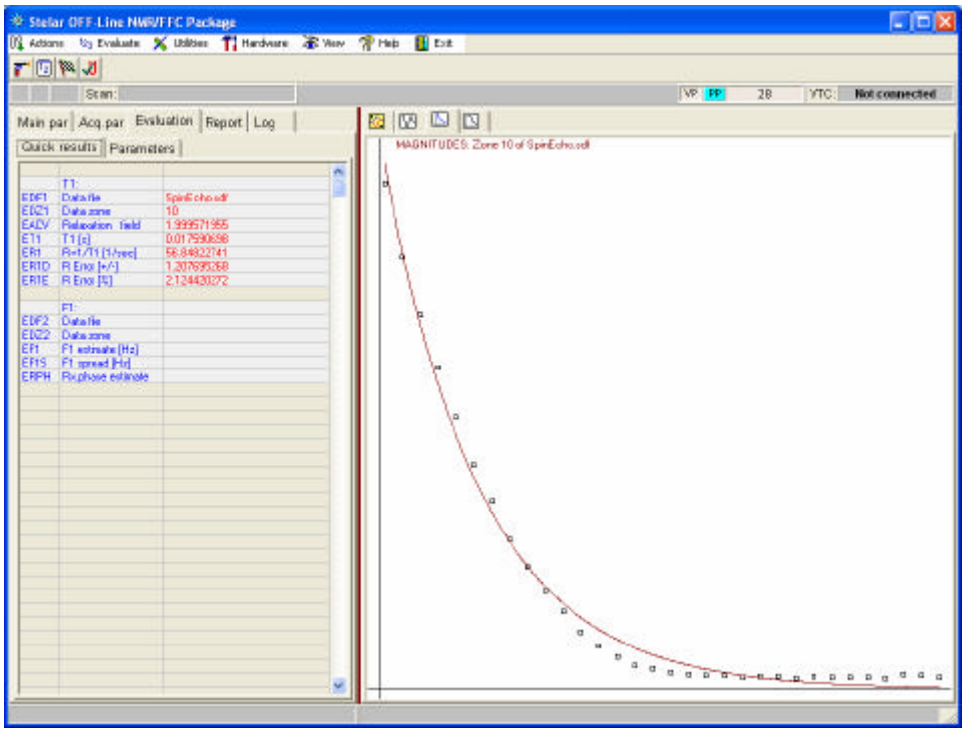
# AcqNMR

# Software manual
# v. 2.1.79



## 4.2 Software Review

### 4.2.1 Main Window

The main program window contains the following components:

- Main menu
- Immediate mode buttons
- Scans counter
- Top-macro panel
- Multi-page control
  - Main parameters sheet
  - Acquisition parameters sheet
  - Evaluation parameters sheet
  - Reports sheet
- Overlapped display screens
  - XY monitor screen
  - Data Display (DisDat) Screen
  - Multi-Block graph
  - User graph
  - System log
  - Parameter option
- Status bar

#### 4.2.1.1 Main Menu

The main menu contains a number of items, each of which opens a pull-down window with a submenu. To select a main menu item, either click on it or press the **Alt** key together with the letter (underlined), which identifies the menu.

List of main menu items:

- Tools
- Evaluate
- Hardware
- Configuration
- View
- Help
- Exit

Tools:

The Tools entry of the main menu contains these submenus (*) and commands (-):

* Load parameters …. to load parameters from a file
- Save parameters …. to save current parameters in a parameter file
- Clear modified flags …. to remove the modified-parameter flags and display enhancements
- List pulse sequence …. to print the detailed list of pulser channel settings in current sequence.
- Execute macro …. to run a macro command

Load parameters:

Full sets of program parameters can be loaded from two different sources: parameter files or data files. The *Load parameters* submenu (part of the main menu *Tools*) therefore contains two items:

- from a Parameter file

To load a complete set of system parameters from a previously saved parameters file, use this submenu and select the option 'from a parameters file'. When a file open dialog appears, select an existing parameters' file (default extension .par) and press the Open button to load the set of parameters it contains.
Not all parameters are loaded. Some parameters, such as those describing the instrument's hardware configuration, may not be modified at all. Generally, only those parameters which are User-accessible are loaded. Consequently, different users may exchange parameter files among themselves even though their instrument may have incompatible configurations.

- from a Data file

To load a complete set of system parameters from a previously saved data file zone, use this submenu and select the option 'from a Data file'. When a Data source dialog appears, select a zone of an existing data file (default extension .sdf) and press the Open button to load the set of parameters it contains.

The criteria for parameters loading are the same as in the case of 'loading parameters from a parameters file'.

For more information about the parameters used, see the separate topic on Parameters

*List Pulse Sequence*

prints-out the detailed list of pulser channel settings in current sequence. This is rather technical information and may be of interest to you if you develop and test novel pulse sequence. It is also helpful whenever you have any doubts about the details of the Stelar implementation of any of the pre-programmed experiment.

*Execution of a Macro command*

This command of the main menu permits to execute a macro. The system prompts the operator for the input of a macro command name and proceeds further based on the commands written in that particular Macro.

Evaluate:

The Evaluate entry of the main menu contains the following commands:
- Evaluate data …. On-line data evaluation
- Correct F1 using last data …. Used for fast offset correction
- Correct T1MX using last data …. Used for fast maximum T1 correction

Hardware:

The Hardware entry of the main menu contains these commands (-) and submenus (*):
- Go and adjust …. Repeat the experiment while adjusting parameters
- Update …. Forced update of hardware interfaces
- Boot and Update …. Reinitialize AQM and update hardware interfaces
- Tune probe …. Probe-tuning procedure
- Calibrate magnet offset …. Trims magnet current offset (null value)
- Simple scope …. Use the ADC's as simple scope inputs
- Magnet On …. To switch on the magnet
- Magnet Off …. To switch off the magnet
- Printer setup …. Standard windows printer setup protocol
* Diagnostics …. Run a diagnostics form (Window)

Configuration:

The configuration entry of the main menu contains this submenu:

*System log features*:

To trace the hardware diagnostics activity

- Trace ON …. Write detailed output to the Log (full trace of all activities)
- Trace OFF …. Standard Log output only

*Edit system files*:

To edit the default files such as *.def, defaultmacro, defaultsequences

View:

The View entry of the main menu contains these commands:

- XY monitor …. Brings to front the XY monitor graphic sheet
- DisDat window …. Brings to front the Display Data graphic sheet
- Multi-block window …. Brings to front the multi-block graphic sheet
- User graph …. Brings to front the user defined graphic sheet
- System log …. Brings to front the System Log sheet
- Parameters options …. Shows the parameter options (full width parameters display)
- Hide options …. Hides the parameter options

Run:

To run any other external applications like, calculator, notepad etc.

Help:

The Help entry of the main menu contains these commands (-):

- Contents …. Opens the run-time manual at the Contents page
- Index …. Displays the keyword search form of the run-time manual
- About …. Shows basic features of the program (version, copyright)

Exit:

This command is used to quit the NMR32 program.

#### 4.2.1.2 Immediate mode buttons

The immediate mode buttons are located in the upper left part of the main window, just under the window's main menu. They are:

- Button **Go**

Starts data acquisition.
If another acquisition is in progress it generates a warning and does nothing. Instead, it stops a non-critical activity (go and adjust, digital scope) which is underway and then Goes.
Before starting an acquisition, the system tests the current status of the interface and automatically executes the following actions:
- If the Spinmaster console interface (CI) is not switched ON then a proper error message is issued and nothing more happens (the same applies to the cases where an attempt to communicate with the CI fails).
- If the CI is on line but not booted (its programs were as yet not loaded), the CI booting routine is executed before proceeding further. This is completely automatic (just wait for few seconds).

