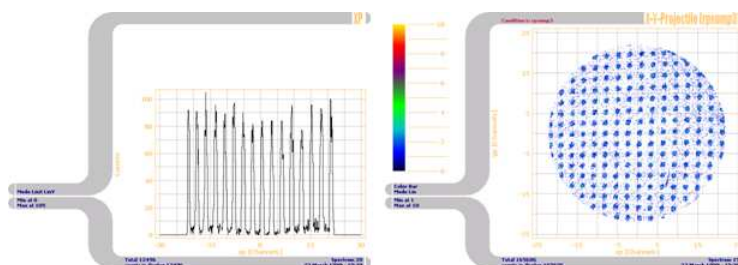


Figure 5: Example for a 1- and a 2-dimensional spectrum

Defining a spectrum with a condition:

```
define2 0,1000,5,y,y-axis,0,1000,5,z,z-axis,,x-cond,Y-Z-Spectrum
```

will result in a 2 dimensional spectrum showing the y- and the z- coordinate under the condition that the x-coordinate was between 0 and 1000.

The Startup-Online.ccf and Startup-Offline.ccf

We have prepared a startup command file startup-online.ccf and startup-offline.ccf which contains all commands for reading out the TDCs **TDC8HP**, **HM1** or **TDC8PCI2**. It provides already most of the desired definitions, i.e. 2d position spectra and time-of-flight spectra in various coordinate representations. The program calls a sub-script for defining hardware-specific parameters which must be chosen (default: **TDC8HP**). Due to this modular construction it is possible to use almost the same data analysis sequences for different hardware, i.e. TDC types. If you use other hardware than the TDC8HP you are requested to modify one command:

For **HM1**: remove the “;” in front of the command lines:

```
;execute HM1-DAQ-Parameters.ccf and ;LoadDAQModule DAq_HM1.dll,Applicationpath
```

For **TDC8PCI2**: remove the “;” in front of the command lines

```
;execute TDC8PCI2-DAQ-Parameters.ccf and ;LoadDAQModule DAq_TDC8PCI2.dll,Applicationpath
```

Generally, a “;” at the beginning of a command line disables the command, i.e. it is only considered as a comment.

The startup.ccf begins with the following commands/parameter settings common to all different hardware components, followed by hardware-specific commands, data analysis specific commands and finally commands to define spectra and begin the data acquisition (see sub-sections):

```
Restart                                reset of any earlier commands and definitions setpath
APPLICATIONPATH                        sets the file paths to the directory containing the Cobold.exe

LoadDAnModule  DAn_Standard.dll,Applicationpath    loads the DAn module
LoadDAQModule  DAq_TDC8HP.dll,Applicationpath      loads the hardware DAq module.

Parameter 2                                     For HM1 or TDC8PCI2 replace TDC8HP
Time stamp for an event as obtained from the CPU   (or TDC in case of the TDC8HP) in seconds. Setting
this parameter to 1 or 2 will record the time stamp with
each event as 32bit or 64bit value (0 = acquisition start).
Please note that the accuracy of the recorded time is not

guaranteed.                                     In case of the HM1 and TDC8PCI2 the CPU time is written with
an accuracy of a several microseconds. In case of the TDC8HP
the internal TDC clock is with an accuracy of 25 ps.
0 = no Timestamp,
1 = 32Bit Timestamp (Low.Low, Low.High)
2 = 64Bit Timestamp (Low.Low, Low.High, High.Low,

High.High)
Parameter 5                                     Time scaling (internal parameter). Used to calibrate the time
stamp.
Parameter 6                                     DAQ-version number (internal parameter)
Parameter 7                                     Start time of list mode file (internal parameter)
Parameter 8                                     DAQ-ID (internal parameter)
DAQ_ID_RAW          0x000000    for RAW (no Data)
DAQ_ID_HM1          0x000001    for HM1 (single)
DAQ_ID_TDC8         0x000002    for TDC8/ISA/PCI
DAQ_ID_CAMAC        0x000003    for CAMAC
DAQ_ID_2HM1         0x000004    for 2 HM1
DAQ_ID_2TDC8        0x000005    for 2 TDC8
DAQ_ID_HM1_ABM      0x000006    for HM1 ABM Mode
DAQ_ID_HM1_LR       0x000007    for HM1 LR Mode
DAQ_ID_HPTDC        0x000008    for TDC8HP

Parameter 9                                     LMF-version number (internal parameter)
Parameter 20                                    TDC resolution in ns. (internal parameter. Do not adjust)
Parameter 21                                    TDC data type information (internally set)
0 = Not defined
1 = Channel information
2 = Time information (in ns)

Parameter 32                                    number of channels (from 1) to be read out
Parameter 33                                    number of hits per channel to be read out
Parameter 40                                    DataFormat (Internally set)
Parameter 50                                    unique ID-number checking compatibility of CCF with DAn and
DAQ
```

Hardware specific commands: DAq-parameters and coordinates

The following commands are part of the startup*.ccf, for details please refer to the **TDC8HP** manual. For the commands in the other parameter files for **HM1** and **TDC8PCI2** please also refer to the specific manuals:

```
Parameter 50,201102080000;      check-ID (tests compatibility of CCF/DAq)

Parameter 53,1;                  display only every (n)th event but write all events to hard drive (for high rate
measurements)

Parameter 60,0;                  0 = dont read driver config file (default 0)
Parameter 61,0x000000;          RisingEnable, 0 = none (e.g. 0x40 = channel 7)
Parameter 62,0x0ff1ff;          FallingEnable, Channel 1-9 on first TDC and channels 1-8 on seconds TDC
Parameter 63,0;                  TriggerEdge, 0 = falling
Parameter 64,8;                  TiggerChannel,channel 8 for trigger
Parameter 65,0;                  OutputLevel, 0 = false
Parameter 66,1;                  GroupingEnable, 1 = true = 25ps binsize and max. ±200ms range
;                                0 = false = 16ps binsize and max. ±32ms range
Parameter 67,0;                  AllowOverlap, 0 = false (0 = default) br />
Parameter 68,310;                TriggerDeadTime, time in ns (recommended value: 10ns more than parameter 70)
Parameter 69,-300;               GroupRangeStart, time in ns
Parameter 70,300;                GroupRangeEnd, time in ns
Parameter 71,0;                  External Clock, 0 = false (0 = default, should be 1 if two TDCs are synched)
Parameter 72,1;                  OutputRollovers, 1 = true (1 = default)
Parameter 78,1;                  VHR (TDC8HP only: 0 = 100ps LSB, 1 = 25ps LSB)
Parameter 79,0.2;                Group Timeout in seconds (default 0.2s, do not adjust)
Parameter 80,0;                  INL, 0 = false = do not read file (default)
Parameter 81,0;                  DNL, 0 = false = do not read file (default)

Parameter 86,1;                  MMXEnable (never set to 0, always 1)
Parameter 87,1;                  DMAEnable (never set to 0, always 1) br />
Parameter 88,0;                  time zero channel:
;                                set all trigger times (parameter 64) relative to last hit in this channel
;                                please set to 0 if not used
;                                "Grouping" must be disabled (parameter 66)
Parameter 89,0x00000000;         Trigger channel mask (active only when parameter 66 is set to 0)
```

From these DAq parameters the following are of special interest and are therefore explained also here:

```
Parameter 53                      NumberOfDAQLoops (normally 1)
                                  To increase online event reading speed you can increase this value.
                                  If set to n then you'll process n Events in the DAq module before
                                  returning the last event to CoboldPC. In online processing only
                                  the nth event will be passed to the analysis but all events will
                                  be written to the LMF.

Parameter 64                      TriggerChannel: Determines the trigger input channel (1-9)
Parameter 68                      TriggerDeadTime: Defines the time range in ns before a next signal
                                  in the trigger channel will be considered as the trigger for the
                                  next event. Recommended value: 10ns more than parameter 70.
Parameter 69                      GroupRangeStart: from this time (in ns, minimum -2E5) relative to the
                                  trigger signal all signals in the other channels are registered as
                                  belonging to the same event until
Parameter 70                      sets the GroupRangeEnd as time in ns (maximum 2E5) after the trigger for this event.

Parameter 78                      VHR-flag: Enables the TDC8HP's very high resolution mode with an
                                  LSB of 25ps (set to 1). Per default this parameter is set 1
                                  (25ps LSB). For multi-hit measurement with short dead-time demand
                                  it is recommended setting this parameter to 0.

UserFCall,SetDAQCoordinates,T1Ch??n,T1Ch??S?? defines coordinates for number of hits per
                                              channels and values for each hit according
                                              to parameters 32 and 33. This command
                                              (for parameter 32 = 8 and parameter 33 = 2)
                                              is the equivalent to the following "manual"
                                              coordinate definition block:
Coordinate T1Ch01n,T1Ch01S01,T1Ch01S02      note that it is possible to define several coordinates in
one                                           command line separated by ",".
Coordinate T1Ch02n,T1Ch02S01,T1Ch02S02      Example: T1Ch02n is the coordinate for the number of hits
Coordinate T1Ch03n,T1Ch03S01,T1Ch03S02      in TDC channel 2
                                              T1Ch03S01 is the coordinate for the value of hit 1in
Coordinate T1Ch04n,T1Ch04S01,T1Ch04S02      TDC channel 3
Coordinate T1Ch05n,T1Ch05S01,T1Ch05S02      T1Ch04S02 is the coordinate for the value of hit 2in
Coordinate T1Ch06n,T1Ch06S01,T1Ch06S02      TDC channel 4
Coordinate T1Ch07n,T1Ch07S01,T1Ch07S02
Coordinate T1Ch08n,T1Ch08S01,T1Ch08S02
```

It is mandatory that the number and order of these so-called DAq coordinates are in accordance with the algorithms in the DAq.dll and the DAn.dll and also with the settings of parameters 32 and 33. The hardware coordinates and the time stamp coordinate (optionally) are stored in the list mode file if this function is enabled before the data acquisition starts (see new command).

The previous parameters of the DAq part have the function to define and organize the hardware (and are mandatory), the set of so-called DAn parameters is used in the data analysis part. During offline analysis of an earlier acquired list mode file some of these parameters are automatically set from the parameter information (settings during data acquisition) that is stored in the header of the list mode file.

For further computations with the obtained raw data (DAq coordinates), the DAN.dll as a data analysis subprogram uses these DAq coordinates and creates computed coordinates (DAn coordinates), such as the position or time sum (TOF) derived from the raw data. It also comprises some correction, shifting and rotation computations and coordinate system transformations, so that the basic computations for experiments with a position and time sensitive detector are already available without changing the DAN.dll as it was supplied with the Cobold program. The DAN-coordinates are internally treated as independent coordinates and are internally listed by numbers, following the last DAq coordinate. However, the DAN coordinates will not be appended in the list mode file. The DAN.dll may be altered using the Microsoft-C++ compiler 10 (Visual Studio 2010 Professional) or higher. The list of coordinates may be changed (with any text editor like Microsoft notepad.exe), creating additional coordinates (and parameters) for further computation, unused DAN coordinates may be removed. Any newly defined coordinate is available for further computations. Note that the program will only operate normally, if all definitions are in accordance with the DAq.dll and DAN.dll used. After the new and start commands the program makes a consistency check and may give an error message if the number of coordinates and parameters defined are not sufficient, however, it will not detect all possible discrepancies.

Analysis specific commands: DAN parameters and coordinates

The parameters from 1000 onwards (up to 1999) are reserved for DAN-parameters. Note that some parameters (for DAq and DAN) are set automatically or values may be overwritten when reading a previously recorded list mode file. Private Parameters should not start before 2000!

The following DAN parameters used in the DAN part can have the function of variables for computations, of pointers or of flags. Some are mandatory, some are optional. The Standard DAN will use the parameter range 1000-1999.

```
Parameter 1050      check-ID (tests compatibility of CCF/DAN, must be set to 201102080000

Parameter 1000      internal DAN calibration parameter, do not change
Parameter 1002      Hexanode calculation flag      0 = no Hexanode
                                                         1 = Hexanode
                                                         If a Hexanode is used additional calculations are required to retrieve
                                                         the position information. For these parameters and coordinates please
                                                         refer to the add-on manual.
Parameter 1003      R-Phi coordinate conversion      0 = RAD [-p..p]
                                                         1 = RAD [0..2p]
                                                         2 = DEG [-180..180]
                                                         3 = DEG [0..360]
                                                         This parameter defines the angular range and unit for the phi coordinate
                                                         in the R-Phi representation of the 2d-image.
Parameter 1004      (not used for TDC8HP) This pointer value defines for the DAN program part
                                                         the position in the coordinate list where the first of the TDC data
                                                         appears (s1). Usually you can set this value also to 0 and the program
                                                         will automatically enter the right number.
Parameter 1005      Start of DAq Data for DAN
Parameter 1006      Start of DAN Data This pointer value defines the position in the coordinate
                                                         list where the DAN coordinates begin, i.e. it should equal the number of
                                                         hardware coordinates If you want to analyze the data from the first hit you
                                                         can set this value also to 0 and the program will automatically enter the
                                                         right number.
Parameter 1007      Hit number to be analyzed. Usually the position coordinates are calculated
                                                         from the first hit in the TDC channels (default value: 1). If you instead
                                                         want to get position and time sum coordinates calculated for a different
                                                         hit number you have to enter the hit value here. Note that it can happen
                                                         that the values from different channels do not necessarily correspond to
                                                         the desired particle hit number if reflections on the raw amplifier signals
                                                         or missed signals produce "false" hits in a certain TDC channel.
Parameter 1010      pTPCalX Time to Position calibration factor for PosX coordinate (v1 in mm/ns)
                                                         DLD40: 1.32, DLD80: 1.02, DLD120: 0.77
                                                         For Hexanode* (u): HEX80: 0.737, HEX120: 0.583
                                                         * Please note that it is required to calibrate these numbers for your
                                                         anode more accurately. Please contact service@roentdek.com
Parameter 1011      pTPCalY Time to Position calibration factor for PosX coordinate (v1 in mm/ns)
                                                         DLD40: 1.43, DLD80: 1.13, DLD120: 0.82
                                                         For Hexanode* (v): HEX80: 0.706, HEX120: 0.567
                                                         These two parameters define the value of position to time
                                                         calibration, the effective signal propagation speed
                                                         across the delay-line. It depends on the size and geometry of the
                                                         delay-line used. The suggested values are only accurate within few percent
                                                         for a given delay-line. If a higher precision is needed one needs to make
                                                         a position calibration with a test mask in front of the detector. If the
                                                         detector image boundary has an oval shape, exchange the values for X and Y
                                                         (only for DLD) and try again sorting the data (may be the physical
                                                         dimensions of the anode have been exchanged during mounting).
                                                         * Please note that it is required to calibrate these numbers for your
                                                         anode more accurately. Please contact service@roentdek.com
Parameter 1012      pTPCalW Time to Position calibration factor for the w-layer (Hexanode only)*:
                                                         HEX80: 0.684, HEX120: 0.540
```

Parameters 1013 to 1019 can be used for defining bin sizes of spectra (see spectra definition commands below)

```
Parameter 1013      should be 1/4th of pTPCalX use for TDC8PCI and TDC8HP
Parameter 1014      should be 1/4th of pTPCalY use for TDC8PCI and TDC8HP
Parameter 1015      should be 1/20th of pTPCalX only for TDC8HP (for HM1 use 1/15 or multiples of pTPCalX)
Parameter 1016      should be 1/20th of pTPCalY only for TDC8HP (for HM1 use 1/15 or multiples of pTPCalY)
Parameter 1017      should be 1/80th pTPCalX only for TDC8HP and para 78 set to 1
Parameter 1019      should be 1/80th pTPCalY only for TDC8HP and para 78 set to 1
```

Parameter 1020 pCOx Rotation Offset Center for coordinate PosX
 Parameter 1021 pCOy Rotation Offset Center for coordinate PosY
 These parameters define the center point for an online detector image rotation and also the center point in the X/Y plane for a coordinate transformation into R/Phi representation. Note that R/Phi transformation will only give good results if the position unit is mm (see parameter 1000).
 Parameter 1022 pRotA Rotation Angle in mathematical direction (counter clock wise) for an online detector image rotation (value to be supplied in RAD or DEG depending on parameter 1003)
 Parameter 1023 X-value of center for r/phi coordinate computation
 Parameter 1024 Y-value of center for r/phi coordinate computation
 Parameter 1025 MCP channel number. If the trigger signal is NOT the MCP signal the parameter 1025 shall contain the channel number of the MCP signal to achieve sum spectra with the time reference set to the MCP signal.

Parameters 1026 to 1032 can be used to change the assignment of the "raw" DAQ coordinates from TDC channels to the calculated (DAn) coordinates x1, x2, ... p;

Parameter 1027 channel number for x1
 Parameter 1026 channel number for x2
 Parameter 1028 channel number for y1
 Parameter 1029 channel number for y2
 Parameter 1030 channel number for z1 (ignored if parameter 1002 = 0)
 Parameter 1031 channel number for z2 (ignored if parameter 1002 = 0)
 Parameter 1032 channel number for TOF (-1 if not used))
 Parameter 1035 pOPx Offset for PosX
 Parameter 1036 pOPy Offset for PosY
 These two parameters are offset (additive) constants for shifting the detector image in the X/Y plane. Note, that in case of the Hexanode these values define the offsets for the calculated x and y and not for the raw u and v values.
 Parameter 1037 pOPw Offset for third anode layer (added to w, only for Hexanode)
 Parameter 1038 pOSum Offset for Sum/Diff coordinate calculations.
 This offset value is an additive constant to all time Sum/Diff coordinates
 Parameter 1040 Reset EventCounter (1: reset after "new" command, 0: no reset)
 Parameter 1041 Integration time in seconds for RealTriggerRate coordinate
 Parameter 1060 Condition flag: value will appear as value in coordinate Condition1

The following DAn coordinates are by definition only the additional coordinates that are computed from the (raw) DAQ coordinates retrieved from the hardware or from a previously accumulated event file (not part of a list mode file). Here, only one set of delay-line coordinates for one of the hits (default: first hit, see parameter 1005) is selected and position and time coordinates are calculated. If you have changed parameter 2, 32 or 33 from their default value (first hit only) or if you sort a list-mode file acquired with a non-default parameter settings you need to adjust the (pointer) parameters 1005 and 1006. It is such possible to apply the position and time calculations to the next hits if such are (or have been) acquired. The DAn.dll will read the values of the status registers and the value in the 4 (Hexanode: 6) raw position coordinates (and optional TOF) defined by parameter 1005 (default: first hits) and calculate the desired position and time information. Note that even for the use of a DLD (4 delay-line signals only), the coordinates for two additional delay-line signals (as from a Hexanode) are defined but set to 0. A set of DAn coordinates is created by using the defined set of DAQ coordinates:

Coordinate AbsoluteEventTime defines the absolute event time and
 Coordinate DeltaEventTime the time between one event and the next
 Coordinate EventCounter event number since start or last event number reset
 Coordinate True internal coordinate
 Coordinate False internal coordinate
 Coordinate ConsistencyIndicator **The value of this coordinate is: $\sum u \cdot 2^{i-1}$, i is the TDC channel, u =1, if at least one hit in the TDC channel i was registered, otherwise 0. If each TDC-channel for the selected hit number has received at least one hit of the value is 15 for a DLD and 63 for a Hexanode. This assumes that the first TDC channels are used for the delay-line signals.**
 Coordinate PLLStatusLocked internal coordinate for HML (see manual), **must always be defined**
 Coordinate RealTriggerRate calculates the trigger (count) rate (please see parameter 1041)
 Coordinate Condition1 the value of this coordinate is set by a condition command
 Coordinate n1,n2,n3,n4,n5,n6,n7,n8 number of hits in the TDC channels 1-8 (not higher than parameter 33)
 Coordinate x1,x2 values in the TDC channels 1-6, "position calibrated" in mm.
 Coordinate y1,y2 calibrated
 Coordinate z1,z2 values (in ns).
 Coordinate TOF values in the TDC channel 7, time calibrated in ns
 Coordinate raw_x,raw_y,raw_w difference of TDC channel values 1/2, 3/4, 5/6 (uncalibrated)
 Coordinate raw_sumx,raw_sumy,raw_sumw sum of TDC channel values 1/2, 3/4, 5/6 (uncalibrated)
 Coordinate raw_sumxyw sum of TDC channel values 1 to 6 (uncalibrated)
 Coordinate raw_diffxy sum of TDC channel values 1/2 minus sum of 3/4 (uncalibrated)
 Coordinate sumx,sumy,sumw,sumxyw same as raw_sum... but calibrated in ns and shifted (parameter values) example: sumx = x1 +x2 + pOSum
 Coordinate diffxy same as raw_diffxy but calibrated in ns and shifted (parameter values)
 Coordinate PosX,PosY calibrated position coordinates after shift/rotation (parameter values) example: PosX = x1 - x2 + pOPx, and possibly rotated (for Hexanode: Xuv + pOPx and Yvw + pOPy)*
 Coordinate r,phi calibrated position coordinates in R/Phi coordinate system
 Coordinate Xuv,Yuv,Xuw,Yuw,Xvw,Yvw **only for Hexanode:** calibrated position coordinates retrieved from the respective two layers
 Coordinate dX,dY control coordinates: difference between Xuv/Xvw and Yuv/Yvw
 Coordinate reflection_in_MCP control coordinate: time between second and first hit in TDC channel 8 (MCP) in ns.
 Coordinate reflection_in_x1,reflection_in_x2 control coordinates: time between hit on one delay-line contact and

```
Coordinate reflection_in_y1,reflection_in_y2      second hit on the other contact of the same delay-line, for all
layers and
Coordinate reflection_in_z1,reflection_in_z2      all ends (the latter two only for Hexanode
Coordinate Const1,Const2,Const3,Const4,Const5,Const6,Const7,Const8      internal coordinates for the constants 1 to 8

CoordinateSet n_matrix_y,T1Ch01n,T1Ch02n,T1Ch03n,T1Ch04n,T1Ch05n,T1Ch06n,T1Ch07n,T1Ch08n
CoordinateSet n_matrix_x,Const1,Const2,Const3,Const4,Const5,Const6,Const7,Const8
```

The keyword "CoordinateSet" combines several coordinates in a group. In the example above the coordinates T1Ch01n to T1Ch08n are combined in a group with the name "n_matrix_y". In the histogram definitions these group names can be used as if they were normal coordinates. Thus the command

```
define2 0.,9.,1.,n_matrix_x,channel number,0.,8.,1.,n_matrix_y,counts,none,always,hit statistics
```

results in a 2D-histogram which is filled at the following 8 bin positions: x=Const1 / y=T1Ch01n, x=Const2 / y=T1Ch02n, x=Const3 / y=T1Ch03n, x=Const4 / y=T1Ch04n, x=Const5 / y=T1Ch05n, x=Const6 / y=T1Ch06n, x=Const7 / y=T1Ch07n, x=Const8 / y=T1Ch08n

Spectra and condition definition commands

The final purpose of the data acquisition is to display and analyze the acquired data. For this purpose it is possible to define spectra for displaying all defined coordinates. A spectrum is a histogram with a fixed bin width either with a one- or two dimensional array of "bins". For a one-dimensional spectrum (for example a time spectrum) this array is a row along the ordinate (X-axis) of a graph, the bins correspond to the values of the corresponding coordinate. When data is acquired or read from a list-mode file, the value of the coordinate for each event will be attributed to the closest bin's value and the histogram content in this bin will be incremented by one unit (along the Y-axis of the graph). For example such a histogram (spectrum) could show the distribution of time of flight values for a number of acquired events.

Likewise it is possible to display two-dimensional spectra, i.e. the coincident occurrence of values in two coordinates within the corresponding bin widths (for example the 2d position distribution of detected particles). To visualize such a histogram the two coordinates span a plane (X/Y), the value in each bin (Z) is displayed as gray or color scale, also contour lines or scatter plots can be used for the display. The range of the displayed spectra in X, Y (and Z), the bin size and the "unit" of incrementing can be defined for optimal visualization and manipulation.

To analyze higher dimensional coordinate correlations it is possible to "gate" the sorting process into a histogram (spectrum) by defining a condition for this spectrum. Such a condition can be a "window" (or "region of interest") on the occurrence of a certain range of values in a third coordinate for the events. For example: if one needs to visualize the (2d) position spectra of particles as function of their time-of-flight (TOF) one can define several conditions (gates) on the TOF coordinate (e.g. time sum peaks) and several 2d position spectra with the different conditions. It is possible to couple different conditions (e.g. by an "AND") to allow the analysis of even higher dimensional coordinate correlations.

For details about the definition of spectra and conditions, for spectrum manipulation options and data I/O to other programs please refer to the **CoboldPC** manual. In the following you find some pre-defined conditions (as examples) and spectra as part of the startup.ccf that will allow you to view the most important coordinates. For example, you will immediately be able to see a position spectrum. You may later edit the startup.ccf and all sub-scripts to adjust them to your needs, e.g. setting the right condition gates on the time sum peak(s), omitting spectra that you do not need, adjust parameters (for shifting or rotating the spectra, calibrating position and time), changing or appending spectrum definitions. For the Hexanode an extra software package is available to optimize its function. Please contact **RoentDeK** on the availability of specific software packages for your application.

The following condition and spectra definition commands are recommended for first time users. Those definitions disabled by the ";" may also be of use and can be activated by removing the ";" in front of each command line:

```
condition ConsistencyIndicator,14.5,15.5,four;      true if x1,x2,y1 and y2 signals were registered
condition ConsistencyIndicator,62.5,63.5,six;      true if x1,x2,y1,y2 and z1,z2 signals were registered
condition four,or,six,clean_hit
```

Note that for the Hexanode it is recommended to rename the condition six by clean_hit here and comment out 2 lines:

```
;condition ConsistencyIndicator,14.5,15.5,four;      true if x1,x2,y1 and y2 signals were registered
condition ConsistencyIndicator,62.5,63.5, clean_hit; true if x1,x2,y1,y2 and z1,z2 signals were registered
;condition four,or,six,clean_hit

condition sumx,1,10000,sumx
condition sumy,1,10000,sumy
;condition sumy,1,10000,sumw
condition sumx,and,sumy,sumxy
```

This defines a more specific filtering for "clean" events. The boundary parameters in the condition for sumx and sumy conditions should be narrowed according to the actual time peak widths/positions.

```
new
;new hardware
```

This command is defining the type of session but also validates/checks many parameters and coordinate commands on consistency.

```
define1 0,32,1,T1Ch01n,,none,always,T1Ch01n
define1 0,32,1,T1Ch02n,,none,always,T1Ch02n
define1 0,32,1,T1Ch03n,,none,always,T1Ch03n
define1 0,32,1,T1Ch04n,,none,always,T1Ch04n
define1 0,32,1,T1Ch05n,,none,always,T1Ch05n
define1 0,32,1,T1Ch06n,,none,always,T1Ch06n
define1 0,32,1,T1Ch07n,,none,always,T1Ch07n
define1 0,32,1,T1Ch08n,,none,always,T1Ch08n
define1 -2,66,1,ConsistencyIndicator,,none,always,ConsistencyIndicator
define2 0.,9.,1.,n_matrix_x,,0.,8.,1.,n_matrix_y,,none,always,hit statistics
```

These spectra display the number of hits per TDC channels in various representations.

```
define1 -12000,12000,1,T1Ch01S01,T1Ch01S01 (x1 raw),none,always,T1Ch01S01,,true
```

```
define1 -12000,12000,1,TlCh02S01,TlCh02S01 (x2 raw),none,always,TlCh02S01,,true
define1 -12000,12000,1,TlCh03S01,TlCh03S01 (y1 raw),none,always,TlCh03S01,,true
define1 -12000,12000,1,TlCh04S01,TlCh04S01 (y2 raw),none,always,TlCh04S01,,true
define1 -12000,12000,1,TlCh05S01,TlCh05S01 (z1 raw),none,always,TlCh05S01,,true
define1 -12000,12000,1,TlCh06S01,TlCh06S01 (z2 raw),none,always,TlCh06S01,,true
define1 -12000,12000,1,TlCh07S01,TlCh07S01 (TOF raw),none,always,TlCh07S01,,true
;define1 -12000,12000,1,TlCh08S01,TlCh08S01 (Trigger),none,always,TlCh08S01,,true
```

These spectra show the “raw” (uncalibrated) values of the first hits in the TDC channels

```
;define1 0,10000,1,AbsoluteEventTime,AbsoluteEventTime [s],none,always,Time since Start,,true
define1 0,0.005,0.00001,DeltaEventTime,DeltaEventTime [s],none,always,Time between Events,,true
define1 0,10000000,1000,EventCounter,,none,always,EventCounter
define1 0,100000,10,RealTriggerRate,,none,always,RealTriggerRate,true
;define2 0,100000,100,AbsoluteEventTime,,0,100000,100,RealTriggerRate,,none,always,Rate (time)
;define2 0,1000000,1000,EventCounter,,0,100000,100,RealTriggerRate,,none,always,Rate (eventnumber)
```

These spectra give information on the trigger (count) rate.

```
define1 -1000,9000,4,raw_sumx,,none,always,raw_sumx (channels),,true
define1 -1000,9000,4,raw_sumy,,none,always,raw_sumy (channels),,true
define1 -1000,9000,4,raw_sumw,,none,always,raw_sumw (channels),,true
define1 -2000,20000,4,raw_sumxyw,,none,always,raw_sumxyw (channels),,true
define1 -5000,5000,4,raw_diffxy,,none,always,raw_diffxy (channels),,true
define1 -6000,6000,4,raw_x,,none,always,raw_x (channels),,true
define1 -6000,6000,4,raw_y,,none,always,raw_y (channels),,true
define1 -6000,6000,4,raw_w,,none,always,raw_w (channels),,true
define2 -6000,6000,10,raw_x,x1-x2 raw,-6000,6000,10,raw_y,y1-y2 raw,none,always,X/Y (u/v) raw (channels),true
define2 -6000,6000,10,raw_x,x1-x2 raw,-6000,6000,10,raw_y,y1-y2 raw,none,clean_hit,X/Y (u/v) raw clean (channels)
;define2 -6000,6000,10,raw_x,u1-u2 raw,-6000,6000,10,raw_w,w1-w2 raw,none,always,u/w raw (channels),true
;define2 -6000,6000,10,raw_y,v1-v2 raw,-6000,6000,10,raw_w,w1-w2 raw,none,always,v/w raw (channels),true
```

These spectra give information on computed (raw) time sum and position coordinates.

```
;define1 -1000,1000,p20,x1,ch1 Time [ns],none,always,ch1 (ns),,true
;define1 -1000,1000,p20,x2,ch2 Time [ns],none,always,ch2 (ns),,true
;define1 -1000,1000,p20,y1,ch3 Time [ns],none,always,ch3 (ns),,true
;define1 -1000,1000,p20,y2,ch4 Time [ns],none,always,ch4 (ns),,true
;define1 -1000,1000,p20,z1,ch5 Time [ns],none,always,ch5 (ns),,true
;define1 -1000,1000,p20,z2,ch6 Time [ns],none,always,ch6 (ns),,true
```

If parameters 110-112 = 1 calibrated (ns) time spectra are displayed, as the next spectra for channel 7 and sums/differences.

```
define1 -1000,1000,p20,TOF,ch7 Time (TOF) [ns],none,always,ch7 (TOF) in ns,,true
define1 1,400,p20,sumx,sumx Time [ns],none,always,sumx (ns)
;define1 1,400,0.1,sumx,,none,always,sumx (ns)
define1 1,400,p20,sumy,sumy Time [ns],none,always,sumy (ns)
;define1 1,400,0.1,sumy,,none,always,sumy (ns)
define1 1,400,p20,sumw,sumw Time [ns],none,always,sumw (ns)
;define1 1,400,0.1,sumw,,none,always,sumw (ns)
;define1 1,900,p20,sumxyw,sumxyw Time [ns],none,always,sumxyw (ns)
;define1 -300,300,p20,diffxy,diffxy Time [ns],none,always,diffxy (ns),,true

define1 -100,100,p115,PosX,PosX [mm],none,always,PosX (mm),,true
ddefine1 -100,100,p116,PosY,PosY [mm],none,always,PosY (mm),,true

define2 -100,100,p113,PosX,PosX [mm],-100,100,p114,PosY,PosY [mm],none,always,PosX/PosY coarse (mm),true
define2 -100,100,p113,PosX,PosX [mm],-100,100,p114,PosY,PosY [mm],none,clean_hit,PosX/PosY coarse clean (mm)
;define2 -100,100,p115,PosX,PosX [mm],-100,100,p116,PosY,PosY [mm],none,always,PosX/PosY (mm),true
define2 -100,100,p115,PosX,PosX [mm],-100,100,p116,PosY,PosY [mm],none,clean_hit,PosX/PosY clean (mm)
;define2 -50,50,p117,PosX,PosX [mm],-50,50,p118,PosY,PosY [mm],none,clean_hit,PosX/PosY fine clean (mm)
```

If the parameters 1010 to 1012 are set properly for the delay-line in use the spectra with PosX and PosY as coordinates show position values in mm. Specific conditions remove incompletely registered events and produce “clean” images. The condition clean_hit may be replaced by a more refined condition (i.e. on the time sums).

```
definemulti Overview,PosX/PosY coarse clean (mm),sumx (ns),sumy (ns),ch7 (TOF) in ns
```

gives an overview of several spectra of interest in a multi-spectrum view

Control spectra as the following may be found useful for trouble-shooting

```
;define2 -100,100,p113,PosX,PosX [mm],-100,1000,1,sumx,sumy [ns],none,clean_hit,PosX/sumx (condsumy)
;define2 -100,100,p114,PosY,PosY [mm],-100,1000,1,sumy,sumx [ns],none,clean_hit,PosY/sumy (condsumx)
;define1 -200,200,1,reflection_in_MCP,dt [ns],none,always,reflection in MCP signal (ns)
;define1 -200,200,1,reflection_in_x1,dt [ns],none,always,delay-line reflection in x1 (ns)
;define1 -200,200,1,reflection_in_x2,dt [ns],none,always,delay-line reflection in x2 (ns)
;define1 -200,200,1,reflection_in_y1,dt [ns],none,always,delay-line reflection in y1 (ns)
;define1 -200,200,1,reflection_in_y2,dt [ns],none,always,delay-line reflection in y2 (ns)
;define1 -200,200,1,reflection_in_z1,dt [ns],none,always,delay-line reflection in z1 (ns)
;define1 -200,200,1,reflection_in_z2,dt [ns],none,always,delay-line reflection in z2 (ns)
```

Examples for empty spectra definition (for spectra computations or projections) are

```
;define1 -100,100,1,none,,none,always,Empty 1D
;define2 -100,100,1,none,, -100,100,1,none,,none,always,Empty 2D
```

Now the data acquisition is started:

```
start
show status
view 10
```

Note that the command definitions here shall only allow a “quick start” for the use of our delay-line detectors. More advanced data treatments like defining new (computed) coordinates to the analysis can be done by additionally modifying the DAn.dll using a Microsoft C++ compiler (Visual Studio 2010 or above). Also, there is a “zero-level” of operating for the recent **CoboldPC** 2011 version, allowing to address the **CoboldPC** commands by a “remote control” scripting language.(see section "The CoboldPCCMD Program" for further information).

The CoboldPCCMD Program

The CoboldPCCMD program is designed to give **CoboldPC** 2011 scripting or remote-control functionality.

CoboldPCCMD is a simple CMD (command line) program that enables the user to send commands to **CoboldPC** command line and executes the command in **CoboldPC**. Additionally the program gives some options to inquire the status of CoboldPC.

You can run CoboldPCCMD either in a CMD window or start is as a small GUI (Graphical User Interface) program from your explorer.

Running CoboldPCCMD in a CMD or calling it from your program

CoboldPCCMD.exe "command" or parameter

Possible parameters are:

- DAqState
- DAqMode
- CommandRunning
- Help

Return values for DAqState

- 1 ??? Unknown
- 1 Stopped
- 2 Paused
- 3 Running

Return values for DAqMode

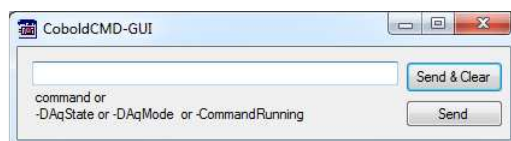
- 1 ??? Unknown
- 1 Unknown
- 2 Hardware
- 3 File

Return values for CommandRunning

- 1 ??? Unknown
- 2 Yes
- 3 No

Running CoboldPCCMD as a GUI program from explorer or calling it with no parameters/options

A small program will pop up giving you a command line like **CoboldPC** with two buttons aside (see picture)



The "Send & Clear" button will send the command and clears the GUI's command line.

The "Send" button will send the command and leaves the command line untouched.

All commands in a **CoboldPC** session have to be entered via the command line or the menu. To ease up the input via command line one can use predefined batchfiles (ext. ccf) with the exe command. If a started batchfile is finished a certain tone sequence will sound. All defined **CoboldPC** commands can be found in the section "Overview of commands".

It is not necessary to type the full command name. One needs only to type as many characters to make the beginning unique with respect to other **CoboldPC** commands. If too few characters are given and the command is ambiguous **CoboldPC** will execute the command which is first in alphabetical order.

Example: **shift** and **show**

Batchfiles can also be addressed without the exe command just by typing the filename (without extension) in the command file. Note that there can be a name conflict with predefined commands.

If a certain character combination is not found in the list of commands or as batchfile in the main folder a certain two-tone combination will sound.

Most commands require one or more command parameters, e.g. ViewSpectrum requires a spectrum number.

A row of command parameters is separated from the command by a space and from each other by comma. The order of the command parameters is defined for each command (see helpfile). Some parameters are always required with the command (as the spectrum number after ViewSpectrum), some are optional as an input of xlow and xhigh after ViewSpectrum n,... (n is the spectrum number here).

If a required parameter is not supplied a Windows menu box will open and require the command parameter input. This is not the case when optional parameters are omitted, then the program uses default values or values given before.

Sometimes a command requires the setting of markers on the displayed spectrum with the mouse. Before the marks are not set the command line is disabled.

Note: in rare cases after mouse operation the command line will be disabled although there is no obvious reason. In this case click on the command line and the blinking command marker indicates that the command line is enabled again.

With arrow up/arrow down you may scroll to the previous command list, you also find the last typed commands in a drop down list (right button next to the command line).

When you use a commands in a batch file that requires for example mouse operation or starts a subroutine you have to follow this command by the wait command. Then the execution of the batch-file is paused until the previous command operation is finished properly.

Examples:

Cursor

Marker

IntegrateSpectrum

File menu commands

The File menu offers the following commands:

Open	Opens an existing document.
Restart	Restarts with an empty Cobold document.
Load DAq dll Module...	Load/Exchange the DAq module dll.
Load DAn dll Module...	Load/Exchange the DAn module dll.
Execute	Execute a command file.
EndBatchJobs	End all running batch jobs.
Save	Saves an opened document using the same file name.
Save As	Saves an opened document to a specified file name.
Print	Prints a document.
Print Preview	Displays the document on the screen as it would appear printed.
Print Setup	Selects a printer and printer connection.
Preferences	Define global CoboldPC settings.
ShellCommand	Execute an external program.
Exit	Exits Cobold.