- If the CI is booted but not updated (the current values of the hardware-related parameters were so far not transmitted to it), the update routine is executed automatically. This may take few seconds depending upon the hardware which must be set-up.
The acquisition can actually start only when the CI is updated; then it runs "in background" meaning that, most of the system commands (e.g those of the evaluation menu) may execute while acquisition is in progress.
On the other hand, one cannot
-Modify an essential acquisition parameter, such parameters are said to be "locked" by the active background process.
-Load a parameters' file
-Load parameters from a pre-existent data file
-Issue a new "Go" or "Go and Adjust" command which is ignored without warning. (You may of course interrupt the acquisition process by the specific "Quit" command. Another way, which always works, but which we do not recommend at all, is by re-booting the system –an action resulting in loosing the acquired data.)
The following actions cause loss of data of an acquisition in progress:
-Switch off either the Host or the Console Interface
-Reset the Host by pushing the RESET button or by pressing the Ctrl-Alt-Del combination.
-Resetting the Host via software from another program (this monstrous way of terminating execution is not as rare as it should be -–beware especially of game programs, test them first).
-Running programs which do such fine things as canceling the Host memory or Disk.
-Beware also of resident programs, they might be in conflict with AcqNMR. Try them first extensively and if you have a problem, contact us.

- Button **Quit**
Waits until current scan is completed and then stops acquisition.
The acquisition normally stops once the pre-set maximum number of scans (MS) has been reached. In some cases, however, the user may want to interrupt the acquisition manually. This is particularly true (in fact unavoidable) when MS has been set to large value. The manual termination can be achieved by pressing Quit button.
The system interrupts acquisition immediately upon reception of the Quit command. If there is a scan in progress at the moment, it waits until the current scan is completed and then stops acquisition. Remember that in multi-block experiments, a scan consists of one FID in each block.

- Button **Cont**
Reserved for a future implementation of the continuation

- Button **Save**
Saves the acquired data to disk. This command initializes the transfer of acquired data from the Console Interface into a disk file on the Host computer. It appends the newly acquired data as a new data zone of the current Data Acquisition File (data FILE). If FILE value is void, displays a file selection dialog and allows the user to specify a data file (either a new one or an old one).

- Button **Zap**
Discards the last acquired data zone of the data acquisition file.
It happens quite often, especially when setting up the instrument, that the last acquired data zone of the Data Acquisition File is of no more use and may be discarded. This button provides a fast way to do so.

- Button **Kill**
Stops the data acquisition without completing scan and saving data.

#### 4.2.1.3 Scans Counter

The scans counter panel is located in the upper part of the main window. It displays the number of scans completed and block number in multi-block experiments. When dummy scans (DS) are performed, they are displayed as negative integers.

The scans counter is updated about once a second. When the actual scanning is faster this will cause the counter to apparently jump over blocks or even whole scans. This is normal and does not imply any loss of data. Remember that the data acquisition and accumulation is carried out by the AQM under stringent pulser control and timing. The PC is simply periodically interrogating the AQM about the current status of the acquisition. When the PC is real busy or holding on for some reason, the scans display may even stop altogether but the scanning and data collecting nevertheless proceed normally. As soon as the PC is free again, the counter (as well as data display) gets updated.

#### 4.2.1.4 Top-Macro panel

The top-macro panel is located in the upper part of the main window, at the side of the scans counter. It is visible only while a macro command is in execution (see macro command mode), in which case it bears the name of the command on reddish background. When a macro calls another one, the text in the panel does not change (that's why it is called top-macro panel).

In the immediate execution mode, the top-macro panel is not visible.

#### 4.2.1.5 Multi-page control

The main multi-page control is a virtual display device bearing several tabbed sheets (pages) among which the User can choose by clicking the desired tab. The following sheets are present:

- Main parameters sheet
- Acquisition parameters sheet
- Evaluation parameters sheet
- Reports sheet

#### 4.2.1.6 Overlapped display screens

These are rectangular areas (screens) in the right part of the main window. Since they are overlapped, only one screen can be seen at a time. The User can select a screen either by clicking the proper item in the View submenu or by clicking the right mouse button on the currently displayed screen - an action which pops-up a fast menu which partially duplicates (and complements) the View submenu of the main menu.

The following screens are available:

- XY monitor screen
- DisDat window
- Multi-Block window
- User graph
- System log
- Parameter option

#### 4.2.1.7 Status bar

The status bar is located at the bottom of the main program window. It is divided into two single-line windows. The left window displays short hints as the User moves the cursor over various controls and menu items.

The right window displays the X and Y coordinates of the mouse cursor when it moves across the XY monitor screen on which there are displayed data. The values are in real-world units (e.g., X is in milliseconds, when an FID is displayed. In the case of multi-block data, the block number is also shown.

### 4.2.2 System Parameters

#### 4.2.2.1 Parameters review

Parameters are used to configure the whole system, set-up the hardware, define the sequence of events during data acquisition, annotate the acquired data, configure some of the data evaluation procedures, display evaluation results, etc., etc. In a certain sense, every statement that can be made about the state of the system and/or the acquired data amounts to specifying the value of some parameter.

The program manages over 200 parameters, of which about one half are displayed. Roughly half again of the latter can be edited by the User while the remaining ones are not accessible but display useful information.

The parameters, together with their attributes (flags, default values and options) are defined in the parameters definition file Parameters.def. This is an installation-specific plain-text file. Extreme caution must be exercised while editing this file since certain modifications are potentially detrimental to the instrument hardware. Users should never do so themselves.

The basic components of a system parameter are:

Name: A long, descriptive name of the parameter

Acronym: A brief name (at most four characters) used to refer to the parameter in situations where long, descriptive names are not acceptable (pulse sequence scripts, macro commands scripts, data files parameter previews, etc.)

Value: Current value of the parameter. This may be a number or a string. Moreover, some numeric values may be defined indirectly by means of an arithmetic expression involving other parameters as arguments

Comment or Options: This is usually a comment string containing a brief description of the parameter. In some cases, however, the string is defined at run time by the system and/or by the User and affects the operation of the instrument

#### 4.2.2.2 Categories of parameters

There are many parameter attributes according to which parameters can be classified. A typical User, however, needs to distinguish only the following categories (they are not mutually exclusive - a parameter may belong to more than one category):

- Accessible parameters
- Inaccessible parameters
- Hidden parameters
- Pulser interval parameters
- File parameters

Accessible Parameters

The value of an accessible parameter can be edited at run time.

When User clicks an accessible parameter's name, its background changes color to light blue - this is the parameter cursor. Each parameter page has its own parameter cursor which is lighted when the page has focus. The cursor can be also moved up and down by means of the up- and down-arrow keys. As it moves, however, it jump over parameters which are inaccessible or hidden.

When a User-accessible parameter is marked by the cursor, pressing the Enter key opens a value-editing box in the parameter's value field (the same effect is achieved by a single mouse click on the value field).

Some parameters (in particular the RF pulses PW,P1,P2,P3,... and the delays D1,D2,D3,...) may be accessible/not accessible depending upon the experiment or context.

The fact that a parameter is accessible and its value can be edited does not mean that the same applies to its options. The two concepts, in fact, have nothing in common (see pulser interval parameters).

Inaccessible parameters

The values of some inaccessible parameters are displayed but cannot be edited. Inaccessible parameters essentially just convey information.
The background color of the name and value fields of an inaccessible parameter is very light gray. Clicking on any of these fields has no effects.

The inaccessibility of some parameters (in particular PW, P1, P2, P3, …. and delays D1, D2, D3, …) depends upon the experiment (pulse sequence) loaded. In principle, inaccessibility is context-dependent.
It may happen that a parameter is inaccessible and therefore its value cannot be edited, but this does not apply to its options. In particular, the options of all pulser interval parameters, which are not hidden, can be always edited.

Hidden parameters

A hidden parameter has a display location but is not used at all in the current context (experiment, pulse sequence). Only its acronym is shown but none of its other attributes. The background color of its field is dark gray and no field is accessible for editing.

Pulser interval parameters

A pulser interval parameter defines the duration of an interval during the execution of an experiment (pulse sequence). It therefore represents an input datum for pulser programming.

Pulser interval parameters therefore include:
- all RF Pulses (PW, P1 to P16),
- all Delays (D0 to D16),
- Receiver Inhibit interval (RINH)
- Acquisition delay (ACQD)
- Sweep time (STIM)

- End-of-sweep accumulation directives (ENDS). This is a pseudo-interval of zero duration whose options define what is known as the receiver phase cycle.

The pulser interval parameters can also be easily recognized by the fact that, when not hidden, their options field has a white background color and can be edited. The options of such parameters in fact determine which pulser channels are active as well as what is the observed channel RF phase and transmitter attenuation.