Open command (File menu)

Use this command to open an existing PC or ATARI document. If the filename extension is **.dmp** CoboldPC tries to load the old ATARI Dump file. In this case the dump file version must be 2.3 otherwise load fails.

For a CoboldPC document file the version number must be newer than 4.1 otherwise load fails.

You can create new documents with the [Restart command](#).

Shortcuts

Toolbar: 

Keys:CTRL+O

Restart command (File menu)

Use this command to restart with a new document in Cobold. Command can also be used in CoboldPC command line.

You can open an existing document with the [Open command](#).

Shortcuts

Toolbar: 

Keys:CTRL+R

Load DAq dll Module...(File menu)

Using this command you can load/exchange the actual DAq dll module of the running **CoboldPC**. This command can also be called by the "LoadDAqModule" command from the command line.

You can also change the DAq dll module from the About-Box.

The DAq module can not be exchanged if the **CoboldPC** is not in "stop" condition (yellow square in the status line).

Load DAn dll Module...(File menu)

Using this command you can load/exchange the actual DAn dll module of the running **CoboldPC**. This command can also be called by the "LoadDAnModule" command from the command line.

You can also change the DAn dll module from the About-Box.

The DAn module can not be exchanged if the **CoboldPC** is not in "stop" condition (yellow square in the status line).

ExecuteCommandFile

Execute a command-file. This file can contain a summary of often used commands. A command file can call another command file.

If execute fails CallMyFunction command is used with the given filename. If that one also fails, execute and callmyfunction are also executed using the macros folder in the application path (if available).

Parameters:

CommandFileName

CommandFileName The name of the command file including drive and path.

Shortcuts

Toolbar: 

EndBatchJobs

Terminates execution of all running batch jobs.

Can be called as command in a Cobold Command File (CCF), as menu command from menu FILE and from the ToolBar.

Shortcuts

Toolbar: 

Save command (File menu)

Use this command to save the active document to its current name and directory. When you save a document for the first time, Cobold displays the Save As dialog box so you can name your document. If you want to change the name and directory of an existing document before you save it, choose the [Save As command](#).

Shortcuts

Toolbar: 

Keys:CTRL+S

Save As command (File menu)

Use this command to save and name the active document. Cobold displays the Save As dialog box so you can name your document.

There are 3 different file formats one can use:

- dcf
- dc fz
- dcfd

dcf is the old Cobold Document File format.

dc fz is the internally compressed Cobold Document File format using zip algorithm

dcfd is the internally compressed Cobold Document File format using deflated algorithm

To save a document with its existing name and directory, use the , use the [Save command](#).

Print command (File menu)

Use this command to print a document. This command presents a Print dialog box, where you may specify the range of pages to be printed, the number of copies, the destination printer, and other printer setup options.

Shortcuts

Toolbar: 

Keys:CTRL+P

Print Preview command (File menu)

Use this command to display the active document as it would appear when printed. When you choose this command, the main window will be replaced with a print preview window in which one or two pages will be displayed in their printed format. The print preview toolbar offers you options to view either one or two pages at a time; move back and forth through the document; zoom in and out of pages; and initiate a print job.

Shortcuts

Toolbar: 

Print Setup command (File menu)

Use this command to select a printer and a printer connection. This command presents a Print Setup dialog box, where you specify the printer and its connection.

Preferences (File menu)

Use this command to define global Cobold settings.

- Load last document on startup
- Ignore Modified Flag
- Preferred Save File Type
- Reroute Cobold File Types
- Use DAq/DAn Wizard (not implemented yet)
- Clear EventData array before DAq execution
- Load Last DAn/DAq dll Modules
- Save DAq SourceCode with LMF
- Save DAn SourceCode with LMF
- Save CCF History with LMF
- Save DAn SourceCode with document
- [Colors](#)
- [ColorBar](#)
- Pprint Page Headers
- Show Splash Screen at Startup
- [Select Decimal Symbol and Delimiter](#)
- Sound Setup...
- Sound Support
- Projection Auto Mode Definition...

ShellCommand

Execute a Win32-Command program.

Parameters:

CommandLine

CommandLine Commandline to be executed.

Note:

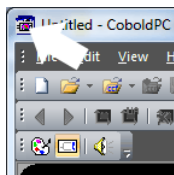
The command line is executed by cmd.exe with /c parameter.

Exit command (File menu)

Use this command to end your Cobold session. You can also use the Close command on the application Control menu. Cobold prompts you to save documents with unsaved changes.

Shortcuts

Mouse:Double-click the application's Control menu button.



Edit menu commands

The Edit menu offers the following commands:


CopyGraphics (print mode)	Copies graphic data from the document to the clipboard in print mode.
CopyGraphics (screen mode)	Copies graphic data from the document to the clipboard in screen mode.
CopyAscii	Copies ascii data from the document to the clipboard.
CopySpectrumBinary	Copies spectrum binary data from the document to the clipboard.
CopyIntegrationData	Copies integration data from the document to the clipboard.
PasteAscii	Paste ascii data from the clipboard to the displayed spectrum.
PasteAscii without clear	Paste ascii data from the clipboard to the displayed spectrum without clearing it before paste.
PasteSpectrumBinary (overwrite)	Paste spectrum binary data from the clipboard to the displayed spectrum overwriting the displayed spectrum.
PasteSpectrumBinary (append)	Paste spectrum binary data from the clipboard by appending it to the spepectrum list.
ExportAscii	Export ascii data from the displayed spectrum to a file.
ImportAscii	Import ascii data from a given file from the clipboard to the displayed spectrum.
ImportAscii without clear	Import ascii data from a given file from the clipboard to the displayed spectrum without clearing it.
ClearClipboard	Clears the clipboard.

CopyGraphics (print mode) command (Edit menu)

Use this command to copy the displayed graphic onto the clipboard as using a printer. This command is unavailable if no spectrum is currently displayed.

Copying data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar: 

Keys:CTRL+ALT+C

Parameters


none

CopyGraphics (screen mode) command (Edit menu)

Use this command to copy the displayed graphic onto the clipboard as using a screen copy. This command is unavailable if no spectrum is currently displayed.

Copying data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar: 

Keys:CTRL+SHIFT+ALT+C

Parameters


none

CopyAscii command (Edit menu)

Use this command to copy the spectrum data onto the clipboard. This command is unavailable if no spectrum is currently displayed.

Copying data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar: 

Keys:CTRL+ALT+A

Parameters

none

Copy Spectrum Binary (Edit menu)

Copies the displayed spectrum as a binary spectrum to the clipboard.

CopyIntegrationData command (Edit menu)

Use this command to copy the spectrum data onto the clipboard. This command is unavailable if no spectrum is currently displayed.

Copying data to the clipboard replaces the contents previously stored there.

Shortcuts

Toolbar: 

Keys:CTRL+I

Parameters

none

Paste Ascii command (Edit menu)

Use this command to copy the spectrum data from the clipboard to the displayed spectrum. The dataformat must be compatible with the defined spectrum. This command is unavailable if no spectrum is currently displayed.

This command is only available via the main menu or a pop-up menu (right mouse click).

Parameters

none

Dataformat:

x *separator* **y**

x *separator* **y** *separator* **z**

x *separator* **y** *separator* **spectrumnumber** *separator* **z**

separator can be a space, comma or tab.

Data-Lines not comparable to this format are ignored!

Paste ASCII without clear (Edit menu)

Paste ASCII data from the clipboard to the displayed spectrum without clearing the spectrum first. That means the clipboard data will be added to the displayed spectrum.

Paste Spectrum Binary (overwrite) (Edit menu)

Paste a binary spectrum from the clipboard to the displayed spectrum. This will replace only the data inside the displayed spectrum not the spectrum title nor the axis definitions.

Paste Spectrum Binary (append) (Edit menu)

Paste a binary spectrum from the clipboard by appending it to the spectrum list.

ExportAscii command (Edit menu)

Use this command to copy the spectrum data from the displayed spectrum to a file. This command is unavailable if no spectrum is currently displayed.

Parameters

Filename

Dataformat:

x *separator* **y**

x *separator* **y** *separator* **z**

separator is a space.

ImportAscii command (Edit menu)

Use this command to copy the spectrum data from a file to the displayed spectrum. The dataformat must be compatible with the defined spectrum. This command is unavailable if no spectrum is currently displayed.

Parameters

Filename

Dataformat:

x *separator* **y**

x *separator* **y** *separator* **z**

x *separator* **y** *separator* **spectrumnumber** *separator* **z**

separator can be a space, comma or tab.

Data-Lines not comparable to this format are ignored!


Import ASCII without clear ... (Edit menu)

Import an ASCII spectrum from file to the displayed spectrum without clearing the spectrum first. That means the clipboard data will be added to the displayed spectrum.

Clear Clipboard command (Edit menu)

Use this command to clear the clipboard. This command is unavailable if no data was stored to the clipboard by CoboldPC.

Shortcuts

Toolbar: 

Keys:

View menu commands

The View menu offers the following commands:

Tool Bars	Shows or hides the toolbar.
Status Bar	Shows or hides the status bar.
Command Bar	Shows or hides the command bar.
Delete Command History	Deletes the history of the command bar.
Zoom	Zoom the display.
Next Spectrum	Display the next spectrum.
Previous Spectrum	Display the previous spectrum.

Toolbar command (View menu)

Use this command to display and hide the Toolbar, which includes buttons for some of the most common commands in Cobold, such as File Open. A check mark appears next to the menu item when the Toolbar is displayed.

See [Toolbar](#) for help on using the toolbar.

Status Bar command (View menu)

Use this command to display and hide the Status Bar, which describes the action to be executed by the selected menu item, depressed toolbar button, and keyboard latch state. A check mark appears next to the menu item when the Status Bar is displayed.

See Status Bar for help on using the status bar.

Command Bar (View menu)

Use this command to toggle the view of the command bar.

Delete Command History (View menu)

Use this command to clear the command bar history.

Zoom

Select the view size of the display functions. (Till now not all modes are working fine!).

Possible values are, 200%, 150%, 100%, 75%, 50%, 25%, 10%, page ratio and fullpage. Except in fullpage the ratio of the two sides is sqrt(2) and the scaling is isotropic.

Next Spectrum (View menu)

Use this command to view the next available spectrum.

Previous Spectrum (View menu)

Use this command to view the previous available spectrum.

Help menu commands

The Help menu offers the following commands, which provide you assistance with this application:

Help Topics	Offers an index of topics in help file.
About	Displays the version number of this application.

Using Help command (Help menu)

Use this command for instructions about using Help.

About command (Help menu)

Use this command to display the copyright notice and version number of your copy of **CoboldPC**.

In here you can also see what DAq/DAn module is loaded or change it.

Overview of commands

General and most frequently used commands

exe	executes a command file script
new hardware	prepares for starting an acquisition
start	starts the acquisition
pause	pauses the acquisition (for starting again use the start command)
stop	stops the acquisition (for starting again use the new hardware and the start command)
clear all	clears the contents of all spectra (instead of all, a certain spectrum number is also possible to clear only one spectrum contents)
restart	deletes all coordinates, parameters, conditions and spectra for a total reset
show	status shows the status report window
help	shows the help file
wait	pauses the batch file until the previous command operation is finished

Note: Some commands, like Define1DimensionalSpectrum, may use the prefix *try*. With the try prefix (including the space character after try) the command performs as equal as without that prefix but gives no warning in case of an error. It simply ignores the error and continues with command execution.

Note: Before being processed all whitespaces (including the tab character) are eliminated before command execution.

Data acquisition handling commands

For data acquisition dll modules use the following commands:

LoadDAqModule	Load/Exchange the DAq module dll.
LoadDAnModule	Load/Exchange the DAn module dll.

For data acquisition handling use the following commands:

ForceLMFFlushAfterEvent	Force LMF Flush After every Event
NewAcquisition	Prepare Cobold for taking data.
PauseAcquisition	Pauses the data acquisition.
RewindListModeFile	Set the read file pointer to the first data in ListModeFile.
StartAcquisition	Starts (restarts) the data acquisition.
StopAcquisition	Stops data acquisition and closes ListModeFile.
Wait	Wait command to block inputs by mouse or keyboard.

For help setting up parameters just type the command without parameters and a dialog box will lead you through the specified command.

Load/Exchange DAq dll Module

Using this command you can load/exchange the actual DAq dll module of the running **CoboldPC**.

The DAq module can not be exchanged the **CoboldPC** is not in "stop" condition (yellow square in the status line).

Parameters:

DAqPathName

DAqPathName Path and file name of the DAq module

DAqPathName,ApplicationPath

DAqPathName Path and file name of the DAq module
ApplicationPath **APPLICATIONPATH to search the DAq module at the path where CoboldPC has been started.**

Load/Exchange DAn dll Module

Using this command you can load/exchange the actual DAn dll module of the running **CoboldPC**.

The DAn module can not be exchanged the **CoboldPC** is not in "stop" condition (yellow square in the status line).

Parameters:


DAnPathName

DAnPathName Path and file name of the DAn module

DAqPathName,ApplicationPath

DAnPathName Path and file name of the DAn module
ApplicationPath **APPLICATIONPATH to search the DAn module at the path where CoboldPC has been started.**

NewAcquisition

This command will prepare Cobold for taking data. The data source can be either hardware data acquisition or offline analysis of an existing ListModeFile.  will appear in the status line to indicate an successful execution.

The filename must not include a "," as part of the name but it may include drive and path information as well.

Parameters:

filename

filename is the name of the ListModeFile that will be analyzed. The filename should not have a "," as part of it's name!

fileread,filename,analysis,startat, events

fileread,lmfbasefile,analysis,startat, events., start, stop, ccfbasefile, saveFlag, clearFlag

fileread type **FILEREAD** to select the file read as data source for the acquisition.

filename Name of the ListModeFile that will be read during acquisition. If no name is given no ListModeFile will be written.

analysis Type **ANALYSIS** if you want to run through the user dependent part of the data analysis or **NOANALYSIS** if you don't.

events The number of events you want to acquire.

startat start event to read on. events from 0 to startat will be skipped

lmfbasefile multi file base filename for lmf writing

start multi file start number
stop multi file stop number
ccfbasefile multi file ccf base name (will be executed before LMF execution run)
saveFlag multifile saveFlag **YES** saves the document after every LMF execution run
clearFlag multifile clearFlag **YES** clears all spectra after every LMF execution run

hardware,filename
 hardware,filename,analysis,events,comment
 hardware,filename,analysis,events,comment,Start,Stop,ccffile,saveFlag,clearFlag

hardware type **HARDWARE** to select the hardware as data source for the acquisition.
filename Name of the ListModeFile that will be written during acquisition. If no name is given no ListModeFile will be written.
analysis Type **ANALYSIS** if you want to run through the user dependent part of the data analysis or **NOANALYSIS** if you don't.
events The number of events you want to acquire.
comment Comment lines written into the header of the ListModeFile for easier identification of the data in the file. For line separator use \n.
Start multi file start number
Stop multi file stop number (must be specified if **Start** is specified.
ccffile multi file ccf base name (will be executed before LMF execution run)
saveFlag multifile saveFlag **YES** saves the document after every LMF execution run
clearFlag multifile clearFlag **YES** clears all spectra after every LMF execution run

This command is only valid if an acquisition has not been started.

Note:

Multi-File-Mode:

In this mode CoboldPC will loop several DAq-cycles automatically. One DAq-Loop is finished if the selected number of events are reached. A number that ranges from **Start** to **Stop** will be appended to the original filename. The extension **.lmf** is appended automatically. If you specify **Yes** for the **Saveflag** then CoboldPC will also store the actual document under the same filename as the listmode-file except the extension is now **.dcf**.

whitespaces will be eliminated for readfile/hardware parameters as well as analysis/noanalysis

ForceLMFFlushAfterEvent


Using this command will force **CoboldPC** to flush its buffers after every event to make sure the data is transferred to the harddisk. This command should only be used at low event rates because it slows down the maximum event rate. At hi event rate the LMF buffer is store frequently to the harddisk.

Parameters:



Flag

Flag **enabled** to force flush, **disabled** to use normal LMF buffering

PauseAcquisition

This command will pause the data acquisition. The acquisition can be restarted by the [StartAcquisition](#) command. This command is only valid if an acquisition has been started. After this command in the status line the  will appear on the status line of CoboldPC

StartAcquisition

This command starts the data acquisition. If a ListModeFile was selected, the file will be opened and the header of the ListModeFile will be filled with special information (e.g. start time of the acquisition).  indicates a running offline dataanalysis (read from a listmode file) while  indicates a running online data acquisition (taking data from the hardware).

StopAcquisition

This command stops the data acquisition. If a ListModeFile was selected, the file is closed and the header of the ListModeFile is updated (stop time of the acquisition and number of events in this file). This command is only valid if an acquisition has been started. The status indicator will be emptied.

Parameters:

mode

mode **type multifilemode to stop multi file mode operation.**
Left blank will stop only the actual data acquisition.

Sleep

Sleep command freezes any command and message execution of CoboldPC for a given time in m (minutes), s (seconds) and ms (milliseconds). During this time even screen updates are not performed. This function can also not be terminated earlier like the WAIT function.

Parameters:

Time

Time Time dependent input for Sleep (see examples)

Time format can be milliseconds, seconds or minutes.

Examples:

Sleep 10s	Sleep for 10 seconds.
Sleep 500ms	Sleep for 500 milliseconds
Sleep 5m	Sleep for 5 minutes

Wait

Wait command to block inputs by mouse or keyboard.

Parameters:

Time

Counts

Time Time dependent input for Wait (see examples)



Time format can be seconds or

day-month-year:hours:minutes:seconds for absolute time or

+day:hours:minutes:seconds.

Counts Count dependent input for Wait (see examples)

Note:

Count input or no input for the wait command is only valid if DAq status is running ( or  indicator in status bar).

Examples:

Wait 10s	Block input for 10 seconds.
Wait 14-04-1999:8:15:00	Block input for till Clock reaches 14 th April 1998 8:15am
Wait +1:20:14:06	Block input for the next day 20hours 14minutes and 6seconds.
Wait	Blocks input until DAq read status is stopped (automatically at EOF).
Wait 100000	Blocks input until 100000 counts are processed (EventCounter == 100000) or EOF is reached.
Wait +100000	Blocks input until the next 100000 counts are processed or EOF is reached.

Definition and delete commands

For defining spectra, parameters... use these functions.

Condition	Define a condition.
Coordinate	Define coordinate(s).
CoordinateSet	Define a coordinateset.
Define1DimensionalSpectrum	Define a 1 dimensional spectrum.
Define2DimensionalSpectrum	Define a 2 dimensional spectrum.
DefineExperiment	Define experiment information
DefineMultiViewSpectrum	Define a multiple spectrum view (up to 4 spectra)
Delete	Delete function.
Grid	Toggle grid on and off.
Marker	Define a marker in a 1- or 2-dimensional spectrum.
Mode	Define lin- or log-modes for the axes.
Mode2D	Define the display type of a 2-dimensional spectrum.
Parameter	Define the value of a specified parameter.
SetAxisText	Define the axis text
SetPath	Define default paths for file io.
WeighParameter	Define the value of a specified weigh parameter.

For help setting up parameters, type the command without parameters and a dialog box will lead you through the specified command.

Condition

Defines or changes a condition.

Parameter:

Coordinate,MinimumValue,MaximumValue,NewConditionName

Condition1,Operation,Condition2,NewConditionName

NOperation,Condition,NewConditionName

<i>Coordinate</i>	the coordinate that will be checked.
<i>MinimumValue, MaximumValue</i>	the checked range.
<i>NewConditionName</i>	the name of the defined condition.
<i>Operation</i>	Can have the following values.
and	both of the conditions are true.
or	either condition is true.
xor	one condition is true, the other false. ⁴
nand	none of the conditions are true.