To edit a pulser interval parameter, select it with the parameters cursor (see accessible parameters) and press 'O'. To edit all options, use the '*View|Parameter* options' menu and click the field of the desired parameter. This, in fact, is the only way to edit the options when the parameter's value is inaccessible.

File parameters

Parameters like FILE, EDF1, EDF2 denote data files. For such parameters, the value field contains the 'clean' file name (without path) while the path to the file's directory is written by the system into the parameter's options field. This is only system-supplied information; the options field of these parameters is not directly editable.

#### 4.2.2.3 Editing of Parameters

To edit an accessible parameter, click its value field or press “Enter” while the parameter cursor is positioned at the parameter's name. As a result, a combo box appears over the parameter's value field and a new value may be entered.

The values listed in the pull-down window of the combo box depend upon the type of parameter:

- for parameters which have a limited set of allowed values, a list in the pull down menu assists in the selection. If the parameter is a string (e.g. PHCY), the input value must be one of the list, otherwise the input is not accepted. If the parameter is of a numeric type (e.g., FLTR) the admissible value closest to the one entered is selected.
- for parameters, which are not limited to a predefined set of values, the pull-down window shows the editing history of the parameter, i.e., a certain number of values, which were input recently by the User. This is helpful since it often happens that one wants to return to a previously used value.

Many numeric parameters are limited to either integer values and/or to an allowed range of values. For such parameter, whatever value is input, it is always checked and, if illegal, the nearest allowed value is automatically chosen.

#### 4.2.2.4 Values of parameters

The parameter value is a string which, depending upon the particular parameter, is interpreted either as an alphanumeric string or as a number. The value may be subject to editing constraints (see parameters editing), which apply also when the parameter is set from a macro command.

The length of RF pulses (**PW90**, **P1** to **P16**) are expressed in degrees of the nutation angle and converted into times before downloading to the pulser using the current 90 degrees pulse width (**PW90**) as conversion factor. If **PW90** is 5 μs and **P1** is set to 180° degree, its length is 10 μs.

The pulser interval parameters represent a special case. Their values are inherently numeric but, unlike other parameters, they admit an indirect definition by means of an arithmetic formula. For example, D2 may be set to (2***D1**) to keep **D2** twice as long as **D1**.

The formula is evaluated during virtual pulser programming (virtual pulser is a software image of the real pulser). Once the actual value is known, the system displays it in the parameter value field by appending the string "=<value>" to the formula. In our example, if D1 is 0.3, the value field of D2 would eventually show "(2*D1) = 0.6". The following rules apply to parameter value formulae:
1. An expression must be enclosed in parenthesis to inform the program that a “calculation” should be needed. The opening parenthesis is what tells the computer that there is a formula rather than a simple value.
2. All four arithmetic operations may be used.
3. Parentheses may be used as usual, provided they are properly closed.
4. There is no limit to the complexity of the formula.
5. Arguments for arithmetic operations may be:
   a) numeric constants (expressed in any numeric format)
   b) acronyms of parameters admitted in formulas (see below).

The parameters admitted in formulas include:
- All pulser interval parameters, except the RF.
- A selection of auxiliary parameters such as **T1MX**, **TPOL**, **RD**, **TAU** and **EDLY**.

If the specified parameter is not allowed in the expression, the program just issue a proper warning.

#### 4.2.2.5 Parameters interdependence

Some parameters, such as the User notes **N1**,**N2**,**N3**, are stand-alones in the sense that they do not interact with others. Most of the core NMR parameters, however, are inter-dependent to a considerable degree. Thus, for example, when a User changes **SW** (the sweep width) then the system:
a) calculates the corresponding Nyquist frequency (taking into account current value of **FTM**),
b) sets the corresponding dwell time **DW**,
c) considering the current value of block size (**BS**), calculates and sets the sweep time (**STIM**),
d) when autofilter parameter (**AFLT**) is set, uses DW to calculate and set the proper filter (**FLTR**).

Such local chains of interdependencies (some with loops) are a rule rather than an exception and their complete map is of considerable complexity. Fortunately, they operate in a completely automatic way and, in most cases, in accordance with User's intuition.

#### 4.2.2.6 Pulser interval parameter options

Every pulser interval parameter has, apart from its value, a User-accessible field in which it is possible to specify a number of additional, mostly hardware oriented properties called options. In particular, they make it possible to define/control:

- Transmitter RF phase.
- Transmitter RF attenuation.
- Generic pulser channels switch
- Accumulation directive(s) known as receiver phase

- In the case of Fast-Field-Cycling, one of four preset magnetic field level.
- In HR-NMR, the decoupler state.
- States of any pulser-controlled device (X-device states).

At run-time, the options are normally hidden from view by one of the graphic windows. In order to gain access to them, use the *View|Parameter* options main menu command or the Show options command present in many of the popup menus (alternatively, press Alt-O). The reverse is achieved by the *View|Hide* options main menu command or by the Hide options command of popup menus (alternatively, press Ctrl-O).
Once displayed, the options can be edited in a way similar to editing of parameter values - just click on the current option value or, if you don't like mouse, press "O" and a combo box shall appear, allowing you to input a new value. The input string may encode any number of elementary options. In order to be accepted, however, it must respect a precise syntax.

The syntax of parameter options supports what was once known as phase cycling. In its present form, this concept goes much further, allowing acquisition cycles in which, during consecutive sweeps (or scans), various hardware devices (not just RF phase selectors) run cyclically through pre-defined sets of states. Taking full advantage of this powerful and exciting feature is very easy and does not require any modification of the program's executable.

(i) Transmitter RF phase
As far as the software is concerned, the RF phase controller is an intrinsic pulser-controlled device which may appear in pulser interval parameters options with the pre-defined device identifier p. As such, it may take part in data acquisition phase cycling.

An argument of the p-device option specification may be any real number between –360 and +360. Its value denotes the RF phase in degrees which (on Stelar instruments, the RF phase is digitally controlled with a resolution better than 1 degree). Alternatively, one can use the following symbols for standard RF phase settings:

| | |
|---|---|
| x | 0 degrees |
| y | 90 degrees |
| –x | 180 degrees |
| –y | -90 degrees |

Example of a valid phase-cycle specification: p(x,–x,45,–45,y,–y).
Clearly, the transmitter RF phase is irrelevant when there is no pulse. Consequently, it is usually specified only for the PW and P1, P2, ... parameters, though it is not formally illegal to specify it in any pulser interval parameter. Such a specification, however, has no practical effect since the RF phase is set to the receiver phase value (parameter RPHS) whenever the transmitter is gated OFF.

When the phase option is omitted, the default value of 0 degrees is assumed.

Exploiting the p-device option, one can easily program the so-called composite pulses with almost arbitrary in-pulse phase variations.

(ii) Transmitter RF attenuation

As far as the software is concerned, the RF attenuation controller is an intrinsic pulser-controlled device which may appear in pulser interval parameters options with the pre-defined device identifier "a". As such, it may take part in data acquisition phase cycling.

The a-device option arguments may be integers in the range from –63 to 63 (do not use the + sign!). The values indicate relative observed transmitter output attenuation in dB.
Example of a valid RF attenuation specification for P1 (attenuated always by 6 dB): a(6)
Example of another attenuation specification for P1 (pulse is alternately On and Off): a(0,63)

Clearly, the transmitter attenuation is irrelevant when there is no pulse. Consequently, it is usually specified only for the PW and P1, P2, ... parameters, though it is not formally illegal to specify it in any pulser interval parameter. Such a specification, however, has no practical effect since the attenuator is set to 63 dB (maximum suppression) whenever the transmitter is gated OFF. This helps the transmitter gate to suppress both RF and noise at any time except during a pulse.

The actual attenuator value set in the hardware is the sum of the a-option specification and the value of the parameter TATT (base transmitter attenuation) which thus represents a pre-defined null level. The resulting sum must be within the range 0 - 63 dB. When negative, it is automatically reset to 0; when greater than 63, it is automatically set to 63. For example, when TATT = 15 then a(–3) evaluates to 12 but a(–18) evaluates to 0 and, likewise, a(3) evaluates to 18 but a(63) evaluates to 63.

Exploiting the a-device option, one can easily program the so-called profiled pulses with almost arbitrary amplitude envelopes.
**Warning:** Although the attenuations are specified in dB, the actual output level need not be what you expect since the specification applies to the input signal level to the final transmitter booster. It then depends upon the latter unit's class (A,B,C) and linearity, what the output pulse RF amplitude shall actually be. With most class C transmitter boosters, the non-linearity is quite strong and the transition from completely OFF to completely ON occurs within a limited range of some 12 dB (the purpose of the TATT parameter is in fact to adapt the transmitter input level to its opening threshold). If you need to know quantitatively the correspondence between the attenuation level settings and the actual output power, you should pre-calibrate them.

(iii) Pulser channels switch
The pulser channels switch is an intrinsic pulser-controlled device which may appear in pulser interval parameters options with the pre-defined device identifier c. As such, it may take part in data acquisition phase cycling.
The c-device option arguments may be strings of letters, each of which specifies a combination of pulser channels which should be made active during the specified interval.

(iv) Receiver Phase
The special parameter ENDS (End of sweep) located at the bottom of the *Acq.par|Delays* parameters table is somewhat anomalous since

a) its value is irrelevant and inaccessible,
b) it accepts option specifications like a pulser interval parameter but
c) it does not correspond to any actual time interval within a pulse/experiment sequence.

The purpose of this parameter is to define what we call accumulation directives. Accumulation directives tell the acquisition & accumulation manager board (AQM) what to do after every pulser scan/sweep. A list of such directives is transferred to the AQM every time the pulser is programmed so it can be viewed as an extension of the pulser program.

At present, the ENDS parameter's options follow the general syntax of all pulser interval parameter but only the phase option is significant and has any practical effect on the data. It tells the AQM in which way the acquired data should be summed/subtracted to the accumulation buffer after every scan - a piece of information which is an essential part of any phase cycling process.

This information is traditionally referred to as the receiver phase - a term originated from early days of NMR when data accumulation was carried out by rather inflexible hard-wired gear and the flexibility required by phase cycling was achieved by using a special receiver phase selector acting on the two reference channels of the RF phase detector. Today, this kind of approach has been all but abandoned since it requires additional hardware where software is sufficient along with being much more flexible.

There are four different modes of "adding" newly acquired data to the accumulation buffers. These are distinguished by the receiver phase settings x, −x, y, and −y in the p(...) option of the ENDS parameter. Considering that the AQM manages two input channels (the in-phase A-channel and the out-of phase B-channel) and two corresponding accumulation buffers (I and II), the four accumulation modes are:

| Rec."phase" | A buffer | B buffer |
|---|---|---|
| x | added to I | added to II |
| −x | subtracted I | subtracted from II |
| y | added to II | subtracted from I |
| −y | subtracted from II | added to I |

With this convention, setting up both single- and quadrature-detection phase cycles is extremely easy since there is a direct correspondence between the RF pulse phases and the receiver phase. In the one-pulse sequence, in particular, when the pulse phase is cycled according to the PW parameter option

p(x,−x,y,−y)

then the receiver phase must be cycled according to the corresponding ENDS option

p(x,−x,y,−y)

in order to accumulate the data according to the quad-detection CYCLOPS technique.

(v) Magnetic field level (in FFC)

A User-defined X-device is any external piece of hardware which can be controlled by means of one or more pulser output channels (TTL control voltage levels). Any such device can be assigned a device identifier (a small letter other than p or a which are reserved for intrinsic devices). Apart from the identifier, one must also define a set of possible device states, each of which is labeled by a single capital letter and is assigned a distinct combination of pulser control channels.

All X-devices are defined in the Parameters.def text file in a way which shall be described below.

Once the definitions are made, the X-device may appear in pulser interval parameters options and thus take part in data acquisition phase cycling.

Typical examples of X-devices are (assignments based on current Stelar conventions):

m ... fast-field-cycling magnet field multiplexer with the following states:

O ... Off (no pulser channel)
R ... Relaxation field (pulser channel I)
A ... Acquisition field (pulser channel V)
P ... Polarization field (pulser channels IV)

d ... RF decoupler with the following states:
O ... Off (no pulser channel)
C ... CW irradiation (pulser channel O)
B ... Broad-band modulated irradiation (pulser channels OM)

H ... Homodecoupling (pulser channels OX)

We shall use the m-device as an example of how the X-devices are defined. Essentially, just one parameter, SSPC, appearing within the Parameters.def file is involved (if it is not displayed in your Configuration Parameters Table, you can inspect its current settings by means of the macro command ShowNmrPar with argument "SSPC"). Its value is a string (usually HTRRIVOMXFDSZ) which assigns a letter to every pulser output channel (ask Stelar for the order in which the respective BNC connectors are mounted on the AQM back panel).

What interests us more, however, is the options string of the SSPC parameter. Even though SSPC is not a pulser interval and, even if displayed, its options string is not accessible, the content of its options string is not a comment. It is the place where all X-devices are defined.

In the FFC case that we are discussing, there the following specification:

m(O=, R=I, A=V, P=IV)

which is interpreted by the system as follows: There is a device with identifier m which has four pulser-controlled states labeled as O, R, A and P. Each state is assigned a combination of pulser channels (using the pulser channel definitions discussed in the preceding paragraph). Thus, for example, when the delay interval D3 has been given the elementary option m(P), the m-device shall be driven in state P during that interval, meaning that pulser channels I and V shall be both active. This is all we need to define and synchronize FFC magnet cycling with the pulse sequences!
Let us now consider a hypothetical example. Suppose we wish to interface a UV flash lamp controlled by a single ON/OFF pulser channel, synchronized with the pulse sequences with full data-acquisition cycling capabilities. We shall assign the letter f to this device (notice that letters p, a and m are already taken up) and denote its states as O (flash OFF) and F (flash ON). We must edit the Parameters.def file, setting the parameter's SSPC options string to

m(O=, R=I, A=V, P=IV) f(O=,F=O)

In this case the pulser channel used for the lamp control shall be 'O'. We have chosen this one since it is still free. Keep in mind that channels T (transmitter gate), R (receiver gates), F

(internal CPU flag), D (internal digitizer strobe) and S (sweep start) have pre-assigned meanings and may not be used. One cannot use the channels specified by the parameters TXEN (transmitter enable) and PINC (phase increment) and the channels used up by other devices (in this case I and V which control the magnet)also.

The last thing to check is the TTL channel polarity defined by the parameter PCPM (pulser channel polarity mask). If the letter in the PCPM value string corresponding in position to letter O in SSPC value string is H, the channel is at high TTL voltage when active and at low TTL voltage when inactive (positive logic); otherwise, when the PCPM letter is L, the situation is inverted (negative logic). Since there are also moments when the pulser output is disabled, we have to check that in such a state the 'O' output is low and the lamp is OFF. This is configured by means of a pull-up/pull-down jumper on the AQM board (again, if you encounter difficulties, consult Stelar).

At this point it is sufficient to reenter the AcqNMR32 program and the f-device becomes operative. For example, should we desire in some sequence to have the magnet in the P-state (polarization field) and the lamp in the Y-state (ON) during the D3 interval, we would give D3 the options string

m(P) f(Y)

#### 4.2.2.7 Parameters options syntax

Pulser interval parameter options are specified by means of a single-string script containing any number of elementary options which may be (but need not be) separated by white space. All specified elementary options shall be in effect during the pulser interval specified by the parameter.

Each elementary option specification consists of:

- Device identifier, which is a small letter specifying either an intrinsic device or a User-defined one.
- A list of arguments enclosed in parentheses and separated by commas.

Each argument defines a state of the specified device. For User-defined X-devices, the states are represented by capital letters. In the case of intrinsic devices, numeric values and special codes may be meaningful (see below). The list of arguments corresponds to one complete cycle, specific to the particular combination of pulser interval & device (cycle lengths for different devices may be different). When there is just one argument, the device shall be set to the specified state in all scans.

Multiple elementary options with the same device are not allowed and shall not be accepted.
Likewise, elementary options specifying an unknown device shall not be accepted

Example of a valid options string: Assume that the specified options of D3 are

x(L) y(A,B) z(P,Q,R)

presuming that there are three X-devices x, y, z which can be set to the following states:

L for device x,
A, B for device y,
P, Q and R for device z.