NOperation

not	the condition is false.
-----	-------------------------

Example:

```
condition coordinate1,100,350,condition1;  test condition1 between 100 and
                                           350
condition coordinate2,500,600,condition2;  test condition2 between 500 and
                                           600
condition
condition1,and,condition2,condition3;      require both condition1 and
condition not,condition1,condition4;      condition2 to be true
                                           not condition 1
```

Coordinate

This command will define one or more coordinates.

Parameters:

Coordinate1[,Coordinate2,Coordinate3...]

Examples:

```
coordinate x          define a coordinate with the name x
coordinate sum1,sum2  define coordinates with the names sum1 and sum2
```

CoordinateSet

This command will define a coordinateset. The given comment will be implemented into the coordinate definition.

Parameters:

SetName,Coordinate1[,Coordinate2,Coordinate3...]

Examples:

```
coordinateset set1,x,y  defines a coordinateset with the name set1. Set
                        included coordinates x and y
```

Define1DimensionalSpectrum

Defines a 1-dimensional spectrum. If the leading character of **Minimum**, **Maximum** or **Binsize** is a P or p then the following number will be interpreted as a parameter. The content of the parameter will be used for the specified definition.

Parameters:

Minimum,Maximum,Binsize,Coordinate,XaxisText,WeighParameter,Condition,Name,LogXFlag,LogYFlag
Minimum,Maximum,Binsize,CoordinateSet,XaxisText,WeighParameter,Condition,Name,LogXFlag,LogYFlag

<i>Minimum</i>	Lower boundary of the spectrum.
<i>Maximum</i>	Upper boundary of the spectrum.
<i>Binsize</i>	Binning of the spectrum.
<i>Coordinate (Set)</i>	Coordinate or Coordinate Set for this spectrum or none .
<i>XAxisText</i>	If not empty this text will be used for the X axis text.
<i>Weigh Parameter</i>	Index of the Weigh Parameter or none
<i>Condition</i>	Condition for this spectrum or always .

<i>Name</i>	Title of the spectrum.
<i>LogXFlag</i>	true to set logx mode in spectrum.
<i>LogYFlag</i>	true to set logy mode in spectrum.

Examples:

```
define1dimensionalspectrum 0,500,5,x,,none,csum,X-Projection
```

```
define1dimensionalspectrum 0,500,5,none,,none,always,X-Projection,,true
```

Note:

Due to the floating point problematic CoboldPC will accept values for coordinates even if they are $10e-9$ * Minimum below or $10e-9$ * Maximum above the range definition.

The channel acceptance (except definition borders) is from Channel-Binsize/2 to Channel+Binsize/2. The right value is not included.

If AxisDefault button is selected in the dialog box, then there is no field available to enter the axis text.

Using Coordinate Set instead of Coordinate allows to fill this spectrum in one event by every coordinate defined in the set.

Define2DimensionalSpectrum

Defines a 2-dimensional spectrum. If the leading character of **XMinimum**, **XMaximum**, **XBinsize**, **YMinimum**, **YMaximum** and **YBinsize** is a P or p then the following number will be interpreted as a parameter. The content of the parameter will be used for the specified definition.

Parameters:

XMinimum,XMaximum,XBinsize,XCoordinate,YMinimum,YMaximum,YBinsize,YCoordinate,WeighParameter,Condition,Name,Mode2d,LogZFlag,XAxisText,YAxisText
XMinimum,XMaximum,XBinsize,XCoordinateSet,YMinimum,YMaximum,YBinsize,YCoordinateSet,WeighParameter,Condition,Name,Mode2d,LogZFlag,XAxisText,YAxisText
XMinimum,XMaximum,XBinsize,XCoordinate,YMinimum,YMaximum,YBinsize,YCoordinateSet,WeighParameter,Condition,Name,Mode2d,LogZFlag,XAxisText,YAxisText
XMinimum,XMaximum,XBinsize,XCoordinateSet,YMinimum,YMaximum,YBinsize,YCoordinate,WeighParameter,Condition,Name,Mode2d,LogZFlag,XAxisText,YAxisText

<i>XMinimum</i>	Lower x-boundary of the spectrum.
<i>XMaximum</i>	Upper x-boundary of the spectrum.
<i>XBinsize</i>	Binning for the x-axis.
<i>XCoordinate (Set)</i>	Coordinate or Coordinate Set for x-axis or none .
<i>XAxisText</i>	If not empty this text will be used for the X axis text.
<i>YMinimum</i>	Lower y-boundary of the spectrum.
<i>YMaximum</i>	Upper y-boundary of the spectrum.
<i>YBinsize</i>	Binning for y-axis.
<i>YCoordinate (Set)</i>	Coordinate or Coordinate Set for y-axis or none .
<i>YaxisText</i>	If not empty this text will be used for the Y axis text.
<i>Weigh Parameter</i>	Index of the Weigh Parameter or none
<i>Condition</i>	Condition for this spectrum or always .
<i>Name</i>	Title of this spectrum.
<i>LogZFlag</i>	true to set logz mode in spectrum.
<i>Mode2d</i>	Define viewing mode of the displayed spectrum. (see Mode2d)
	valid values are, color , gray , grey , scatter , density , contourcolor , contourgray , contourgrey .

Examples:

```
define2dimensionalspectrum 0,500,5,x,,200,700,8,y,,none,csum,Position Picture,false,gray
```

```
define2dimensionalspectrum 0,500,5,x,,200,700,8,y,,none,always,Dummy 2D Spectrum,,gray
```

Note:

Due to the floating point problematic CoboldPC will accept values for coordinates even if they are $10e-9$ * Minimum below or $10e-9$ * Maximum above the range definition.

The channel acceptance (except definition borders) is from Channel-Binsize/2 to Channel+Binsize/2. The right value is not included.

If AxisDefault button is selected in the dialog box, then there is no field available to enter the axis text.

Using Coordinate Set instead of Coordinate allows to fill this spectrum in one event by every coordinate defined in the set.

DefineExperiment

Define the relevant information about the experiment and this document.

DefineMultiDimensionalSpectrum

Defines a multi-dimensional spectrum.

Parameters:

Name,sp1[,sp2,sp3,sp4]

<i>Name</i>	Title of the spectrum.
<i>Sp1</i>	Number or Name of upper left spectrum to be displayed
<i>Sp2</i>	Number or Name of upper right spectrum to be displayed
<i>Sp3</i>	Number or Name of lower left spectrum to be displayed
<i>Sp4</i>	Number or Name of lower right spectrum to be displayed

Examples:

definemultidimensionalspectrum MultiView,1,2
definemultidimensionalspectrum MultiView,1,2,,4
definemultidimensionalspectrum MultiView,1,2,,4

Delete

Function to delete parts of a CoboldPC document.

Parameter:

Type,Number1[,Number2,Number3...]

<i>Type</i>	Spectrum	Delete a spectrum from the document
<i>Number#</i>		Spectrum number.

Note:

For Number(s) like 5,6,7,8 you may write 5-8

Grid

Turn grid on or off in a spectrum.

Parameters:

Flag

<i>Flag</i>	On or Off	Specify on to turn grid on Specify off to turn grid off
-------------	-------------------------------	--

Note:

For dialog box support leave parameters empty.

Marker

Defines and displays a dashed marker in a 1- or 2-dimensional spectrum.

While the crosshair cursor is displayed the mouse cursor is disabled.

Parameters: (1-Dimension Spectrum)

Pos1[,Pos2,...,PosN]

Parameters: (2-Dimension Spectrum, orthogonal mode)

[Mode = o]

xPos1

yPos1

.

.

xPosN

yPosN

Parameters: (2-Dimension Spectrum, r-phi mode)

Mode = r

xPosCenter1

yPosCenter1

r1

phi1

.

.

xPosCenterN

yPosCenterN

rN

phiN

Parameters: (2-Dimension Spectrum, box mode)

[Mode = b]

xPos1 1st corner

yPos1 1st corner

xPos1 2nd corner

yPos1 2nd corner.

.

xPosN 1st corner

yPosN 1st corner

xPosN 2nd corner

yPosN 2nd corner

<i>Mode</i>	Blank or O for orthogonal cursor, r for r-phi cursor (2D only), b for box cursor (2D only)
<i>xPos</i>	Position of x-axis cursor
<i>yPos</i>	Position of y-axis cursor
<i>xPosCenter</i>	Position of x-axis center cursor
<i>yPosCenter</i>	Position of y-axis center cursor
<i>r</i>	radius
<i>phi</i>	angle
xPos 1st corner	Position of x-axis cursor of the first corner
yPos 1st corner	Position of y-axis cursor of the first corner
xPos 2nd corner	Position of x-axis cursor of the second corner
yPos 2nd corner	Position of y-axis cursor of the second corner

Note:

Markers can be removed via the [remove](#) command

Box Markers will draw the box in solid line and outside the box dotted lines

Mode

Defines the linear or logarithm mode for the axis of the displayed spectrum. For a 1-dimensional spectrum, the x- and/or the y-axis can be set to linear or logarithm mode. For a 2-dimensional spectrum, only the z-axis can be set to linear or logarithm mode.

Parameter:

Axismode

<i>Axismode</i>	lin or log for a 2-dimensional spectrum. linlin both axis in linear mode. linlog x-axis linear-, y-axis log-mode. loglin x-axis log-, y-axis linear-mode loglog both axis in log-mode.
-----------------	--

Note:

if not specified other in the spectrum the minimum value display on an axis is 0.7 if log is selected for that axis

Mode2D

Define viewing mode of the displayed spectrum. This function is only available for 2-dimensional spectra. For a contour plot, the smoothing function will not affect the original data.

Parameter:

Viewmode

Contourmode[,# of lines,# of iterations]

<i>Viewmode</i>	Color	color-scale plot.
	Gray	gray-scale plot.
	Grey	grey-scale plot.
	Scatter	scatter plot.
	Density	density plot.
<i>Contourmode</i>	Contourcolor	color-scaled contour plot.
	Contourgray	gray-scaled contour plot.
	Contourgrey	gray-scaled contour plot.
<i># of lines</i>		number of lines in the contour plot. (default is 10)
<i># of iterations</i>		number of iterations of smoothing before displaying the contour plot. (default is 10)

Parameter

Defining or modifying a parameter. The given comment will be implemented into the parameter definition.

Parameters:

Number,Value

<i>Number</i>	The number of the parameter.
<i>Value</i>	The content of the specified parameter.

Note:

To define a hexadecimal number use **0x** before the number or **h** after the number.

SetAxisText

Define the axis text. This text will replace the coordinate text. To remove selected text leave entry for axis text empty.

Parameter:

Spectrum,XText,YText

<i>Spectrum</i>	Define spectrum number.
<i>Xtext</i>	Text to replace x coordinate.
<i>Ytext</i>	Text to replace y coordinate.

SetPath

Define the initial default path for Cobold Documents, Command Files, List Mode Files, Import- and Export Files

Parameter:

Path

<i>Path</i>	New initial path for File IO. APPLICATIONPATH will set the path to the directory where CoboldPC is located, WORKPATH sets the default working directory. Width1Width3Width1526Width3Width6996
-------------	---

WeighParameter

Defining or modifying a weigh parameter. The given comment will be implemented into the weigh parameter definition.

Parameters:

Number,Value

<i>Number</i>	The number of the weigh parameter.
<i>Value</i>	The content of the specified parameter.

Note:

To define a hexadecimal number use **0x** before the number or **h** after the number.

Display commands

For displaying spectra or information about parameters, conditions, coordinates or spectra, use these functions.

CopyASCII	Copy displayed spectrum channel values to clipboard.
CopyGraphics	Copy displayed spectrum graphic to clipboard.
CopyIntegrationData	Copy displayed spectrum integration information to clipboard.
Cursor	Display a cursor marker without storing data.
ExportASCII	Export displayed spectrum channel values to a file.
ImportASCII	Import channel values from a file to the displayed spectrum.
NextSpectrum	Display the next available spectrum.
NoDisplay	Removes any spectrum from the display area.
OverlaySpectrum	Overlay a spectrum to the displayed one.
PasteASCII	Copy channel values in ASCII format from the clipboard to the displayed spectrum.
PreviousSpectrum	Display the previous available spectrum.
Show	Show the specified lists.
UpdateSpectrum	Update the display of the spectrum being viewed.
ViewSpectrum	View the specified spectrum.

For help setting up parameters, type the command without parameters and a dialog box will lead you through the specified command.

CopyASCII

Copy displayed spectrum channel values to the clipboard.

Note:

Delimiter and Decimal Point Symbol must fit.

See **File\Preferences\Select Decimal Symbol and Delimiter** menu.

CopyGraphics

Copy displayed spectrum graphic to the clipboard.

Note:

Delimiter and Decimal Point Symbol must fit.

See **File\Preferences\Select Decimal Symbol and Delimiter** menu.

CopyIntegrationData

Copy displayed spectrum integration information to the clipboard.

Note:

Delimiter and Decimal Point Symbol must fit.

See **File\Preferences\Select Decimal Symbol and Delimiter** menu.

Cursor

Display a cursor marker. On exiting cursor mode the last cursor values are not stored into the spectrum.

While the crosshair cursor is displayed the mouse cursor is disabled.

Parameters:

mode

Mode Blank or O for orthogonal cursor, r for r-phi cursor (2D only)

ExportASCII

Export displayed spectrum channel values to a file.

Parameters:

filename

filename is the name of export-file.

Note:

Delimiter and Decimal Point Symbol must fit.

See **File\Preferences\Select Decimal Symbol and Delimiter** menu.

ImportASCII

Import channel values from a file to the displayed spectrum.

Parameters:

filename[, ClearFlag]

filename is the name of import-file.

ClearFlag Type **NoClear** if you want not to initially clear the spectrum.

Note:

If ClearFlag is not given the spectrum will always be initially cleared.

Delimiter and Decimal Point Symbold must fit.

See **File\Preferences\Select Decimal Symbol and Delimiter** menu.

NextSpectrum

View the next available spectrum.

NoDisplay

This function removes any spectrum from the display area and prevents its redrawing.

OverlaySpectrum

Overlays a specified spectrum with a specified color.

Parameters:

Spectrum,Color,[Spectrum,Color,...]

Spectrum Spectrum number to be overlayed

Color R G B value of the color.

Note:

Samples for R G B values:

0xff 0xff 0xff

255 0 0xc0

PasteASCII

Copy channel values in ASCII format from the clopboard to the displayed spectrum.

Parameters:

ClearFlag

ClearFlag Type **NoClear** if you want not to initially clear the spectrum.

Note:

If ClearFlag is not given the spectrum will always be initially cleared.

Delimiter and Decimal Point Symbold must fit.

See **File\Preferences\Select Decimal Symbol and Delimiter** menu.

PreviousSpectrum

Display the previous available spectrum.

Show

Display the specified list or status. For show status the display is updated every second.

Parameters:

Type[,Text]

Type Type one of these words to specify the list. **Coordinates, Conditions, IntegrationData, ListmodeFileHistory, Marker, Message, Overlay, Parameters, WeighParameters, Spectra, Status, Rate** (Event Rate).

Text Text to be displayed in a message box if Type is **Message**. A newline command can be given by the \n charcter sequence.

Note:

In the **IntegrationData** or **Marker** list you can copy the list to the clipboard.

In the Spectra List you can activate the selected spectrum by pressing the Go To button or by double clicking the line of the spectrum that should be displayed.

In the **Spectra** list you can display the selected spectrum by clicking the **Go To** button or double clicking the spectrum line. Here the **Go To** button is also the default button

All literal sorts are caseless!

UpdateSpectrum

Updates the display of the spectrum being viewed or rescales all spectra at once.

Parameters:

Time s

Counts c

DeleteLoops d

<i>Time s</i>	Update time in seconds
<i>Counts c</i>	Update time in "counts"
<i>DeleteLoops d</i>	Delete update after "d" display updates
<i>All</i>	rescales all defined spectra

Note:

If the parameters for this function can not be determined the update time will be set to 30s.

Example:

Update 10s	update displayed spectrum every 10 seconds
Update 10000c	update displayed spectrum every 10000 counts.
Update 1s,5d	Update displayed spectrum every second and delete displayed spectrum content after each 5 th update.
Update	Update displayed spectrum only this time and rescale last free coordinate.
Update all	rescales all defined spectra

ViewSpectrum

Display the specified spectrum or display the boundaries of conditions defined by the coordinates of the displayed spectrum.

Parameters:

Spectrum[,XMinimum,XMaximum,YMinimum,YMaximum,ZMinimum,ZMaximum]

Spectrum[,x,y,z,r]

Condition

<i>Spectrum</i>	Spectrum number or title to be displayed.
<i>XMinimum,XMaximum</i>	Boundaries for x-axis.
<i>YMinimum,YMaximum</i>	Boundaries for y-axis.
<i>ZMinimum,ZMaximum</i>	Boundaries for z-axis.
<i>x,y,z</i>	Indicates the rescaling of the selected axes.
<i>r</i>	Indicates recaling of all possible axes.

Note:

To change the appearance of the spectrum, use the **Mode** or **Mode2d** function.

Rescaling any axis will automatically rescale the last axis of the spectrum (for a 1-dimensional spectrum the y-axis and for a 2-dimensional spectrum the z-axis).

If the value of minimum axis is not specified for log scaling then the minimum is set to 0.7

Example:

View 15	Display spectrum 15
View Condition	Display condition defined by the spectrums coordinate(s).

Mathematical commands

For mathematical operations, use these functions.

AddConstant	Add a constant value to a spectrum.
AddSpectrum	Add two spectra.
BinomialSmooth	Perform a binomial smooth.
ContourSmooth	Smooth a contour plot.
DivideConstant	Divide spectrum by a constant value.
DivideFunctionQ	Multiply all channels by $X^{(1/Q)}$.
DivideSpectrum	Divide two spectra.
FitSpectrum	Fitting Spectrum Data
GaussSmooth	Perform a smooth using a gaussian distribution.
GP1Correct	Correct the differential nonlinearity of the GP1-TDC
IntegrateSpectrum	Integrates a defined area in the displayed spectrum.
MeanSmooth	Perform a mean smooth.
MultiplyConstant	Multiply a spectrum by a constant value.
MultiplyFunctionQ	Multiply all channels by x^Q .
MultiplySpectrum	Multiply two spectra.
ProjectSpectrum	Makes a projection.
RotateSpectrum	Rotate a two dimensional spectrum.
SubtractConstant	Subtract a constant value from a spectrum.
SubtractFit	Subtract the last fit from the given spectrum.
SubtractSpectrum	Subtract two spectra.

For help setting up parameters, type the command without parameters and a dialog box will lead you through the specified command.

AddConstant

Adds a constant value to all channels in a spectrum.

Parameters:

Spectrum1, Value, Spectrum2

<i>Spectrum1</i>	Source spectrum.
<i>Value</i>	constant value (floating point value).
<i>Spectrum2</i>	Destination spectrum.

Note:

$Spectrum2 = Spectrum1 + Value$

AddSpectrum

Adds the contents of two spectra, stores result in third. The two spectra must have the same dimensional definition but need not have the same boundaries. If boundaries are not equal, only the overlapping region will be operated on.

Parameters:

Spectrum1, Spectrum2, Spectrum3

<i>Spectrum1</i>	Source spectrum 1.
<i>Spectrum2</i>	Source spectrum 2.
<i>Spectrum3</i>	Destination spectrum.

Note:

$Spectrum3 = Spectrum1 + Spectrum2$

BinomialSmooth

Smooth spectrum using a binomial distribution.

Parameters:

Spectrum1, N, Spectrum2

N

<i>Spectrum1</i>	Source spectrum.
<i>N</i>	Number of bins for smooth. (uneven integer number ≥ 3)
<i>Spectrum2</i>	Destination spectrum.