The options specify that during D3 the three devices shall be in the following states:

x=L, y=A, z=P in the first scan
x=L, y=B, z=Q in the second scan
x=L, y=A, z=R in the third scan
x=L, y=B, z=P in scan #4
x=L, y=A, z=Q in scan #5
x=L, y=B, z=R in scan #6
x=L, y=A, z=P in scan #7 (the same situation as in the first scan),
etc., in a cycle of total length 6.

In order to complete the example, we need to know where and how are the devices x, y, z (and their possible states) defined. For this, we must distinguish between

a) pre-defined intrinsic devices and
b) User-defined devices which require a definition in the Parameters.def file.

Please go through section 3.2.2.6 for more details.

#### 4.2.2.8 Saving (Loading) Parameters in (from) a Parameters File

A parameter file is not to be confused with a data file since it contains only parameters and no data. The default extension for this parameters file is *.par

To save the current set of parameters in a parameters file, use the submenu *Tools*|*Save* parameters. When a file save dialog appears, specify a file name (or select an existing one) and press the Save button.

When the specified file already exists, you shall be asked whether it may be overwritten; otherwise, a new file is created. Keep in mind that you don't need to specify the file name extension (though you may do so). When missing, the default extension .par of parameter files is automatically appended.

A parameters file is a plain text file containing a list of all parameters.
Each parameter entry contains its current value and, if operational, its current options.

Loading parameters from a parameters file or data file

Loading parameters from a parameters file is quite straightforward and requires only the specification of the path\name of an existing parameters file.
Not all the parameters are actually saved/loaded in/from the parameter files. In general, only the parameters which characterize the data are considered; these are the same which are saved together with data in individual zones or data files. Parameters which regard only the data system environment (file name, current path, current zone), the console interface (CI) technical characteristics or other hardware configuration parameters are obviously not transferred.

Notice that upon exit AcqNMR program, the system always saves its last parameters, and re-loads them (if present) upon executing again. This "last parameters" file is called $$Last$$.par. You may reload it at any time if you wish to restore the start-up conditions. If you delete this file, a set of primordial parameters will be created, based upon the sacred parameter definitions file (*.def) (please do not think about modifying these *.def files).

### 4.2.3 Immediate Execution Mode

In the immediate execution mode the user manually edits the system parameters and then uses the immediate mode buttons to acquire data (**Go** button) and save them (**Save** button).

In this mode there is very little automation but the User has the possibility to modify literally every aspect of the experiment. It is used, in general, for preparatory work on a new type of sample/samples or for development of new experimental techniques.
While the system acquires data in the immediate mode, actions which use the instrument's hardware (another acquisition, probe tuning, digital scope, magnet null current calibration, diagnostics, etc.) are forbidden. It will also not allow the User to change any of the system parameters, which might have an effect on the data (while parameters which do not affect data may still be edited).

In order to proceed with any such action, the User must make sure that the data acquisition has stopped and that there are no unsaved data on the AQM (Acquisition and Control Manager) board.

For the first part (stopping acquisition), he/she must either
- wait until the preset maximum number of scans (MS) has been reached,
- or press the Quit button and wait until the current scan is completed and acquisition stops,
- or press the Kill button to interrupt the acquisition and discard the acquired data.

For the second part (getting rid of acquired data), he/she must either
- press the Save button and save the data in a new zone of a data file,
- or press the Kill button to discard the data (thereafter the Kill button will become disabled).

### 4.2.4 Macro command mode

The macro command mode is entered through the 'Tools|Run macro' menu option. Macro commands represent a powerful automation feature. Once a macro command is launched, the system:

1. Displays the command name in the top-macro panel
2. Disables all the immediate mode buttons except Kill which may be used to interrupt the macro.
3. Disables the main menu and even the window's system menu.
4. Disables editing of all parameters.

The User may still inspect parameter tables, report sheet, XY monitor, system log, etc. but no change is allowed in the parameter.

As soon as the macro command's execution terminates, the system reverts to the immediate execution mode.

### 4.2.5 Evaluation of acquired data

An on-line numeric evaluation of the acquired data is provided by the 'Evaluate|Evaluation dialog' menu command. The program displays a dialog where the User can select:

1. The data source, i.e., a **zone** in a **data file**.
2. The **evaluation procedure** to carry out.
3. The destination of the results (report sheet and/or export file)
4. An optional **export file** for the results.

Once the selections are made and the Execute button of the dialog is pressed, the selected evaluation procedure is carried out and the results are sent to the selected destinations. Some of the evaluation procedures, such as raw data list, may appear trivial. However, since they convert binary data to a formatted ASCII form, they represent a useful way of exporting raw data to external data-evaluation programs.

Estimation of **F1** and receiver phase (**RPHS**):

Select "correct F1 using the last data" option in Evaluate menu or press |Ctrl> and |F6> keys together to compute the offset F1 and the receiver phase **RPHS** from the acquired data. In the case of multi-block data, it averages the results over a range of blocks (**EWIB**, **EWEB**). The final data are displayed in the Evaluation page and wherever **F1** parameter is listed. The FID window which should be used for the calculation, as well as the range of blocks which should be used, are specified by the parameter **EWIP**, **EWEP**, **EWIB**, **EWEB.**

Estimation of $T_1$ and $T_2$:

Select "Evaluation Dialog" in Evaluate menu to estimate the relaxation time and rate of a multi-block data array under the assumption that the phenomenon is a single exponential decay. It accepts only multi-block data. The calculation is based on the magnitude averages taken over an FID window. Statistical evaluation of the result is also carried out and all the results are displayed in the Evaluation page. The FID window which should be used for the calculation, as well as the range of blocks that should be used, are given by the parameters **EWIP**, **EWEP**, **EWIB**, **EWEB.**

Notice that:

1. Data evaluation is always possible even while immediate mode data acquisition is in progress, but
2. it is disabled during a macro command execution.
3. More complex data evaluation procedures may be implemented as macro commands.

## 4.3 AcqNMR Parameters

In this section we will review and comment all the FFC acquisition parameters which are of any relevance to the user. We will list all the parameters which could appear in principle.

In *Main parameter* page:

**SMPL** : Sample Name - plain ASCII input

**EXP** : Experiment type

Always accessible. Setting a new EXP starts an extensive update of many parameters. In particular, all Pn and Dn (pulse and delay) are reset (including option switches), as well as all parameters related to specialized hardware. According to the selected experiment, whole groups of parameters may be declared –na– (not applicable), others may become active and still others may be enabled/disabled for User access.

**TEMP**: Temperature of the sample
This parameter is not interfaced hence it acts like a comment notes.

**FILE** : Acquisition Data file name
Always accessible. It enables the file access dialog box and asks the user to enter/select the data file name.

**SF** : Spectrometer frequency (reference for the measuring channel)
When **SF** is changed, the system will update the field of acquisition (**BACQ)** according to the current **NUC**. Do not forget that, each time you change **SF**, you must change/tune, the spectrometer hardware – probe head.

**F1** : Observe offset
Carrier offset from **SF** (in Hz).

**RFA** : RF attenuation
RF level attenuation in dB.

**GAIN** : AF Gain
AF receiver gain factor

**T1MX**: Maximum T1 value in seconds

**TPOL** : Polarization time in seconds

**SWT** : Switching time of the magnet (in seconds)

**RD** : Re-cycle Delay in seconds. Time between repetition of experiments.

**TAU** : Delay tau (in seconds). Duration of relaxation field (**BRLX**)

**EDLY** : Echo delay in microseconds. Delay between 90 degree and 180 degree pulse.

**BPOL** : Polarization Field in MHz

**BRLX** : Relaxation Field in MHz

**BACQ**: Acquisition Field in MHz
These field values are specified by means of $^{1}H$ Larmor frequency.

In *Acq.Par* page:
in *User* Page

**USER** : User ID
**OPER** : Operator name
**MXZN**: Number of valid data zone in the data file.

**N1** to **N4** and **AUX1** to **AUX3**: are notes. Users can use these columns to write some comments about their sample/experiment etc. These comments are stored in the data file when you save the data.

in *Basic* Page

**NUC** : Nucleus

A list of nucleus will be displayed when you click this item. User can select the nucleus (or type 1H, 2H, 31P etc.) according to their experiment. The chemical symbol input is case-insensitive. When changing nucleus, the acquisition field (**BACQ**) value will change automatically according to the System Frequency (**SF**).