Note:

If only N is presented, Spectrum1 and Spectrum2 is set to the displayed spectrum. If no spectrum is displayed and only N is presented then the function call will do nothing.

ContourSmooth

Smooth 2-dimensional spectrum using a contour plot smoothing function (only for 2-dimensional spectra).

Parameters:

Spectrum1,N,Spectrum2

N

Spectrum1 Source spectrum.
N Number of iterations.
Spectrum2 Destination spectrum.

Note:

If only N is presented, Spectrum1 and Spectrum2 is set to the displayed spectrum. If no spectrum is displayed and only N is presented then the function call will do nothing.

DivideConstant

Divide all channels in a spectrum by a constant value.

Parameters:

Spectrum1,Value,Spectrum2

Spectrum1 Source spectrum.
Value constant value (floating point value).
Spectrum2 Destination spectrum.

Note:

$$\text{Spectrum2} = \text{Spectrum1} / \text{Value}$$
DivideFunctionQ

Multiplies all channels by $x^{(1/Q)}$. X is the channelnumber. (only available for 1-dimensional spectra).

Parameters:

Spectrum1,Spectrum2,Q

Spectrum1 Source spectrum.
Spectrum2 Destination spectrum.
Q Q-Value (floating point value).

Note:

$$\text{Spectrum2} = \text{Spectrum1} * x^{(1/Q)}$$
DivideSpectrum

Divides the contents of two spectra, stores result in third. The two spectra must have the same dimensional definition but need not have the same boundaries. If boundaries are not equal, only the overlapping region will be operated on.

Parameters:

Spectrum1,Spectrum2,Spectrum3

Spectrum1 Source spectrum 1.
Spectrum2 Source spectrum 2.
Spectrum3 Destination spectrum.

Note:

$$\text{Spectrum3} = \text{Spectrum1} / \text{Spectrum2}$$
FitSpectrum

Fitting spectrum data with the given function. The boundaries are inclusive.

Parameters:

Type,XMinimum,XMaximum

Type Fit function type.

Linear Regression for these function types
LINEARREGRESSION,
LOGARITHMREGRESSION,
POWERREGRESSION,
EXPONENTIALREGRESSION,

	GAUSS, POLYNOM
<i>XMinimum,XMaximum</i>	Boundaries for x-axis
<i>Order</i>	Order of the polynom (2<=order<=9)

Note:

For cursor input of the boundaries do not enter data for the boundaries fields.

For gauss fits the peak should lie inside the boundaries to get accurate and correct fit results.

GaussSmooth

Smooth a spectrum using a gauss distribution.

Parameters:

Spectrum1,Sigma,Spectrum2

Sigma

<i>Spectrum1</i>	Source spectrum.
<i>Sigma</i>	Sigma value in the gaussian formula. (FWHM??)
<i>Spectrum2</i>	Destination spectrum.

Note:

If only Sigma is presented, Spectrum1 and Spectrum2 is set to the displayed spectrum. If no spectrum is displayed and only Sigma is presented then the function call will do nothing.

GPICorrect

The function corrects the differential nonlinearity of the GP1-TDC by multiplying the counts of every second channel by a given number. After multiplication is done the spectrum is normalized so the total counts of the spectrum will be the same as before this operation.

Parameters:

Spectrum[,CorrX[,CorrY]]

<i>Spectrum</i>	Spectrum to correct.
<i>CorrX</i>	Correction value for x axis
<i>CorrY</i>	Correction value for y axis (in 2 dimension spectrum only)

Note:

To automatically adjust the values vor Corrx and Corry do not specify these values.

IntegrateSpectrum

Integrates a defined area in a one- or two-dimensional spectrum. The integration data is stored temporary in the spectrum but it is not saved with the dump file. A new integration will override the previous integration data if it is present. Each spectrum can hold one integration data! To remove the information use the [remove](#) function. The boundaries are inclusive.

Parameters:

BFlag

XMinimum,XMaximum[,BFlag]

XMinimum,XMaximum,YMinimum,YMaximum[,BFlag]

<i>XMinimum,XMaximum</i>	Boundaries for x-axis
<i>YMinimum,YMaximum</i>	Boundaries for y-axis in a two-dimensional spectrum
<i>BFlag</i>	TRUE includes all boundaries for integration
	FALSE will include only left (and in 2-dim spectrum lower) boundaries into integration

Note:

For cursor input of the boundaries do not enter data for the boundaries fields.

If no Bflag is given it is set automatically to **TRUE**.

MeanSmooth

Smooth a spectrum using a mean value smoothing function.

Parameters:

Spectrum1,N,Spectrum2

N

Spectrum1 Source spectrum.
N Number of bins for smooth. (uneven integer number >= 3)
Spectrum2 Destination spectrum.

Note:

If only N is presented, Spectrum1 and Spectrum2 is set to the displayed spectrum. If no spectrum is displayed and only N is presented then the function call will do nothing.

MultiplyConstant

Multiplies all channels by a constant value.

Parameters:

Spectrum1,Value,Spectrum2

Spectrum1 Source spectrum.
Value constant value (floating point value).
Spectrum2 Destination spectrum.

Note:

$\text{Spectrum2} = \text{Spectrum1} * \text{Value}$

MultiplyFunctionQ

Multiplies all channels by x^Q . X is the channelnumber. (only available for 1-dimensional spectra).

Parameters:

Spectrum1,Spectrum2,Q

Spectrum1 Source spectrum.
Spectrum2 Destination spectrum.
Q Q-Value (floating point value).

Note:

$\text{Spectrum2} = \text{Spectrum1} * x^Q$

MultiplySpectrum

Multiplies the contents of two spectra, stores result in third. The two spectra have to have the same dimensional definition but need not to have the same boundaries. If boundaries are not equal, only the overlapping region will be operated on.

Parameters:

Spectrum1,Spectrum2,Spectrum3

Spectrum1 Source spectrum 1.
Spectrum2 Source spectrum 2.
Spectrum3 Destination spectrum.

Note:

$\text{Spectrum3} = \text{Spectrum1} * \text{Spectrum2}$

ProjectSpectrum

The function projects a two-dimensional spectrum on x or y axis in a given range to a one-dimensional spectrum. The boundaries are inclusive.

Parameters:

Spectrum1,Spectrum2,Axis,From,To

Spectrum1 Source spectrum.
Spectrum2 Destination spectrum

Axis **auto** to let CoboldPC automatically define the spectrum
From **x** or **y** to specify the axis to project to.
To lower boundary.
 upper boundary.

Note:

For cursor input of the boundaries do not enter data for the boundaries fields.

For Dialog-Box input do not enter parameters for the function.

RotateSpectrum

The function rotates a two dimensional spectrum by a given angle. The origin of rotation can be specified.

Parameters:

From,Angle,XOrigin,YOrigin,To

<i>From</i>	Source spectrum.
<i>Angle</i>	Angle of rotation.
<i>XOrigin</i>	X value of origin of the rotation point.
<i>YOrigin</i>	Y value of origin of the rotation point.
<i>To</i>	Destination spectrum.

SubtractConstant

Subtract a constant value from all channels in a spectrum.

Parameters:

Spectrum1,Value,Spectrum2

<i>Spectrum1</i>	Source spectrum.
<i>Value</i>	constant value (floating point value).
<i>Spectrum2</i>	Destination spectrum.

Note:

$Spectrum2 = Spectrum1 - Value$

SubtractFit

Subtract the last fit from the given spectrum.

Parameters:

[From, To]

<i>From</i>	Source spectrum.
<i>To</i>	Destination spectrum.

Note:

If no **From** and **To** is given this command will use the displayed spectrum for **From** and **To**.

If **To** is not given, **From** will be set to **To** value.

If fit is of type GaussFit then only the peak information will be subtracted. The y-offset of the GaussPeak will be ignored

SubtractSpectrum

Subtract the contents of two spectra, store results in third spectrum. The two spectra must have the same dimensional definition but need not have the same boundaries. If boundaries are not equal, only the overlapping region will be operated on.

Parameters:

Spectrum1,Spectrum2,Spectrum3

<i>Spectrum1</i>	Source spectrum 1.
<i>Spectrum2</i>	Source spectrum 2.
<i>Spectrum3</i>	Destination spectrum.

Note:

$Spectrum3 = Spectrum1 - Spectrum2$

Spectrum handling commands

For special spectrum manipulations, use these commands.

CalibrateSpectrum	Calibrate a spectrum.
ClearSpectrum	Clear one or more spectra.
CopySpectrum	Copy a spectrum.
CutOffNegativeValues	Set all negative values in spectrum to zero.
ExpandSpectrum	Expand a spectrum.

Remove	Remove additional data from a spectrum
SetChannelToValue	Set a channel in a spectrum to a specific value.
SetSpectrumTitle	Set the title of a spectrum.
ShiftSpectrum	Shift the spectrum.
ZeroSpectrum	Set a region in a spectrum to zero.

For help setting up parameters, type the command without parameters and a dialog box will lead you through the specified command.

CalibrateSpectrum

This function will calibrate the specified spectrum.

Parameters:

Type,XMinimum,NewXMinimum,XMaximum,NewXMaximum,NewXAxisText[,YMinimum,NewYMinimum,YMaximum,NewYMaximum,NewYAxisText]

<i>Type</i>	LIN performs a linear calibration.
<i>XMinimum</i>	old x-minimum value
<i>NewXMinimum</i>	new x-minimum value
<i>XMaximum</i>	old x-maximum value
<i>NewXMaximum</i>	new x-maximum value
<i>NewXAxisText</i>	new x-axis text
<i>YMinimum</i>	old y-minimum value
<i>NewYMinimum</i>	new y-minimum value
<i>YMaximum</i>	old y-maximum value
<i>NewYMaximum</i>	new y-maximum value
<i>newYAxisText</i>	new y-axis text

Note:

If no boundaries are entered a cursor-dialog based procedure will be used for boundary input.

ClearSpectrum

This function will clear the specified spectra.

Parameters:

Spectrum1[,Spectrum2,Spectrum3,...]

Word

<i>Spectrum1..n</i>		Spectrum number(s) to be cleared.
<i>Word</i>	all	will clear all spectra.

CopySpectrum

Copies one spectrum to another one. The source and the destination spectra must have the same dimension. If the spectrum borders differ, only the overlapping area is copied. The bin-sizes need not to be the same.

Parameters:

Spectrum1,Spectrum2

<i>Spectrum1</i>	The source spectrum.
<i>Spectrum2</i>	The destination spectrum.

CutOffNegativeValues

This command sets all negative values of the displayed spectrum to zero.

Parameters:

Spectrum

<i>Spectrum</i>	The spectrum number.
-----------------	----------------------

Note:

Function will use the displayed spectrum if no spectrum number is given.

ExpandSpectrum

Expands either a 1- or a 2-dimensional spectrum using a cursor to determine the expansion area. For the second cursor input the delta area is also displayed.

Remove

Remove special data from a spectrum.

Parameter:**FIT**

FIT,Spectrum-Number(s)

MARKER,ALL

MARKER,InSpectrum,Marker-Number(s)

MARKER,InSpectrum,ALL

CALIBRATION,Spectrum-Number(s)

INTEGRATIONDATA

INTEGRATIONDATA, Spectrum-Number(s)

OVERLAY,ALL

OVERLAY,InSpectrum,ALL

OVERLAY,InSpectrum,Spectrum-Number(s)

CONDITION

CONDITION,Spectrum-Number(s)

UPDATE*InSpectrum* Function work in this spectrum.*Spectrum-Number(s)* spectrum number(s).*Marker-Number(s)* marker-number(s) that should be removed. **ALL** will remove all marker information of the selected spectrum. If no spectrum is selected the action is performed on the displayed spectrum.**Note:**For **IntegrationData** the integration list is also updated!For reactivate display of **CONDITION** information use **SHOW** command.

If no inspectrum is defined then the function will operate the displayed spectrum.

For Number(s) like 5,6,7,8 you may write 5-8

SetChannelToValue

Sets the specified channel to a given value.

Parameters:

Spectrum,XChannel,Value

Spectrum,XChannel,YChannel,Value

Spectrum Spectrum number.*XChannel* Channel of the x-axis.*YChannel* Channel of the y-axis (only for 2-dimensional spectra).*Value* New value.**SetSpectrumTitle**

This command shifts the specified spectrum by an offset value(s).

The shift is performed by redefining the spectrum definition, so no data will be lost by the shift.

Parameters:

Spectrum,NewTitle

Spectrum Spectrum number.*NewTitle* New title of the spectrum.**ShiftSpectrum**

This command shifts the specified spectrum by an offset value(s).

The shift is performed by redefining the spectrum definition, so no data will be lost by the shift.

Shift can also be performed using the keyboard.

Parameters:

Spectrum,XOffset[,YOffset]

Spectrum Spectrum number.*XOffset* Offset for the x-axis.*YOffset* Offset for the y-axis (only for 2-dimensional spectra).

ALT+Cursor shift spectrum by 80% in the given direction

CTRL+ALT+Cursor shift spectrum by 10% in the given direction

ZeroSpectrum

Zeros a specified spectrum in the given area.

Parameters:

Spectrum,XMinimum,XMaximum[, YMinimum,YMaximum]

Spectrum Spectrum number.
XMinimum,XMaximum Area in x-direction that will be zeroed.
YMinimum,YMaximum Area in y-direction that will be zeroed (only for 2-dimensional spectra).

User Calls

User Calls offers a method to communicate directly with the CoboldPC program. This allows the call of specific CoboldPC commands from "inside" CoboldPC called by a users Dll.

[CallMyFunction](#) The function allows the user to call a special function call (**DoFunction**) an a user written Dll.

CallMyFunction

The function allows the user to call a special function call (**DoFunction**) an a user written Dll.

A sample solution is included in the CoboldPC distribution. The solution contains all that is needed for such a project. The Code contains a sample source that is used in the **Start.ccf**.

Parameters:

DllPathName[,Input1[,Input2,[...]]]

DllPathName Name and path of the Dll to call. The Dll must provide an exported function

```
void DoFunction(CString csInput[],
               unsigned __int32 ui32InputLength,
               unsigned __int32 ui32WordCount);
csInput            Input Strings as an array of CString
                  csInput[0] is the call command
                  csInput[1] is the DllPathName
                  csInput[2] is Input1
                  csInput[3] is Input2
                  ...
ui32InputLength   dimension of the array
ui32WordCount    size of the input array
```

Input1 input string 1
Input2 input string 2
 ... more input strings

Sample from Start.ccf

Call UserFCall,SetDAQCoordinates,T1Ch??n,T1Ch??S??

Note:

No input string is needed

DllPathName can be skippt by colon.

Introduction

For most applications the standard DAn.dll subroutine provided with **CoboldPC** is sufficient for most data acquisition and analysis needs.

However for special and more flexible DAn.dll it is possible for the user to adjust the DAn.dll according to his needs by compiling a new DAn.dll.

The main aim of changing the code usually is to define additional coordinates that are computed from other coordinates (for example to make an R/Phi representation of an acquired detector image). Make sure that you adjust your .ccf according to newly defined control parameters or coordinates. The newly defined coordinates will be treated as the other predefined coordinates, i.e. they can be visualized in spectra in on-line and off-line analysis.

Note that in the .lmf only those coordinates defined by the hardware are stored (those containing the complete hardware information). So you may address a given .lmf-file with different DAn.dll, as long as your DAn.dll complies with the DAq (the control parameters and coordinate definition required by the hardware must be consistent). To use your self-compiled you must copy your new DAn.dll to the main **CoboldPC** folder.

Compiler Requirements

To be able to build your own DAn.dll you need to fulfill the following software requirements

- MS Visual C++ 2010

Sample Projects for DAn.dll are available for Microsoft Visual Studio 2010 (see the Sources folder in your installation directory of **CoboldPC**).

To build your own DAq.dll you need to fulfill the following software requirements

- MS Visual C++ 2010

Sample Projects for DAq.dll are available for Microsoft Visual Studio 2010 (see the Sources folder in your installation directory of **CoboldPC**).

The sample source codes are only available if this option was selected during the installation of **CoboldPC**.

Note that the compiler is not part of the program package and the support from **RoentDefk** in building your own DAn.dll can only be directive. There are well documented program examples included in the program package, so a medium skilled programmer will be able to compile his own .dll-file if the hardware and software requirements are met. Usually the DAn.dll provided by us is written in C++.

How to change the source code of the DAn

If you use the Microsoft Visual Studio 2010 or higher please open the file "CDAn Standard.sln". In the compiler environment open the file "StandardDAn.cpp". Now you have the source code of the DAn on the screen.

When compiling the DAn please make sure to compile in "Release mode" and not in "Debug mode". The debug version of a program is always much slower (about a factor of 2) than the release version.

In the following it is assumed that the MS-Visual-C++-Compiler is used.

General structure of a DAn

In general a DAn contains four important sub-functions:

- CDAN_API LPCTSTR AnalysisGetInformationString()
- CDAN_API BOOL AnalysisInitialize(CDDoubleArray &EventData, CDDoubleArray &Parameters, CDDoubleArray &WeighParameter)
- CDAN_API void AnalysisProcessEvent(CDDoubleArray &EventData, CDDoubleArray &Parameters, CDDoubleArray &WeighParameter)
- CDAN_API void AnalysisFinalize(CDDoubleArray &EventData, CDDoubleArray &Parameters, CDDoubleArray &WeighParameter)

"AnalysisGetInformationString" is the function that provides an informational string for **CoboldPC**. This string is displayed during startup in the "Startup Banner" or in the "About Box". You can give for example a short hint about what the DAn is doing. This helps later to identify compiled DAn modules during startup.

"AnalysisInitialize" is called once at the beginning of the data acquisition when the "new" command is executed in **CoboldPC**. In this sub-function some global variables are initialized.

"AnalysisProcessEvent" is called for each event. Here all the calculations on the data are performed.

"AnalysisFinalize" is called when the LMF reached its end or the "stop" command is executed.

All data transfers between the DAn, DAq and **CoboldPC** is done via three arrays. These arrays are one dimensional arrays of type "double". The arrays are defined in a class. To access elements there are two methods possible. (The access has changes in comparison to older **CoboldPC** versions!). For example:

- EventData.GetAt(index)
- EventData.SetAt(index, value)

or

- double value = EventData[index]
- EventData[index] = value

The three arrays are

- 1) EventData-array

This array is one to one related to the list of coordinates which is defined in the CCF-file. **CoboldPC** reads the data either online from the TDC or offline from the disc and writes this raw data into the first part of the Coordinate-list (which is the same as the EventData-array). **CoboldPC** then enters the sub-function "AnalysisProcessEvent". Here all the calculations are done. The results of these calculations are written back into the same array EventData but below the raw data.

- 2) Parameters-array

This array is filled by **CoboldPC**. When a ccf-file is executed **CoboldPC** takes all Parameter definitions in the ccf-file and fills the values into "Parameters". Please note that the first parameter in the CCF-file has the index 1 whereas the first element in "Parameters" has index 0 (when programming in C; in Fortran the first index is 1). This is due to the indexing convention for arrays in C (or Fortran).

- 3) WeightParameter-array

This array can be used to transfer weight-parameters between the DAn and **CoboldPC**. This is needed if some spectra have to be filled with other values than 1.0.