**NS** : Number of actually completed scans

Inaccessible from the keyboard, maintained by the system. This parameter indicates the number of scans which have been completed during an acquisition.

**MS** : Maximum number of scans

Always accessible. Allowed values are 0 to 2147483647 (i.e. $2^{31}$-1). This is the maximum number of desired scans. After a data acquisition has been started, it proceeds until it is either interrupted by the operator or until MS scans are completed. If **MS** is set to 0, the acquisition proceeds indefinitely and can be stopped only by the operator. When setting **MS**, the acquisition time (**TTIM**) is recalculated.

**DS** : Dummy scans

Dummy scans are useful to pre-establish the thermal cycle of the magnet before starting acquisition. They are very important when using short recycle delays (**RD**) and a high number of scans.

**BS** : Block Size

Number of data points in a single block. **BS** must be a power of 2; its minimum value is 2 in single detector mode and 4 in quadrature detection. The maximum value is 32768 (32K). In quadrature detection (**FTM**=0) BS is split into halves, one reserved for the in-phase ("real") data and the other for the out-of-phase ("imaginary") data.

**STIM** : Sweep time

This sweep time is calculated from the block size (BS) and sweep width (SW). During this time magnet is in Acquisition field.

**SW** : Sweep width.

Spectral window width in Hz (Nyquist frequency). The maximum range is 10MHz. In practice, **SW** determines the dwell time (**DW**) i.e. the time interval between two consecutive samples of the data. The relationship is

**SW** = 0.5/**DW** in single detection and **SW** = 1/**DW** in quadrature.

In both cases, **SW** equals the Nyquist frequency. The latter is the maximum frequency which, when present, is correctly defined by the digitized data; any frequency which is higher will be folded back into the Nyquist range.

SW can therefore also be considered as the Spectral Window for the "true" signals. Ideally, however, any signal frequency (spectral line in HR-FFC), which is outside this window, should be filtered-out before the digitization. Consequently, a change in **SW** not only causes an update of **DW** but also sets a default value of **FLTR** (the frequency cut-off of the LF filters) if you enable auto-filter (**AFLT**) mode in *Conf* window.

**RTIM** : Repetition time.

Not accessible to the user. It is the repetition interval between the scans in seconds. When **T1MX**, **TPOL** are changed, RTIM also changes accordingly.

**TTIM** : Total time

Not accessible. Total time required for one experiment. This value is calculated from the relation **TTIM** = **RTIM***MS**.

**FLTR** : Cut off frequency of the audio signal filters.

in *Pulse* page:

**PW** : Acquisition pulse (default value is $90^o$)

**P1** to **P16** : Pulser channel

The values of RF pulses (**PW**, **P1** to **P16**) are given in degrees of the nutation angle. During virtual pulser programming these values are converted to time intervals using the current 90 degrees pulse width (**PW90**) as conversion factor. Thus if **PW90** is 5 μsec and **P1** is set to 180 degrees, it eventually evaluates to 10 μsec.

in *Delay* page:

**D0** to **D16** : Delay channel

Depends on the sequence being used, these delay channels are activated and assigned the values automatically according to the pulse script.

in *nDim* page:

**NBLK** : Number of Blocks in an experiment

The number of different **TAU** intervals is specified through the **NBLK** parameter (number of blocks). The default value of **NBLK** is 1 to save the time (after setting all parameters specify the number of blocks, otherwise, the program will evaluate all **TAU** intervals at any change of parameters resulting in an unnecessary wastage of time).

**BACR** : Block Acronym

Parameter which changes between the blocks (usually **TAU**)

**BINI** : Initial value of the Block parameter

For example, in PP/S sequence, the initial value of BINI is 4***T1MX**.

**BEND** : Final value of the Block parameter (default value is 0.001sec).

**BGRD** : Block Grid type

Type of point distribution from **BINI** to **BEND** (Linear or Logarithmic or user specified).

**BLIST**: User specified point distribution from **BINI** to **BEND**.

For example, if the user wants to use two different sets of point distribution over the range, use the following rule

$A_1$:$B_1$:T:N;$A_2$:$B_2$:T:N

where A and B are initial and end values of the range, T is type of point distribution either LIN or LOG and N is the number of points in the range A and B. One can use expressions also in the place of A and B but they should be enclosed by parenthesis.

Example: 0:0.1:LOG:8;0.1:10:LIN:10

OR P1,P2,P3,……..Pn where P's are points. Ex. 0,0.1,0.2,0.5,0.9,1.5,2;5,10,20

in *Conf* page:

**HEAD**: Probe head
Type/code of the used probe head. This is just a note.
**PW90** : 90 degree pulse width.
Duration of 90-degree pulse in microseconds.

**PDMX**: Phase/Diode detection
Select P for phase detection mode or D for diode detection.

**FTM** : Fourier Transform detection mode.
Always accessible. Allowed values: -1, 0, +1. **FTM** is used to specify the detection mode. When **FTM**=0 (which is the default value) the quadrature detection is used, while **FTM**=+1 or –1 indicates the use of a single detector channel. In the later case, there is no difference in the acquired data for the two settings; they are used only in the data evaluation routine to achieve proper orientation of the spectra.
The logic behind the three values is linked to the carrier frequency with respect to the spectrum:
-1 indicates that the carrier is at the left border of the spectrum (high carrier offset, low field);
0 indicates that the carrier is at the centre of the spectrum and thus implies the use of quadrature detection;
+1 indicates that the carrier is at the right border of the spectrum (low carrier offset, high field).

The system has no way to check whether the **F1** offset really is where it should be with respect to the spectrum. An incorrect indication may result in folding or in the reversal of the spectrum.
It is important to note that on systems which actually have two detectors in quadrature, the **FTM**=0 settings will result in the use of both of them. Systems with only one digitizer will "simulate" the other by replacing the second channel data with zeros. It is still possible to take advantage of the **FTM**=0 mode, provided that one uses a correct phase cycle (pseudo-quadrature). See also the Quadrature Detection section of this manual for more details.

**B1** : B1 field in Gauss. This value is determined from **PW90**.

**RINH** : Receiver inhibit
After application of a **PW90** pulse, the receiver is disabled for a **RINH** time to avoid loading the audio filter. **RINH** is of the order of the receiver + probe dead time.

**ACQD**: Acquisition Delay
Acquisition starts after this delay time from the acquisition Pulse ($90^o$pulse).

**RPHS** : Receiver Phase
To maximize the in-phase signal just after the FID.

**QPHC**: Quadrature phase setting
To adjust the orthogonality of the two signal channels. Further details can be found from the Quadrature Detection section.

**PHCY** : Phase Cycle Enable.
To enable or disable Phase cycling method. Further details can be found from Phase Cycling section.

**AFLT** : Auto filter mode
This parameter enables/disables automatic audio filter settings (**FLTR**)

**DAAM** : Data acquisition and accumulation mode
*BASIC*: the normal mode used
*MUTE*: this mode completely suppresses monitoring of data during acquisition except for the scans counter which remains active.

**DAAP** : delay between acquisition enquiries (in milliseconds). To control the rate at which the PC checks the progress of acquisition. Its value is in milliseconds. The recommended default value is 200 (allowed range is 10 to 2000)
For a detailed description of these two parameters, see section 4 in Appendix – 1.

in *Hard* page:

**INST** : Instrument type (not accessible)

**B0** : Larmor field (in Tesla)
The field corresponding to System Frequency (**SF**) and **F1** (**B0** = **SF**+**F1**)

**IOFF** : Magnet current offset.
When running relaxation measurements at low magnetic fields, the current offset of the magnet becomes very important. A high positive offset could limit the minimum Larmor frequency for the relaxation profile to several kHz (this will originate an artificial plateau), while a negative offset will produce undesired effects on the magnetization. For more details please see the section 2.4.2 Null current calibration.