Sample for a User defined DAn.dll written in C++

This code is based on the Standard DAn which is provided together with **CoboldPC**. However, the code has been simplified in some parts: It works only with the TDC8 and with a standard rectangular delay line detector (DLD). Naming convention: In the following code variables beginning with "i" are of type integers, "d" are of type double and "p" are pointers.

```

#define DAQ_VERSION4          20020408
#define DAq_VERSION          20020408
#define LMF_VERSION          5

#define DAN_VERSION          20020513
#define DAN_SUBVERSION      0x0007

////////////////////////////////////
// global variables definitions:
LARGE_INTEGER    lipTimeStamp;    // TimeStamp time information
int              ipTimeStamp;     // TimeStamp information type
double           dpTimeScaling;   // Time Scaling (ticks per second)
CTime            ctpLMFStartTime; // Start Time of LMF
int              iDaq_ID;        // parameter 8
double           dpTDCResolution; // parameter 20
int              ipTDCDataTypes;  // parameter 21
int              ipNumberOfChannels; // parameter 32
int              ipNumberOfHits;  // parameter 33
int              ipStartDaqData;  // parameter 105
int              ipStartDAnData;  // parameter 106
int              UseHit;          // parameter 107
double           dpTPCalX;        // parameter 110
double           dpTPCalY;        // parameter 111

double           dStartDAnCoordinates;
int              iDataOffset;
int              iIndexDataOffset;
int              ipStartDaqTDCData;

double           EventCounter;    // EventCounter for the data in LM-file

////////////////////////////////////
// AnalysisGetInformationString
////////////////////////////////////
// is called during startup procedure of CoboldPC
// return value is a string that will be displayed in
// the splash box and the about box
CString myInfoString;
CDAN_API LPCWSTR AnalysisGetInformationString()
{
    myInfoString.Format("RoentDek Simple V4 (Version %08d - %04d - %04d)",DAN_VERSION,DAN_SUBVERSION,LMF_VERSION);
    return LPCWSTR(myInfoString);
}

////////////////////////////////////
// AnalysisInitialize
////////////////////////////////////
// is called when new command is executed
// return value is false -> error during initialize
// true -> initialize ok

CDAN_API BOOL AnalysisInitialize(CDDoubleArray *pEventData,CDDoubleArray *pParameters, CDDoubleArray *pWeighParameter)
{
    _doldEventTime = 0.0;
    _dStartEventTime = 0.0;

    // transfer parameters
    // in the GetAt lines you'll find the following correction values
    // 0.1 for float to int conversion
    // -1 if parameter is for indexing
    ipTimeStamp      = (int)(pParameters->GetAt(1)+0.1);    // parameter 2
    dpTimeScaling    = pParameters->GetAt(4);                // parameter 5
    ctpLMFStartTime   = CTime((time_t)((int)(pParameters->GetAt(6)+0.1))); // parameter 7
    iDaq_ID          = (int)(pParameters->GetAt(7)+0.1);    // parameter 8
    dpTDCResolution  = pParameters->GetAt(19);               // parameter 20
    ipTDCDataTypes   = (int)(pParameters->GetAt(20)+0.1);    // parameter 21
    ipNumberOfChannels = (int)(pParameters->GetAt(31)+0.1);  // parameter 32
    ipNumberOfHits   = (int)(pParameters->GetAt(32)+0.1);    // parameter 33
    ipStartDaqData   = (int)(pParameters->GetAt(104)+0.1) - 1; // parameter 105
    ipStartDAnData   = (int)(pParameters->GetAt(105)+0.1) - 1; // parameter 106
    dpTPCalX         = pParameters->GetAt(109);              // parameter 110
    dpTPCalY         = pParameters->GetAt(110);              // parameter 111

    EventCounter = 0;

    iDataOffset = 0;
    iIndexDataOffset = 2;

    int iDataFormat = (int)(pParameters->GetAt(39)+0.1);
    switch(iDataFormat)
    {
        case LM_BYTE:
        case LM_SBYTE:
            break;
        case LM_SHORT:
        case LM_SSHORT:
        case LM_CAMAC:
            if(ipTimeStamp == 0) { iDataOffset = 0; iIndexDataOffset = 4; }
            if(ipTimeStamp == 1) { iDataOffset = 2; iIndexDataOffset = 2; }
            if(ipTimeStamp == 2) { iDataOffset = 4; iIndexDataOffset = 0; }
            break;
        case LM_LONG:
        case LM_SLONG:
        case LM_FLOAT:
            if(ipTimeStamp == 0) { iDataOffset = 0; iIndexDataOffset = 2; }
            if(ipTimeStamp == 1) { iDataOffset = 1; iIndexDataOffset = 1; }
            if(ipTimeStamp == 2) { iDataOffset = 2; iIndexDataOffset = 0; }
            break;
        case LM_DOUBLELONG:
        case LM_SDOUBLELONG:
        case LM_DOUBLE:
            if(ipTimeStamp == 0) { iDataOffset = 0; iIndexDataOffset = 1; }
            if(ipTimeStamp == 1) { iDataOffset = 1; iIndexDataOffset = 0; }
            if(ipTimeStamp == 2) { iDataOffset = 1; iIndexDataOffset = 0; }
            break;
        default:
            break;
    }
}

```



```

    }

    // correct start address of DAq coordinates:
    // start address of DAqData in coordinate block in CCF-file:
    if(ipStartDAqData < 0)
        ipStartDAqTDCData = iDataOffset;
    else
        ipStartDAqTDCData = ipStartDAqData + iDataOffset;

    pParameters->SetAt(104,double(ipStartDAqTDCData-iDataOffset)+1);

    // start address of DANData in coordinate block in CCF-file:
    if(ipStartDANData < 0) {
        ipStartDANData = ipStartDAqTDCData + (ipNumberOfChannels * (ipNumberOfHits+1));
    }

    pParameters->SetAt(105,double(ipStartDANData)+1);           // write information back to parameter 106

    return true;
}

////////////////////////////////////
// AnalysisProcessEvent
////////////////////////////////////
// is called when "new" command is executed
//CDAQ_API LARGE_INTEGER DAqTimeStamp;
//__declspec(dllimport) int DAqTimeStamp;
__declspec(dllimport) int DAqTimeStamp;

CDAN_API void AnalysisProcessEvent(CDoubleArray *pEventData,CDoubleArray *pParameters, CDoubleArray *pWeighParameter)
{
    double    AbsoluteEventTime; // ns since start
    double    DeltaEventTime;    // This Time - PreviousTime
    double    x1,x2,y1,y2;       // TDC data
    double    x,y;
    double    sumx,sumy;
    int       n1,n2,n3,n4;       // Number of hits for channels 1 to 4
    int       coordinate_address;

    // correct position of Event Data addresses due to time stamp information
    // at the beginning of the coordinate list in ccf-file.
    CorrectEventDataPosistion(pEventData,iIndexDataOffset,ipStartDANData);

    // get time-stamp information
    if(iDataOffset)
    {
        AbsoluteEventTime = GetEventTime(pEventData,pParameters);           // AbsoluteEventTime
        DeltaEventTime = GetDeltaEventTime(pEventData,pParameters);         // AbsoluteDeltaEventTime
    }

    EventCounter = EventCounter + 1;

    // Get Status Information

    // Extract TDC-data from the pEventData-array:
    UseHit = 1;
    coordinate_address = ipStartDAqTDCData+0*(ipNumberOfHits+1);
    n1 = (int)( pEventData->GetAt(coordinate_address) + 0.1 );

    coordinate_address = ipStartDAqTDCData+1*(ipNumberOfHits+1);
    n2 = (int)( pEventData->GetAt(coordinate_address) + 0.1 );

    coordinate_address = ipStartDAqTDCData+2*(ipNumberOfHits+1);
    n3 = (int)( pEventData->GetAt(coordinate_address) + 0.1 );

    coordinate_address = ipStartDAqTDCData+3*(ipNumberOfHits+1);
    n4 = (int)( pEventData->GetAt(coordinate_address) + 0.1 );

    coordinate_address = ipStartDAqTDCData+0*(ipNumberOfHits+1) + UseHit;
    x1 = pEventData->GetAt(coordinate_address);

    coordinate_address = ipStartDAqTDCData+1*(ipNumberOfHits+1) + UseHit;
    x2 = pEventData->GetAt(coordinate_address);

    coordinate_address = ipStartDAqTDCData+2*(ipNumberOfHits+1) + UseHit;
    y1 = pEventData->GetAt(coordinate_address);

    coordinate_address = ipStartDAqTDCData+3*(ipNumberOfHits+1) + UseHit;
    y2 = pEventData->GetAt(coordinate_address);

    // convert to nanoseconds
    x1 = x1 * dpTDCResolution;
    x2 = x2 * dpTDCResolution;
    y1 = y1 * dpTDCResolution;
    y2 = y2 * dpTDCResolution;

    x = dpTPCalX*(x1 - x2)/2.0;
    y = dpTPCalY*(y1 - y2)/2.0;

    // sums and differences
    sumx = x1 + x2;
    sumy = y1 + y2;

    //////////////////////////////////////
    // write all data back to coordinates in ccf
    //////////////////////////////////////
    pEventData->SetAt(ipStartDANData+0,AbsoluteEventTime);
    pEventData->SetAt(ipStartDANData+1,DeltaEventTime);
    pEventData->SetAt(ipStartDANData+2,EventCounter);
    pEventData->SetAt(ipStartDANData+3,n1);
    pEventData->SetAt(ipStartDANData+4,n2);
    pEventData->SetAt(ipStartDANData+5,n3);
    pEventData->SetAt(ipStartDANData+6,n4);
    pEventData->SetAt(ipStartDANData+7,x1);
    pEventData->SetAt(ipStartDANData+8,x2);
    pEventData->SetAt(ipStartDANData+9,y1);

```

```

        pEventData->SetAt(ipStartDANData+10,y2);
        pEventData->SetAt(ipStartDANData+11,x);
        pEventData->SetAt(ipStartDANData+12,y);
        pEventData->SetAt(ipStartDANData+13,sumx);
        pEventData->SetAt(ipStartDANData+14,sumy);
    }

//////////
// AnalysisFinalize
//////////
// is called when analysis is stopped (not paused!)
CDAN_API void AnalysisFinalize(CDoubleArray *pEventData,CDoubleArray *pParameters, CDoubleArray *pWeighParameter)
{
}

```

Sample ccf-file for the above sample-DAn

The following ccf-file works with the above sample DAn. Please note how the list of Parameters and the list of Coordinates is one to one related to the arrays "pParameters" and "pEventData" in the above source code.

```

restart

Parameter 1,0      ; io-hardwareaddress of the ATMD-Boards
Parameter 2,2      ; save TimeStamp
Parameter 3,5      ; system Timeout Time in s

Parameter 14,0     ; trigger mode for common input

Parameter 30,30    ; Event Open Time in us for detecting an event.
Parameter 32,4     ; number of Channels (reread during offline)
Parameter 33,1     ; number of hits (reread during offline)

parameter 45,0x80  ; gate delay (in common start always use 0x80)
parameter 46,1080  ; gate open
parameter 47,0     ; write empty events
parameter 48,1     ; trigger at falling edge
parameter 49,0     ; trigger at rising edge

Parameter 105,0    ; Parameter 105 = Start of DAq Data for DAn (Start Coordinate)
                  ; 0 = automatic
Parameter 106,0    ; Parameter 106 = Start of DAn Data (Start Coordinate)
                  ; 0 = automatic
Parameter 110,0    ; Parameter 110 = pTPCalX = Time to Point calibration factor for x (mm/ns)
Parameter 111,0    ; Parameter 111 = pTPCalY = Time to Point calibration factor for y (mm/ns)

; timestamp:
; -----
Coordinate TRaw1,TRaw2,TRaw3,TRaw4

; TDC raw Data:
; -----
Coordinate TDC1Hits,TDC1DataHit1
Coordinate TDC2Hits,TDC2DataHit1
Coordinate TDC3Hits,TDC3DataHit1
Coordinate TDC4Hits,TDC4DataHit1

; DAn results:
; -----
Coordinate AbsoluteEventTime
Coordinate DeltaEventTime
Coordinate EventCounter
Coordinate n1,n2,n3,n4
Coordinate x1,x2
Coordinate y1,y2
Coordinate x,y
Coordinate sumx,sumy

; Spectra definitions:
; -----
define1 0,32,1,n1,,none,always,n1
define1 0,32,1,n2,,none,always,n2
define1 0,32,1,n3,,none,always,n3
define1 0,32,1,n4,,none,always,n4

define1 0,2500,1,TDC1DataHit1,,none,always,TDC1 Hit1
define1 0,2500,1,TDC2DataHit1,,none,always,TDC2 Hit1
define1 0,2500,1,TDC3DataHit1,,none,always,TDC3 Hit1
define1 0,2500,1,TDC4DataHit1,,none,always,TDC4 Hit1

define2 -300,300,1,x,-300,300,1,y,,none,always,xy_plot

new
start
show status

```

LMF Data Format

The List-Mode data file consists of 2 sections. The first section is the data-header. The second is the data-area. The main header is constructed as follows.

if LMHeaderVersion is 0 or 0x74656	if LMHeaderVersion is 0x74657 for the lower 24bits
unsigned int LMHeaderVersion	unsigned int LMHeaderVersion;
unsigned int LMDDataFormat;	unsigned int LMDDataFormat;
unsigned int LMNumberOfCoordinates;	unsigned __int64 LM64NumberOfCoordinates;

unsigned int	LMHeaderSize;	unsigned __int64	LM64HeaderSize;
unsigned int	LMUserHeaderSize;	unsigned __int64	LM64UserHeaderSize;
unsigned int	LMNumberOfEvents;	unsigned __int64	LM64NumberOfEvents;
CTime	LMStartTime;	CTime	LM64StartTime;
CTime	LMStopTime;	CTime	LM64StopTime;
CString	LMVersionString;	CString	LM64VersionString;
CString	LMFilePathName;	CString	LM64FilePathName;
CString	LMComment;	CString	LM64Comment;

unsigned int is an unsigned 32bit integer number.

unsigned __int64 is an unsigned 64bit integer number.

CTime and CString are classes of the Microsoft Foundation Class 4.x of the Visual C++ Compiler of Microsoft.

LMDataFormat can have the following values:

```
// definitions for main program event data format
#define LM_BYTE      1      // 8bit integer
#define LM_SHORT     2      // 16bit integer
#define LM_LONG      3      // 32bit integer
#define LM_FLOAT     4      // 32bit IEEE float
#define LM_DOUBLE    5      // 64bit IEEE float
#define LM_CAMAC     6      // 24bit integer
#define LM_DOUBLELONG 7      // 64bit integer
#define LM_SBYTE     8      // signed 8bit integer
#define LM_SSHORT    9      // signed 16bit integer
#define LM_SLONG     10     // signed 32bit integer
#define LM_SDOUBLELONG 11   // signed 64bit integer
#define LM_USERDEF   -1     // user will handle the
                          // reading
                          // of file and filling of
                          // EventData Array
                          // except reading the main

                          // header file
```

If the lower 24 bits of the version id is equal to 0x74657 then the upper bits are coding if DAn, DAq source code is included as well as the CCF history. If that bit is set then the appropriate code is included.

DAQ_SOURCE_CODE	bit 31	0x80000000
DAN_SOURCE_CODE	bit 30	0x40000000
CCF_HISTORY_CODE	bit29	0x20000000

If the LMUserDefinedHeaderSize is not equal to zero then a user defined header will follow. The size of this data block is stored in the LMUserDefineHeaderSize variable of the main header block.

After this header block the event data follows as described in DAq.DLL source code.

Config File is readonly ...

The full message is:

Config File is readonly or could not be created.
See help file for further information!

CoboldPC aborts now!

There are two possible names for the config file depending on the environment your are running.

- CoboldPC2011-10.1_x86.Config for Win32 or (x86)
- CoboldPC2011-10.1_x64.Config for Win64 or (x64)

The Config file stores all settings of CoboldPC like

- Window Position
- last loaded DAq/DAn dll module
- color settings

and more.

These informations are no longer stored in the registry!

Possible locations for the config files are.

1. path where CoboldPC were started
2. User application folder as %APPDATA%\RoentDek Handels GmbH\CoboldPC 2011\10.1.1109.1
3. User Temporary folder as %temp%

If you have no write access or the file is write protected then the next location will be used. If the position 1-3 are failing then CoboldPC will terminate because without config file the program can not run!

Could not load compressed ...

The full message is:

Could not load compressed CoboldPC document
due to missing or invalid COM registration.

See help file for further information!

CoboldPC is using the .NET Framework as part of the whole program. Functionality of the .NET Framework is located in special Dynamic Link Libraries (DLL) using the COM interface. These DLLs have to be registered in a special way and must not be moved from one location to another. This registration is done during the setup process.

You may try to repair/reinstall CoboldPC to fix this problem.

If you still get the same message then contact RoentDek for further instructions.

Could not save compressed ...

The full message is:

Could not save compressed CoboldPC document
due to missing or invalid COM registration.

See help file for further information!

CoboldPC is using the .NET Framework as part of the whole program. Functionality of the .NET Framework is located in special Dynamic Link Libraries (DLL) using the COM interface. These DLLs have to be registered in a special way and must not be moved from one location to another. This registration is done during the setup process.

You may try to repair/reinstall CoboldPC to fix this problem.

If you still get the same message then contact RoentDek for further instructions.

MRDCompress.dll ...

The full message is:

MRDCompress.dll
is either not available or not registered!
Compressed CoboldPC document files
can not be opened or saved.

Please see help file section 'Installation'
for further information!

CoboldPC is using the .NET Framework as part of the whole program. Functionality of the .NET Framework is located in special Dynamic Link Libraries (DLL) using the COM interface. These DLLs have to be registered in a special way and must not be moved from one location to another. This registration is done during the setup process.

You may try to repair/reinstall CoboldPC to fix this problem.

If you still get the same message then contact RoentDek for further instructions.

MRDUtilities.dll ...

The full message is:

MRDUtilities.dll
is either not available or not registered!
Compressed CoboldPC document files
can not be opened or saved.

Please see help file section 'Installation'
for further information!

CoboldPC is using the .NET Framework as part of the whole program. Functionality of the .NET Framework is located in special Dynamic Link Libraries (DLL) using the COM interface. These DLLs have to be registered in a special way and must not be moved from one location to another. This registration is done during the setup process.

You may try to repair/reinstall CoboldPC to fix this problem.

If you still get the same message then contact RoentDek for further instructions.

HIDInfo.dat file is missing ...

The full message is:

HIDInfo.dat file is missing!
Please contact RoentDek for further help.

HIDInfo.dat file is a special hardware configuration file. This file is normally installed during the setup procedure. It is normally stored and used from the same location as the config file (see Config file).

Possible locations for the HIDInfo.dat files are.

1. path where CoboldPC were started
2. User application folder as %APPDATA%\RoentDek Handels GmbH\CoboldPC 2011\10.1.1109.1
3. User Temporary folder as %temp%

If you have no write access or the file is write protected then the next location will be used. If the position 1-3 are failing then CoboldPC will terminate because without HIDInfo.dat file the program can not run!

During the startup procedure of CoboldPC the HIDInfo.dat file is updated/created automatically if network access is available.

Changes in Version 4.2.1.2

New or changed Functions

Function [view](#)

Instead of parsing the spectrum number it is also allowed to parse the spectrum title. In this case the title has to be typed exactly as it was defined. The meaning of the rest of the parameters is unchanged.

Error in Condition definition solved. If a Coordinate is not defined a message will be displayed on screen and the definition will be rejected.

The [CopySpectrum](#) is now selectable in the *Spectrum handling commands* section of the HELP file.

The [show](#) command is now implemented.

New fitting-function [fit](#) was added to do basic fits.

New [CalibrateSpectrum](#) function.

New [Remove](#) function.

A number is added to the marker-information on screen for easier identification and deletion of marker or marker pairs.

Printout is now correct for color printers. Monochrome and color printout should have the same size. For better achieving the image is now slightly smaller.

All internal subdocuments have now their own document number. This will allow the use and conversion of older document version in newer versions Cobold programs.

Now all number spectrum information is stored in floating point data format. This allows non integer numbers for the definition of a spectrum.

Printing 2-dimension spectra with a monochrome printer in gray scale will no longer print a white pixel for the highest counts.

[Define2Dimensional](#) is expanded for a Mode2d parameter to predefine the display type of a 2-dimensional spectrum.

New [Delete](#) function to delete parts of a document.

A date/time information is now available in a displayed spectrum.

Help command is now also available through command line.

Known Bugs

Changes in Version 4.2.1.3

New or changed Functions

Function [open](#)

This function now reads also ATARI Dump Files. The filename extension must be *.dmp*!

Function [restart](#)

This function is new.

Function [integratespectrum](#)

This function is new.

Function [remove marker](#) has been altered

The cursor channel is now no longer needed.

Function [projectspectrum](#)

This function is new.

Changes in Version 4.2.1.4

New or changed Functions

Function [show](#)

All index numbers start with 1. But in the user dependent part of the program the index still starts with 0 in C or C++ and with 1 in Fortran.

Do not forget to change all existing Cobold Command files by adding 1 to the limit number. Limit numbers smaller or equal 0 are not allowed!

Marker and Cursor function switched their meaning.

If you are forced to save the prompted file name should be now correct!

The Command-Input line has been moved from the status bar to a separate command bar. A clear history buffer button is also available now.

Contour Plot for two-dimensional spectra is now possible, but when switching between windows the displayed spectrum can be redrawn several times before coming to an end.

Major improvement in EventFileRead routine reduces the file management time and speeds up the reading of ListModeData up to a factor of 4.

Update of the [show](#) status command has been fixed to an automatic update for every second.

For Cursors and Markers there is now an X and Y marker available. Only in special cases like projection there is only an X or Y marker present.

New [grid](#) function.

Problems in defining a condition with negative numbers are solved.

New [NoDisplay](#) function has been added.

Cobold do not crash in data acquisition any longer if the device is full.

Loading old ATARI-Dump files should not crash any longer. The crash was caused by a not defined coordinate in a spectrum (either one or two-dimensional).

Changes in Version 5.0.1.5

New or changed Functions

For better performance the drawings of the screen are prepared in memory. Therefore you can no longer see how a spectrum is drawn.

Due to memory consumption this new drawing technique is only available under Windows NT 3.51 or higher.

Also the Zooming possibility (over menu command or over the task bar) is only available under Windows NT 3.51 or higher.

If the display is set to a fixed zoom resolution there is no need for rescaling of the displayed spectrum if the user uses resizes the window or uses the sliders to see a different part of the spectrum. Only in Page-Width and Full-Page mode a redraw of the spectrum is needed when the display window is resized. This makes the window handling much faster than in the previous version.

The document format has been modified to store:

- Last displayed spectrum
- Zooming of the display
- Slider position of the display
- Command history

The document is converted on the fly. All newly stored CoboldPC documents are stored in the new format.

The format of calling the user defined functions in the ANALYSIS and DAQ DLL where changed for better performance and handling with the main program (see the sample DLL source-code files).

Getting a new document in CoboldPC will now delete the history automatically.

The cobold.ini file in the windows root directory is no longer needed. You may delete this file. All informations are now stored either in the registry or in the document file.

CoboldPC now detects if you are trying to read a newer document file. If you try this the load aborts and the restart function is called internally. Older documents are converted to the actual file format during read in.

The reading of ATARI Dump Files will now load dumps correctly even if a spectrum is defined with empty coordinates or conditions.

Changes in Version 5.0.1.6

New or changed Functions

For better performance with new hardware modules it was necessary to modify the List-Mode Header-File format. Older List-Mode data can be read with this new CoboldPC version without any changes. A new flag in the List-Mode Header is indicating now if the data in the List-Mode file is written in integer (8, 16 and 32 bit) or in floating point (32 and 64 bit) values. The Data-Format must be defined in the user dependent DAQ module.

The NODISPLAY command now works also under Windows NT.

Displacement in printout with a resolution other than 600 dpi is now corrected and resolution independent.




The DISTORTIONVALUE function gives now the possibility to avoid structures that comes from incompatible channel definition of the spectra and the channel offered by the DAQ hardware.

Hopefully the "show status" command will now no longer result in an exception with a program termination.

Creating a listmode file in a security protected area will no longer crash CoboldPC. The program no longer crashes if writing a listmode file and the disk gets full.

Passing a CoboldPC a document file will no longer prevent CoboldPC from starting.

Save button is now disabled if the document need no save. It gets active if changes a made to the document structure.

In the status line there is a new Indicator field inserted for easier status of a running acquisition.  indicates that the command **new** or **pause** were executed successful.  indicates a running offline dataanalysis (read from a listmode file) while  indicates a running online data acquisition (taking data from the hardware). If no indicator is present no active acquisition command is running.

High resolution timers are now available in the Cobold.dll file.

Changes are made to the [view](#) function. It is now possible to use this function to rescale axes.

For better views on dark screen the grid of a spectrum is displayed now a lighter color (same color as the surrounding bars).

Changes in Version 5.0.1.7

New or changed Functions

It is now possible to enter also hexadecimal numbers as a [parameter](#) value.

The new menu command [preferences](#) allows now to set the color for each CoboldPC color-object. The definition is made for screen and print output separately.

The problems with flat toolbar-buttons should be solved now.

Redraw of CoboldPC window is now handled correctly if the program is in cursor mode.

Changes in Version 6.0.1.8

New or changed Functions

CoboldPC will now remember the window state and size

Wait command is now implemented. You should use this command to hold Batch-File execution till EOF is reached.

Import and Export ASCII will now read or save an ASCII file into or from the displayed spectrum.

Clipboard data will can be erased easily any time. If Clipboard data is present at program termination, CoboldPC will now ask to delete this data.

Indicator SPD (Speedometer) and LMS (List-Mode-Status) is now present in the status-bar.

Condition name will now be displayed in the title bar.

The Display of the conditions window can now be shown in the spectrum with the view condition command. This command will affect only the displayed spectrum. To turn off the display globally set condition color to background color.

Overlay of spectra is now implemented.

Overlay parameter added to remove function.

In delete and remove function all lists will now first be sorted and then executed. This allows now to give the command "delete sp,1,2,3" to delete spectrum 1,2 and 3.

The integrate command will now store the results in the spectrum. This information will also be saved with the document. The right and top boundaries can now be excluded.

There is now a recent file list for CoboldPC documents, command-files and list-mode-files in the File menu. The open button in the Tool-Bar has now an selector open for the recent document list. Additionally there is also an open button for command files with recent ccf file list selector available.

In printout a header information can now be displayed. This flag can be defined via preference menu. The header information consists of the document title, comment of the last list mode file and the information sting from DAq and Analysis dll.

The last paths for document, list-mode, command-file, import and export file will now be saved and be the default on next call.

The paths for document, list-mode, command-file, import and export file can be set globally by the "setpath" command.

List Mode data format is changed dramatically but old LMF files can still be processed. Now the size of comment is no longer limited. In DAq dll it is possible to implement a user defined additional header for further processing.

It is possible to start many CoboldPC instances. Calling the DAq module in hardware acquisition mode is only possible if only one instance of the program is running.

DefineExperiment gives the user the possibility to store related informations with the Document.

You can also define a Document Tile here. This title will be printed with the header information on the spectrum printout.

Offline correction of the GPI differential non linearity with the gp1correct function.

CoboldPC will no longer use Idle-Time processing to process online or offline data. This processing is now done in a separate thread. This gives better performance in a multi processor machine. The main advantage is that the data taking or analyzing can no longer be stooped by dialog boxes.

A sound can now indicate a command failure or the end of command file execution.

The execution of list mode files will now be stored in a list mode file history. The history can be displayed with the show listmodehistory command.

The help command accepts now a search topic.

A two dimensional spectrum can be rotated by the rotate command.

A fit can now be subtracted from the spectrum.

In preferences it can be defined to load the last loaded CoboldPC document.

File types, dcf, cdf, ccf and lmf can now be routed to CoboldPC or notepad program.

The two dimensional color bar is now configurable.

The command line print command will no longer bring up a printer dialog box. This allows the printout of multiple spectra in a command file.

New gauss fit function implemented.

New polynom fit function implemented.

The setaxistext allows now to set the text for the x or y axis.

The update command allows the update of the displayed spectrum after a predefined time or number of counts. The update can be terminated with the remove command or the remove update button in the Show-Bar.

New Button in Show-Bar to clear all spectra

Calibration Command will no longer be able to remove the calibration with the CALIBRATION REMOVE command. Instead use the REMOVE command.

If a command cannot be identified, CoboldPC will try to execute it as CoboldCommandFile (*.ccf). If execution fails the "unknown command" sound is playing if selected.

In define1dimensionalspectrum and define2dimensionalspectrum it is now possible to define the initial log mode of the spectrum.

Project function is changed to support auto-definition of the project-to spectrum. The command-files function is not affected. See function description for detailed information.

Analysis.Dll has been renamed to DAn.DLL (Data Analysis Dynamic Link Library)

To make Cursor/Marker handling easier only the crosshair cursor will be displayed. The mouse cursor is now disabled to display all of the data.

The show command has **MARKER** as new option to display all marks in a spectrum in a list.

If a spectrum list is shown a spectrum can be select for displaying by clicking the **Go To** button or double clicking the spectrum line.

Build 077

The delimiter and the decimal character can be defined now via menu command **File\Preferences\Define Decimal Symbol and Delimiter**. These settings will be stored automatically in the CoboldPC program.

Copy/Paste, Import and Export functions are using now the **Decimal Symbol and Delimiter** setups.

In PasteASCII, ImportASCII function it is now possible to specify if the spectrum should be initially cleared before the Paste/Import. Corresponding functionality was added to the menus.

CoboldS.Dll is no longer needed in the system.

HM1 and atmd_pci.dll ([acam mess electronic](#)) are needed to support ATMD with HM1 or GP1.

HM1-Multi hit DAn module: An event counter has been added. See Parameter and Coordinate file for further information. Corresponding CCF files where adapted accordingly.

Definition of spectra gives now the possibility to define minimum, maximum and binsize over parameters. See [Define1DimensionalSpectrum](#) or [Define2DimensionalSpectrum](#) for further information.

In 2 dimensional spectra the automatic z-axis scaling will no longer adjust to the range of 10-90% of the real minimum and maximum of the z-axis. The range is now 0-100%. The exact minimum will not be displayed.

For better display in printouts all default yellow colors (Title, AxisText, AxisFrame and ActiveMarker) have been reduced in louninosity by 25%.

In the status box for LMF-Offline read and Online data taking the EST remaining is now more aqurate.

Termination of the program no longer occurs when in a command file the new command for reading an LMF with zero event length is followed by a status.

Error Correction

In Countour-smooth like "contoursmooth 72,x,74" the source spectrum was copied to the destination spectrum. This complete copying confuses the "next" and "previous" spectrum command. This has been corrected.

When canceling the dialog box in the execute command will bring up the dialog box again. This has been corrected.

Project command will now check spectra number and type more carefully to avoid CoboldPC from a crash.

In converting the "real" channel to an "internal" channel the conversion from double to integer format is now accurate. This will minimize a "moire" effect in sorting data to spectrum.

Marker function now detects out of definition setting and corrects it.

DAq dll

The CAMAC DAq dll is now able to send or receive more than one CAMAC word.

This DLL will now write by default 32bit data. The DLL is also available for writing 24bit or 16bit (separate dll's)

Known bugs

Under certain conditions it is possible that the percent value and display of the progress bar in the status-box are wrong. This seems to be an error in the MFC42 or MSVCRT dll.

Changes in Version 6.2.0.81

New or changed Functions

Build 078

Libraries for HM1.dll and Cobold.dll changed. So please use the new hm1.lib and cobold.lib file to compile DAn and/or DAq modules.

Add grippers to toolbars.

Add size-gripper to ListBox.

HM1SH Module has been altered to support also NoResolutionAdjust Mode. Please See the CCF and Parameter.txt file.

Define1DimensionalSpectrum and Define2DimensionalSpectrum functions has been extended to define the axis text directly during spectrum definition.

Automatic rescaling of axis has been modified to get same results in scaling using different rescale commands.

ASCII export of a one dimensional spectrum will no longer include a double LF.

View a spectrum using spectrum name is working again.

All file dialog boxes, except the List-Mode file dialog boxes do now support the Windows 2000 GUI standard.

New ShellCommand function is executing a Win32-Command shell program.

The search-extension in the LM-file selector box has been changed from *.lmf to *lmf*.

All Text outputs to clipboard changed from data format OEMTEXT to TEXT.

Condition definition will check now if from- is less then to-value. If not both values are exchanged automatically.

Correction of ListMode File Event Number. In an LMF was always one event more written than defined. This is now corrected.

SetSpectrumTitle command added to Help File.

Setting a marker outside of the spectrum definition will no longer terminate CoboldPC with an error.

During data sort it is now possible to weigh data instead of always adding one!

Affected functions are:

[Define1DimensionalSpectrum](#)

[Define2DimensionalSpectrum](#)

[Show](#)

New functions are

[WeighParameter](#)

To implement this functionality it was nessecary to modify the CoboldPC document format. Also the DAn function call have changed. Now you must apply also the WeighParameters!

Build 080

[Remove](#) and [Delete](#) function modified. For lists it is now possible to enter a -. Instead of delete sp,1,2,3,4 you may write delete sp,1-4

Now it is possible to open an still writing list-mode-file in a second channel.

If ColorBar has no color information CoboldPC no longer crashes.

During expansion of a spectrum the delta values are displayed for the second cursor input. This "delta display" is also available in several other "interval functions".

A new type of spectrum definition has been introduced. [DefineMultiViewSpectrum](#). In this spectrum you may display (define) up to 4 spectra. Many functions are disabled while displaying a MultiSpectrum on screen like cursor, calibration

Rescale of x,y or z applies to all spectra when displaying a MultiSpectrum. Update will rescale only the last "free" axis as usual.

Each DAQ Module has a certain set of User-Information storing setup information for the taken run.

It is now possible to copy or paste a spectrum in binary format to the memory (clipboard). Paste will always append the spectrum to the actual spectrum list. The copy/paste command also copies most of the relevant spectrum data.

The DAn module has been extended by the AnalysisFinalize function. This function is called before the DAn is closed.

Build 081

CoboldPC is now "CoboldPC 2002"

New Preference item "Ignore Modified Flag". This allows to ignore the Document Modification flag to avoid questioning on exit. With a modified document CoboldPC can be exited without saving the document and without any further question.

In the "Select New-Function Type" sheet of the New command the state of the buttons are memorized. When entering this box you'll always get the last selection.

Help of GP1Correct function corrected for parameters.

Delimiter for ASCII import/export default changed to "," if not defined.

CAMAC.INI is now CAMAC.CIF (CAMAC Information File). The file is still in ASCII. The CIF file extension is bound to the Notepad program after installation of CoboldPC 2002

Build 082

Update command now recognize also an implied delete. See [UpdateSpectrum](#)

The LMF File Selector box now has 2 text lines for the DAQ-Info text

V5 LMF version introduced to all DAQ dlls (since V3)

Using Update-Function over a long period of time will no longer result in "empty displays". Memory/Handle leak is eliminated.

V5 DAQ introduced to Standard DAn. This includes changes:

- Xvw calculation corrected
- HM1 ABM mode implemented
- 2HM1 support implemented (subversion 0006 of DAn)

CoboldPC crash eliminated when wrong data is given for weighparameterindex in the two dimensional spectrum definition command.

Build 083 (v4)

TDC8HP system introduced to CoboldPC

New LMF Data types for signed 8, 16, 32 and 64 bit data implemented

CAMAC handles now the 24Bit Time Stamp

LONG Time Stamp implemented

In DAn changed calculation for Hex Anode

```
Xuv = x + pOPx
Yuv = 1/sqrt(3) * (x-2y) + pOPy
Xuw = Xuv
Yuw = 1/sqrt(3) * (2w-x) + pOPy
Xvw = (y+w) + pOPx
Yvw = 1/sqrt(3) * (w-y) + pOPy
dX = Xuv - Xvw
```

```
dY = Yuv - Yvw
```

In TDC8 modules orientation of hits reversed in common stop mode

In TDC8 modules hit orientation now corrected

Help file entry "Set channel to value" corrected

CoboldPC 2008 9.1.812.1

New Functions

- CoboldPC is now compiled with the new Visual Studio 2008 compiler. CoboldPC now uses partially the DotNet Framework. Therefore the Framework 1.0, 1.1 2.0, 3.0 and 3.5 should be installed on that computer. Installation file of the DotNet Framework can be obtained by Microsoft (Web-Site).
- Version changed to CoboldPC 2008 - 9.1
- Command-Line version for CoboldPC provides now the possibility to send CoboldPC command from the command line or by using scripting functionality
- Command-Line version now knows the parameters -DAqState -DAqMode -CommandRunning in command-line mode as well as in GUI (graphical user interface) mode.
- EventCounterReset using Parameter. In the Standard-DAn.dll in the functions "AnalysisInitialize" and "AnalysisProcessEvent" parameter 140 will be tested. If set (set = 1) the EventCounter is set to 0 (Zero). After testing it'll be reseted to "not set = 0"
- New Parameter 78 in the HPTDC8 DAq.
VHR parameter (78) implemented.
If VHR is set then parameter 20 is modified to indicate 100ps / channel time resolution. The data of the TDC will be shifted by 2 to the right (means divided by 4. 25ps->100ps).
- Add Preferences PreferredSaveFileType included into menu "FILE"
- New commands
 - XMin
 - XMax
 - YMin
 - YMax
 - ZMin
 - ZMax
 implemented as command. This will modify accordingly the spectrum definition of the displayed spectrum
- Keyboard Command during a displayed spectrum.
Pressing Shift + Cursor will move the displayed area about 80% of the scale in that direction. Up- and Down- for y-axis movements and Left- and Right-Cursor for the x-axis
- The scaling update during Shift + Cursor key command is disabled.
- Expand command will no longer rescale the last axis (y-axis for 1D and z-axis for 2D)
- "view" command modified
Added parameter n. n = norescale
Added parameter l. l = rescale last axis
Added parameter r. r = full rescall of all axis
- New compressed data types added for storing CoboldPC Documents.
dcfz = CoboldPC Document File Zipped
dcfd = CoboldPC Document File Deflated
The default data type is now dcfz if not otherwise specified. During writing the used file extension defines the data format (compression).
- DAn.dll and DAq.dll are now bind to CoboldPC using "Late-Binding" technique. This is done for future functionalities using analysis and data taking dll.
- DAn.dll and DAq.dll can now be replaced by a different DAn or DAq module during runtime of CoboldPC. To support this functionality there a three new menu commands and two new command line commands available. For detailed information please refer to the file menu help section.
- Due to the changes in the load mechanism of the DAn/DAq Dlls the names have also been changed.
The filename structure of the daq.dll is now of the type ABC.dll
A = "DAq"
B = "Testing" or "" for final versions
C = ModuleName like "HM1", "HPTDC8" etc.
The filename structure of the dan.dll is now of the type ABC.dll
A = "DAq"
B = "Testing" or "" for final versions
C = "Standard" or something else for module description
- DAn/DAq information text will now display the word "debug" before "version information" text if the module is compiled in DEBUG mode
- DAq module for HM1 and TDC8PCI2 and HPTDC8 changed mode to LMF6 and USER_DEF to support variable event length. The DAq can still read LMF5 old style LMF files.
- Name change from SoftwareDependent.cpp to SandardDAn.cpp. The file now includes all information.
- Cursor focus is set automatically to command bar when CoboldPC window is activated or clicked
- New implemented CUserSpectrum class. This class can be used to manipulate (i.e. fill) spectra from DAn source code (i.e. multiple fill of one spectrum for one event).
- EndBatchJobs command implemented as menu, toolbar and batch command
- New command "coordinateset" to allow multi fill of a spectrum (see spectrum definition)
- HPTDC DAq - VerifyFlash functionality added