**FOFF** : Magnet B offset (Compensation of background magnetic field)
For more information please refer section 2.4.6 Environmental Field compensation

**SLEW**: Magnet slew rate (MHz/ms)
This parameter determines the slope of the Magnet Switching pattern.

**MTCF**: Magnet Temperature Compensation Factor (Hz/degC)
For further details please refer section 4.9

**SSPC** : Pulser Channels

**PCPM**: Pulser Polarity mask

**TXEN** :Tx enable channel - used to enable the power output of final RF boosters
**PINC** : Phase increment channel – used to advance DDS phase & attenuation stack address

The above four parameters are not accessible.

**TATT** : Transmitter base attenuation
This changes the amplitude of pulsed RF input to the transmitter. The default value is 18.

If your 90-deg pulse length is not short enough, decrease this **TATT** value.

in *Evaluation* page:
in *quick results* page sheet
**EDF1** : Data file name
**EDZ1** : Data zone – Number of the evaluated data zone
**EACV** : Brelax (MHz) - Value of the parameter specified in EACN
**ET1** : $T_1$ (sec) – Value of Relaxation time
**ER1** : $R=1/T_1$ ($sec^{-1}$) – Relaxation Rate
**ER1D** : Error in R (+/–) – Absolute error in Relaxation Rate
**ER1E** : Error in R (%) – Relaxation Rate (and $T_1$) relative error
**EDF2** : Data file
**EDZ2** : Data zone – Number of the evaluated data zone
**EF1** : F1 estimate (Hz) – Mean value of the Signal Offset
**EF1S** : F1 spread (Hz) – Maximum-minimum offset value (for multiblock data only)
**ERPH** : Rx.phase estimate – Estimated Receiver phase (mean value) from the acquired data in *Parameters* page sheet
**EWIP** : Initial point – starting point for the evaluation in a single block
**EWEP**: End point – Final point for the evaluation in a single block
**EWIB** : Initial Block – Starting Block for the evaluation
**EWEB**: End Block – Final block for the evaluation
(for more details about these four parameters, see the following section "Numeric Data Evaluation Procedure)
**EACN** : Acronym of 3rd Dimension parameter in 3D data
**PREP** : Parameters, those are to be displayed when selecting a zone in a data file
This parameter is actually a configuration structure for the *Data File Selection* dialogs. These are displayed at various occasions, including:
- Loading parameters from a data file (menu *Tools*|*Load parameters*|*from a Data File*),
- Selecting a data file & zone in the *Data Evaluation Task dialog* (menu *Evaluate*|*Evaluate data*).
The value of the **PREP** parameter is a list of parameter acronyms, separated by commas, which shall be displayed in the *Parameters preview* window of any *Data File Selection* dialog.
When the list is void (a blank string), all stored parameters of the data file zone are shown.
When a listed parameter is not found among the stored ones, it is ignored (remember that parameters with void values and/or hidden parameters are never stored in data files).
**PREP** itself is not stored in data files but it is listed in parameter files. Its value therefore remains unchanged when loading system parameters from a data file, but it gets updated when loading a parameter file.

**F1CL** : Maximum F1 shift (absolute value)
This parameter is used by the F1 correction routine (menu command *Evaluate*|*Correct F1*). Its purpose is to prevent a “wrong” F1 correction which might occur with very noisy data.
The current value of F1 shall never change by more than the value of **F1CL**. For example, if current value of F1 is 12000 Hz and F1CL is set to 20000, the newly estimated F1 shall not exceed the interval of -8000 to 32000 Hz. Should the calculated F1 value be lower than -8000 Hz, it would be replaced by -8000 Hz; likewise, should it exceed 32000 Hz, it would be reset to that value.

The default value of **F1CL** is 50000 Hz. The User can change this to anything from 10 kHz and 100 kHz (clearly, only positive values are allowed). The limits are specified in the Parameters.def file and, if need be, may be changed.

Note: The **F1CL** value does not limit the User in directly setting the F1 value (either manually or from a macro). It is tested exclusively upon exit from the F1 correction routine.

**T1CR** : T1 correction routine. **T1CR** is a parameter with a predefined list of possible values. There are three types of relaxation rate estimation procedures that exists in this software. This parameter is to select one out of the three procedures to be used for T1 correction method. (For a detailed description of the individual evaluation procedure see the following section "Numeric Data Evaluation Procedure")

**T1CL** : T1MX correction lower limit.
**T1CU** : T1MX correction upper limit.

These three parameters (**T1CR**, **T1CL**, **T1CU**) are used exclusively by the **T1MX** correction routine (menu command *Evaluate|Correct T1MX*). Their purpose is
a) to specify which T1 evaluation procedure should be used and
b) to prevent a non-sensual T1 correction which might occur when, for whatever reason, the data are pathological.

Note that in FFC (Fast Field Cycling) **T1MX** is a critical parameter since it determines the duration of the intervals when the magnet is switched ON. Imposing an upper limit on **T1MX** is therefore not just a logical thing to do but an important safety feature as well.

**T1CL** is the lower limit for **T1MX**. Should the **T1MX** correction end up with a value smaller than **T1CL**, it would be reset to this limit. The default for this parameter is 0.001s (1ms) with an allowed range of 0.0001s (0.1ms) to 1s.

**T1CU** is the upper limit for **T1MX**. Should the **T1MX** correction end up with a value greater than **T1CU**, it would be reset to this limit. The default for this parameter is 5s with an allowed range of 1s to 10s.
Note: The **T1CL** and **T1CU** values do not limit the User in directly setting the **T1MX** value (either manually or from a macro). They are tested exclusively upon exit from the T1 correction routine.

The default values of all the above parameters, as well as their ranges, are defined in the *Parameters.def* file and, should it be necessary, can be modified.

## 4.4 Numeric Data Evaluation Procedure

### 4.4.1 Evaluation of acquired data

An on-line numeric evaluation of the acquired data is provided by the '*Evaluate|Evaluate data*' menu command. The program displays an evaluation task definition dialog where the User can select:
1. The data source, i.e., a zone in a data file.
2. The evaluation procedure to carry out.
3. The destination of the results (report sheet and/or graphic sheet)

4. An optional export file for the results.

Once the selections are made and the Execute button of the dialog is pressed, the selected evaluation procedure is carried out and the results are sent to the selected destinations. Some of the evaluation procedures, such as raw data list, may appear trivial. However, since they convert binary data to a formatted ASCII form, they represent a useful way of exporting raw data to external data-evaluation programs.

Notice that:
a) Data evaluation is possible even while immediate mode data acquisition is in progress, while
b) it is disabled during a macro command execution.
c) More complex data evaluation procedures can be implemented as macro commands.

### 4.4.2 Numeric Evaluation Review

It is evident that, in principle, a data acquisition package is not responsible for the subsequent evaluation of the acquired data. In NMR, in particular, the types and the complexity of the acquired data are such that off-line evaluation programs (sometimes written ad-hoc by the User) are quite common.

Despite this fact, however, the NMR32 package includes a number of on-line data evaluation routines. The reasons for including such procedures are multiple:

**a) Data export**

Data export to external programs is of course a must. One can always export raw data, but often it is preferable to subject them first to a partial, application-neutral pre-processing (e.g., exporting magnitudes of the signals rather than the in-phase and out-of-phase components).

**b) Routine measurements**

It turns out that in any branch of NMR, some 90% of data processing are standard and can be covered by very few well-defined routines, while the remaining 10% are spread over a vast number of non-standard evaluation procedures. Having the standard routine(s) available on-line within the framework of the data acquisition program is practical.

**c) Experiment optimization**

Even a preliminary evaluation may be extremely useful when one tries to optimize the data acquisition parameters for a particular sample. In NMR relaxometry, for example, an approximate knowledge of the actual relaxation times is essential for proper settings of acquisition parameters.

### 4.4.3 Evaluation Procedure

A run-time list of the currently available data evaluation procedures appears in a list box of the *evaluation task definition dialog*. The procedures can be divided into two categories:

**1.** Procedures applicable to any data set:

- LRD - List of raw data
- LSP - List stored parameters

- WAV - Data Window : Average(s)
- WAM - Data Window: Signed magnitude(s)
- WSM - Data Window: Absolute magnitude (s)
- Offset and Phase estimate(s)

**2.** Procedures applicable only to *multi-block data sets*:

- RAM - Relaxation rate estimate using WAM procedure
- RSM - Relaxation rate estimate using WSM procedure
- RAV - Relaxation rate estimate using real WAV
- T2M - $T_2^*$ estimate using magnitudes
- T2R - $T_2^*$ estimate using real part

All evaluation procedures operate on the data subset specified by the 'data-window parameters'. The procedures listed in category (2) also make use of the parameter **EACN** (Evaluation Acronym). The value of **EACN** does not affect calculations - it just specifies a parameter, which characterizes the whole multi-block data set (e.g., the relaxation field **BRLX**). The value of the latter is listed in the evaluation procedure's reports together with its principal results.