Modified behaviour

- Menu icons are now enabled by default unless not otherwise specified.
- Registry - Classes Root type changed from Cobold.LMFFile to Cobold.ListModeFile
- Improved error information during command file execution
- BinSize displayed in spectrum. Will not be displayed in a calibrated spectrum or a spectrum where the axis are defined by "setaxistext"
- Size of Spectrum-Data array will be displayed in the spectrum list
- Size of Spectrum-Data array will be printed using the spectrum list
- "Update" command expanded by parameter "all" (all will rescale all defined spectra).
- New-Box modified. Only "Read from ListMode File" or "Read from Hardware" will be visible beside the "Select New-Function Type" tab. Radiobuttons on the "Select New-Function Type" will change "Read from ListMode File" or "Read from Hardware" Tab view.
- The following parameters have been removed from the HPTDC8 CCF files.
parameter 45,0x80 ; gate delay (in common start allways use 0x80!!!!)
parameter 46,1080 ; gate open
parameter 47,0 ; write empty events
parameter 48,1 ; trigger at falling edge
parameter 49,0 ; trigger at rising edge
- CDoubleArray class now can used 1D and 2D indexer. All CDoubleArray pointer definitions in DAq and DAn source codes are now references in stead of pointers!
- List-Boxes from the "show" command can now be sorted by any tab. The sort direction will be indicated in the tab
- List-Boxes will now storing their position and size. New button for "default sizes" implemented.
- ListView for LMF History expanded. Informations for
LMF-Events
LMF-Start
LMF-Stop
LMF-Comment
are added

- No Support for Windows 95. All references have been deleted
- Cursor/Marker information in Help File updated for missing rphi cursor/marker
- Self registering file types
dcfz, dcfid
added. dcfz = CoboldPC Document File Zipped, dcfid = CoboldPC Document File Deflated
- "DT_" values in lminfo.h changed from 0..3 to 0x00.. 0x11 (pseudo binary)
- #define DAQ_ID_TCPIP 0x0009 - added in lminfo.h for future use
- CoboldPC2008 no longer deletes private settings if using different versions
- "new" command modified to support "NOANALYSIS" in "HARDWARE" as well as in "FILEREAD" mode.
- AnalysisProcessEvent function in the DAn.dll is now of data type int and therefore returns a value. (value is undetermined and will be used and defined later).
- AnalysisFinalize function in the DAn.dll is now of data type int and therefore returns a value. (value is undetermined and will be used and defined later).
- SubtractFit for GaussFit will no longer subtract the Gaussfit offset.
- In the Show spectrum list box the default button is changed from "Done" to "GOTO".
- Introduced new parameters for HPTDC8 (85 (SSEEnable), 86 (MMXEnable), 87 (DMAEnable) and 79 (GroupTimeout)).
- HPTDC DAQ - internally used _TDC array - modified value for MAXIMUMMAXIMUM_NUMBER_OF_HITS changed from 16 to 500
- HPTDC DAQ - will now continue processing the event even if one hitcounter exceeds MAXIMUMMAXIMUM_NUMBER_OF_HITS
- HPTDC DAQ - call of manager->Start() moved from DAQInitialize to DAQOnlineReadEvent function
- HPTDC DAQ - modified event file structure (see source code [added EventCounter and modified data type for hit-array, hit-data still signed long])
- CoboldPC Main Program - EventData array is cleared (filled with 0.0) before call of ReadEvent routines. The behavior can be controlled by preferences flag.
- CoboldPC Main Program need now three new functions/definitions in DAQ dll modules
CTime *DAQGetLastStartTime();
void DAQSetLMHeaderVersion(unsigned __int32 _ui32LMHeaderVersion);
__declspec(dllexport) unsigned __int32 DAQTimeStamp;
- In DAQ dll Modules data types like int or short are replaced by more accurate types like __int32 or __int16
- HPTDC_Daq - modified Timestamp information. DAQ uses now the more accurate HPTDC internal timestamp.
- Exit command will now also be processed inside a CoboldCommandFile (CCF)
- GaussSmooth, ContourSmooth, BinomialSmooth and MeanSmooth will accept now only one parameter. If only one parameter is provided the function will work on the actual displayed spectrum.
- Order of elements in dialog box for GaussSmooth, ContourSmooth, BinomialSmooth and MeanSmooth changed
- Definition of spectra changed to support coordinate sets as coordinate. (see coordinate sets)
- StandardDAn modified. New coordinates added (False, Const1 .. Const8)
- HPTDC_Daq - modified function to detect "multiple boards needed".
- Coordinates for hdr1 and hdr2 (in the new USERDEF format) and TimeStamp will be handled separately from the EventData array. All references to number of coordinates in LMFs will be corrected.
To perform the new hdr1/2 and TimeStamp handling a new class has been introduced to DAn and DAQ modules with the name of LMFPreEventData. A reference to this class is available in the online/offline read event routines of the DAQ modules and to the AnalysisProcessEvent of the standard DAn modules. Old CoboldPC 2002 LMFs (other than USERDEF) can still be read!

Error Correction

- Calling "execute" command and then cancel the dialog box will bring the execute dialog box again (Corrected)
- Corrected Multiview definition routine (command line input)

Update Information CoboldPC 2008 R2 9.1.909.1

New Functions

- Changed Document-Format (dcf, dcfz and dcfid). CoboldPC can read old files but will write new ones (9.1.909.1 version).
- New functions in UserSP.dll available.
 - GetCoordinateIndexByName(Name) returning the Index
 - GetCoordinateNameByIndex(Index) returning the Name
 the index must be in the range 0 < Index < MaxIndex
 An empty String is returned when Index is out of range.
- New Dll (including source code) available to call a self written function from CoboldPC command line. The Dll implements a DoFunction call that can be called using the call command in CoboldPC command line.
 void DoFunction(CString csInput[], int csInputLength);
 See the UserFCall Project.
 In CoboldPC the DoFunction is called by the command "CallMyFunction".
 CallMyFunction DllPathName,String
 DllPathName is the path and name of the UserFCall.dll
 String is the string transferred to DoFunction
 DllPathName can be empty. If the DllPathName is empty it is assumed that an earlier CallMyFunction has provided a valid UserFCall.dll
 If the DllPathName is given then CoboldPC tries to unbind an earlier call to that dll and rebinds the new one. The UserFCall.dll stays loaded till the next CallMyFunction command with a valid DllPathName.
- GaussFit routine changed to a routine described by Mukoyama, NIM 125 (1975) 289-291 to get better results for a broad variety of gauss curves. Fit marker for gaussfit now showing the given range and in inside really used range.
- 2TDC8PCI2 DAQ module available on request.
- LMF written by DAn (Standard and Empty) and DAQ (HM1, TDC8, TDC8PCI and TDC8HP) modules can now contain their source codes as well as the CCF history till the point of creating that LMF. From these modules also the DCF(x) files can contain the DAn code. This behaviour can be controlled by the Preferences section.
- For source code integration in LMF the Main header has been changed. See section ListModeFile data format section.
- Whitespaces are now removed from the command line before processing the command (this included also leading or trailing tabs).
- try... it is now possible for some commands to have the prefix "try " (including the space). Commands with try are executed but in case of an error there is no warning and the command execution continues with the next possible command.

Modified behaviour

- [Shift](#) Function has been extended to support fine shifts inside spectra. Now you can shift a spectrum also by 10% using CTRL+ALT+Curor Keys
- coordinate, parameter and weighparameter definition are changed to implement comment functionality. The appropriate list boxes are changed accordingly. No CCF changes necessary
- HPTDC DAQ - new parameters (TDCDriverVersion and NumberOfDAQLoops) implemented (52 and 53, see Standard-Parameters.ccf)
- HPTDC DAQ - driver version test implemented and moved to 3.0.1 for HPTDC_DRIVER_3.0.1.dll
- UserFCall.dll now needs UserSP.dll. LIB and H files updated.

- UserSP.dll has new function calls. Please refer to UserSP.h for a full list of function calls.
- Using command xmin, xmax, ymin, ymax, zmin and zmax uses now internally used the spectrum name instead of title to get the index of the spectrum for the definition.
- Cobold.dll has new function calls. (V9.1.909.1 -> ExecuteCommand, GetDAQDAnParameters, GetDAQDAnWeighParameters, GetDAQDAnEventData)
-

Error Correction

- Projecting with automatically defining the projectTo spectrum during DAQ-running mode at high rate now working.
- most call to MFC in UserFCall.dll source (cross thread MFC problematic) no longer results in crash.
- Command line no longer accepting commands when new command is given during DAQ-running mode solved.
- HPTDC DAQ - SyncValidationChannel now correctly transferred to HPTDC.
- "show spectrum" sizeof YBin now applied correctly (also in DefaultSize).
- LMF files for pre V9 are now read correctly.

CoboldPC 2011 R1 10.1.1109.1

New Functions

- Sleep implemented as command. This command freezes CoboldPC completely. s (seconds), ms (milli seconds) and m (minutes) must be used as units.
- New functions in UserSP (for definition see UserSP.h file in the UserSP solution)
 - __int64 GetDisplayedSpectrumNumber();
 - CString GetDisplayedSpectrumName();
 - __int32 GetSpectrumType()
 - CString GetSpectrumName()
 - void SetGaussFitValues()
 - void SetMarker()
- Full screen mode implemented (but not command bar available!)
- Modified Flag (that controls Save Button active) always set during LM Event Read (Online or Offline)
- ForceLMFFlushAfterEvent command. (This slows the eventloop but it should be used only on low event rates anyway)
- Copy Graphic (screen mode) menu command and CopyGraphicScreen command will use screen colors for copy
- CoboldParser DLL introduced for mathematical ParseMathCommand command
- DrawSine.ccf and DrawMandelbrot.ccf introduced to demonstrate new ParseMathCommand command of CoboldPC
- Splash screen display flag in Preferences

Modified behaviour

- CommandLine behaviour improved during automatic update
- CTRL-C (copy Graphics) changed to CTRL+ALT+C, now CTRL-C can be used for command line -copy-
- CTRL-A (copy ASCII) changed to CTRL+ALT+A, now CTRL-A can be used for command line -select all-
- Functions in UserSP
 - GetXYMarker replaced by GetXYCursor
 - GetXYBoxMarker replaced by GetXYBoxCursor
- Marker command changed to allow box marker information (box marker added to marker command)
- show marker box modified to show also new box marker information. copy and print adapted accordingly.
- CHECKID_DAQDANCCF splitted into 2 definitions. CHECKID_DAQCCF (stored in parameter 50) and CHECKID_DANCCF (stored in parameter 1050)
The value for both is now 201102080000
- DAQ Modules
 - LMF_VERSION changed to 10
 - DAQ_VERSION changed to 10
 - TDC8HP
 - Check for Firmware >= 6 implemented
 - if Firmware 6 is detected error messages is displayed when DMAEnable = false
 - creation of tdcManager is now verified and tested if it has been created correctly, if not an error message is displayed
 - Driver version changed to 3.4.2 (Both driver dlls (managed and unmanaged) must be available to CoboldPC)
 - Driver support now 3 card mode
 - Parameters for DAn modules now start from 0-999 instead of 0-99. (parameters 0-999 are reserved for RoentDek DAQ modules!)
 - Changed function names to be more accurate to functionality
 - DAQOnlineInitialize is now DAQLMFWriteUserHeader
 - DAQOfflineInitialize is now DAQOfflineInitialize
- DAn Modules
 - Standard DAn
 - Parameters for DAn Modules now start from 1000 instead of 100 (parameters 1000-2999 are reserved for RoentDek DAn modules!)
- OMP support removed from projects because it is no longer used
- calibration is now modifying the marker list accordingly
- Cobold.dll is renamed to CoboldUtilities_x86.dll or CoboldUtilities_x64.dll
- auto file mode is renamed to multi file mode. it is no longer connected to the "status box" i.e. for abort multi file mode (this is now done by "stop m" command).
- multi file mode is completely redesigned and now able not only to write multi files but also to read them with extra execution of "controlling ccf files" (see "new" command for further information)
- "stop" command may have an new parameter to stop multi file mode (see "stop" command for further information)
- "show rate" and "show status" boxes are now running in separate threads. therefore they are no longer blocking command execution. (This may affect existing ccf files. i.e. new.start...show stat... new.start... ..)
- Menus und Toolbars changed to the "Outlook 2007 black" style so Menus and Toolbars may now be customized
- Gaussfit modified to be more accurate
- Profile information no longer stored in the users registry. Instead a *.Config.xml file is used and normally located in the applications folder. If the source location is not writable then two other places will be used. the users roaming application folder or the users temp folder.
- in LOG scaling the automatic minimum value is changed from 0.1 to 0.7 to use the drawing area more effectively
- "show rate" box is now saving it's window position
- default value of project box modified to "auto mode"
- project box - "To Spectrum" (if available) is set to -1. If you do not change that value an error is displayed
- project box - "To Spectrum" and "Autodefine Spectrum" now as radio button
- initial view of the "update button" removed from toolbar (but can be added using toolbar configuration)

- View Menu filled with new entries like "Next Spectrum", "Previous Spectrum", "Rescale", "Update", "UpdateAll", "Remove Update", "X-Log", "Y-Log", "Z-Log", "Full Screen" and "Clear All Spectra"
- new command redesigned due to new functionality of multi file mode
- NextSpectrum and PreviousSpectrum command no longer added to command history. Old documents are filtered during read.
- Draw axis routine modified to avoid displays like xxxe-17 instead of 0 in axis plotting
- Copy Graphic menu command changed to Copy Graphic (print mode)
- CopyGraphic command changed to CopyGraphicPrint
- Command line is now trimmed with " " (space character) and tabs at start and end of line (or till ";")
- modified error handling on UserCommand
- Startup CCFs modified to make use of the new ParseMathCommand as well as the modified call command
- StatusBar of CoboldPC modified to display other values like EST, progress, events and eventrate
- StatusBar no longer displays help information in first pane
- Reading LMF files now read only to MaxEvents, no longer to EOF (also in multi file)
- FileSelector Box for Windows 7 now with active "Comment" and "CIF" button
- _CONFPLATRD_ added to RoentDek projects to avoid mixing of x64 and x86 elements. stdafx.h changed accordingly to throw error on wrong configuration/platform settings
- plotting axis information changed to show the 0 in linear scale of a spectrum if available

Error Correction

- loss of marker in 2D spectra while using smooth command fixed.
- assertion error on restore property on with empty recent document list fixed.
- execute for "unknown command" reactivated. so if "execute" fails "callmyfunction" is called and if that one also fails then try executing ccfc or dll from macros subfolder located in the application path of CoboldPC
- load/save an DCFZ or DCFD file (Zipped/Deflated Cobold Document File) memory allocation problem solved by fallback to standard DCF (Cobold Document File) store.
- major memory leaks removed. some minor leaks still existing but they are not affecting runtime stability over long CoboldPC runtimes
- spectrum serialize (load or save spectra information) read error due to memory allocation error for DCFZ/DCFD Files will abort the load of that File.
- document save with "no loaded" DAn source code information fixed
- condition redefinition error while in Online/Offline mode solved
- UserFCall error corrected when call with lines like this one

$$\text{UserFCall, math, p113} = \text{p110} / 4.;$$
- XP New File Dialog Information Block error solved
- show marker 1 dim display 3 column corrected
- show marker print button is now working
- commandline execution blocked when exiting "show rate" or "show status" box via "CloseBox" button fixed
- Page numbering in Print List of Parameters, Spectra etc. corrected
- command bar length of 1 character after CoboldPC closed minimized
- EventCountReset flag in DAn now working
- update command like "update 1s,4d" now deletes also in multispectrum mode

CoboldPC 2011 R2 10.1.1203.1

New Functions

- New functions in UserSP

$$\text{void GetSpectrumDataInformation}(_\text{int32 iSpNr, double *pdDataPointer, unsigned long \&ulDimX, unsigned long \&ulDimY});$$
- F5 now calls CoboldPC "UPDATE" function.

Modified behaviour

- HPTDC Driver changed from 3.4.2 to 3.4.3
- Tab order in dialog boxes changed for some dialog boxes to improve jump order in edit fields.

Error Correction

- DAQ Module for HPTDC - Error correction in Low Res calculation.
- DAQ/DAn Modules -CDoubleArray linker error solved for DEBUG und ASSERTVALID function.
- DAn/DAQ SourceCodeInformation of CDAn.cpp and CDAQ.cpp fixed
- DAn deletion of SourceCodeInformation in ExitInstance fixed for DEBUG
- Marker numbering in spectrum corrected, starting now with 0. Remove marker now removes the right markers.
- DAn Standard Module error value changed from -1.200 to -1.e200

CoboldPC 2011 R3 10.1.1203.1

New Functions

- New DAQ Modules
 - HMI
 - TDC8PCI2
- multifile DAQ
 - new blue PAUSE icon introduced
- UserSP
 - new function
 - ClearSpectrum(...)
 - GetMultiFileInformation(...)
 - SignalStopMultiFile()
 - SignalStop()
- Commands:
 - ExportCommandHistory

- UserFCall
 - new function "locate"
- new Startup File created during CoboldPC start. If CoboldPC has problems during startup the file remains and gives options on next start.

Modified behaviour

- multifile DAq online and offline
 - file detection in NewBox for nnnnn.lmf at the end for Online/Offline CCF and LMF files
 - the first found file in the set is now added to the LMF History and LMFPPath
 - no additional files are added to the LMF History and LMFPPath
 - start number = stop number check implemented (giving a warning)
- paths to CommandFiles, Documents, LMFs are written to the config.xml as soon as they reach "add to recent list"
- Show Status box stays open as long as LMF or multi file LMF is in progress
- StartupOnline.ccf and StartupOffline.ccf now again only Startup.ccf
- CallMyFunc call gain listed into commandhistory as well as execute and macro execution
- DAq Module
 - HPTDC
 - ExternalClock Test on TDCnumber > 1 implemented (is now asking what to do)
- DAn Module
- On first start of CoboldPC (XML file FirstCall=true) the DAq module for HPTDC and the DAn module for standard analysis are preloaded without asking
- UserFCall
 - subfunction "gauss" now also callable as "gaus"
- "Show Parameter" box modified to display two new buttons: 1. ViewDAq (0) jumps to parameter 0, 2. View DAn (1000) jumps to parameter 1000
- EventData allocation occurs now on each coordinate definition and not during new command execution

Error Correction

- multifile DAq online and offline
 - error on "new" and "stop multifile" without given "start" fixed
 - offline new command can now be used
- exit command results in crash fixed
- UserFCall
 - error corrections of binsize testing
- NOINSERT in fileread now working
- in NEW box FileRead: the comment of the browsed file is displayed again in single and multifile mode
- crash of CoboldPC on existing HIDInfo.dat.bck file fixed
- overwrite file protection implemented to single and multi file mode
- termination error on open Status Box or Ratemeter Box fixed
- WeighParameter 0 error in spectrum definition
- DAn Module
 - function GetDeltaEventTime and GetEventTime index parameter fixed
- CoboldParser.dll MakeLower problem solved
- MakeUpper in Execute/call/macro section fixed
- Update, UpdateAll behaviour corrected
- Parameter array correction for document version <= DOCUMENT_VERSION_9_1_811_1
- missing calibrate inserted into help index
- SetAxisText command now accepting 0 as spectrum input
- comment shift in show parameters corrected
- del spectrum,# -> spectrum number in spectra after # corrected
- wrong help text for calibrate corrected
- SetAxisText now works also on calibrated spectra