#### 4.4.3.1 Evaluation Data Page

The data window (or data subset) is defined by means of two pairs of parameters within the *evaluation parameters table*:

**a) EWIP** and **EWEP** (*Evaluation Window Initial Point* and *Evaluation Window End Point*, respectively) define a **section of each FID data array** to be used (don't get confused by the term *window* which in this case refers to a section of experimental data rather than a WINDOWS window).
By convention, data points numbering starts at 1. Setting **EWEP** to 0 (default) is equivalent to saying *'until the last FID data point'*. When non-zero, logic commands that **EWEP** should be greater than **EWIP**. Should this not be the case, the evaluation procedure automatically swaps the two values. Since the first few points of a FID are often 'contaminated' by pulse leakage and/or filter settling artifacts, the default value for **EWIP** is 6. However, the User may set it to any value greater than 0. When any of the two parameters is larger than the size of the FID data array, the evaluation procedure replaces it automatically by the actual size.

**b) EWIB** and **EWEB** (*Evaluation Window Initial Block* and *Evaluation Window End Block*, respectively) define an **interval of data blocks** to consider in multi-block experiments (these two parameters are ignored when evaluating a single-block data set).
By convention, data blocks numbering starts at 1. Setting **EWEB** to 0 (default) is equivalent to saying *'until the last data block'*. When non-zero, logic commands that **EWEB** should be greater than **EWIB**. Should this not be the case, the evaluation procedure automatically swaps the two values. The default value for **EWIB** is 1. When any of the two parameters is larger than the actual number of blocks, the evaluation procedure replaces it automatically by the latter value.

**Note:**

The right window displays the X and Y coordinates of the mouse cursor when it moves across the XY monitor screen on which data are displayed. The values are in real-world units (e.g., X is in milliseconds when an FID is displayed. In the case of multi-block data, the block number and point number is also shown.

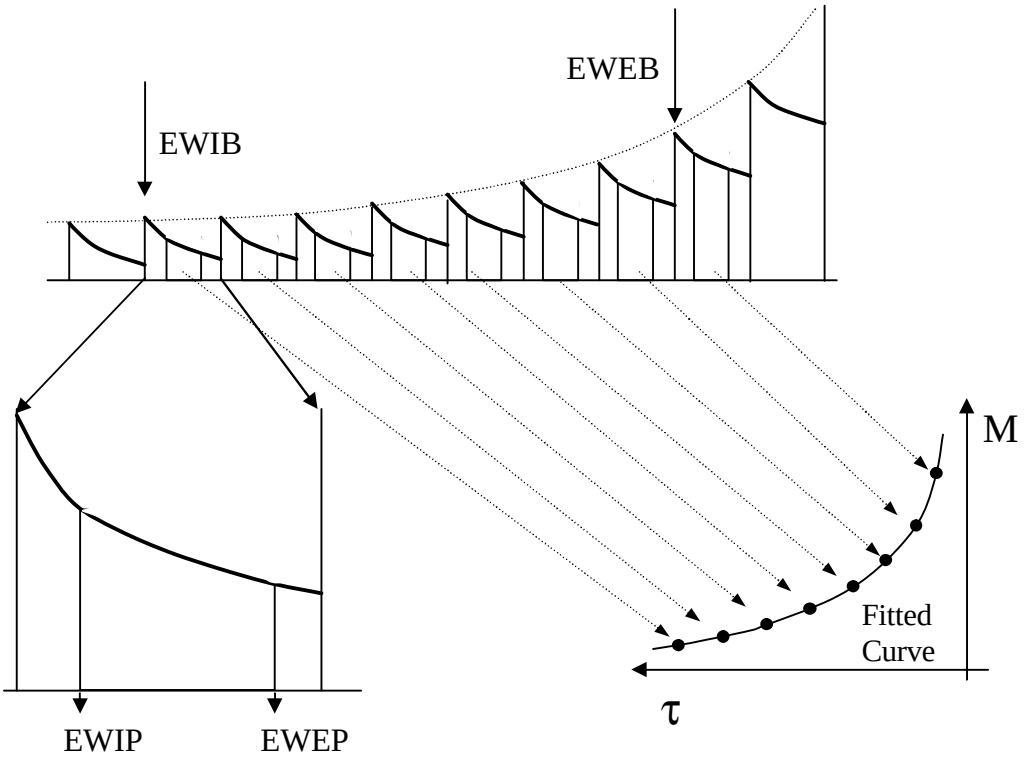



Fig. 15. FID obtained from a typical multi-block sequence

#### 4.4.3.2 List of Raw Data (LRD)

This evaluation procedure does nothing at all with the data except converting them to an ASCII form. It is used principally to **export the raw data** into ASCII files usable by other applications.

This procedure generates a report consisting of a listing of all data points comprised in the selected data zone. Since the report may be quite long (thousands of lines), it may get truncated when sent to the program's *Report Sheet* (1000 lines maximum). However, no such truncation occurs when the report is written into an export file. The following report format is used:

```
  sec              Real            Imag
-------------------------------------------------
Block 1
 1.067e–04      4.902e+01      –7.800e+01
 3.167e–05      1.220e+02      –1.112e+02
...
Block 2
 1.067e–04      1.386e+02      –1.099e+02
 3.167e–05      2.000e+02      –9.197e+01
...
...
```

The 'imaginary' components of each data point are listed only when the data are acquired using quadrature detection (**FTM** = 0). The terms 'Real' and 'Imag' refer simply to the detector channels A (in-phase) and B (out-of phase) and do not imply the employment of any phasing algorithm.

#### 4.4.3.3 List of Stored Parameters (LSP)

This evaluation procedure is much like the **LRD** (list of raw data) except that the Report it generates does not contain data but rather a list of all the parameters stored in the selected data file zone. The parameters are listed in the same order in which they appear in the parameter tables (with a line of hyphens separating parameters belonging to different tables).

Like LRD, when graphic output has been specified, the LSD procedure produces the DisDat graph. This amounts to a very efficient and comfortable data zone preview since one ends-up with a simultaneous view of the DisDat graph and a scrollable list of all pertinent parameters.

#### 4.4.3.4 Data Window Averages (WAV)

This evaluation procedure generates one complex datum for each FID window (i.e., one for each block). The returned value is the arithmetic average of all the data points, which lay within the window. The report generated by the procedure has the following format:

| TAU | R-Average | I-Average |
|---|---|---|
| 2.800e–01 | 1.187e+02 | –4.897e+01 |
| 2.615e–01 | 1.359e+02 | –5.568e+01 |
| ... | | |
| ... | | |

When the original data have no 'imaginary' part (**FTM** ≠ 0), the result has none, too, and the I-Averages are not listed. The terms *R-Average* and *I-Average* refer simply to the detector channels A (in-phase or 'real') and B (out-of phase or 'imaginary'); they do not imply the employment of any phasing algorithm.
The arrayed parameter whose values are listed in the first column is the one whose acronym was specified in **BACR** (*nDim* parameters table) during acquisition of the data. Its values are re-calculated using the stored **BINI**, **BEND** and **BGRD** parameters. When there is just a single block and therefore no BACR parameter, the first column heading is *null* and the value is 0.

NOTE: When **EWIP** = **EWEP**, the FID window consists of just one point. In this case the procedure simply lists the signal values at this point for all blocks of a multi-block experiment.

When graphic output has been specified, the WAV procedure produces the DisDat graph (original, untreated data) plus the Multi-block graph plotting the window averages against the arrayed-parameter values.

#### 4.4.3.5 Data Window: Absolute Magnitude (WAM)

This evaluation procedure generates one real datum for each FID window (i.e., one for each block). The returned value is the average of the magnitudes of all data points, which lay within the window as determined by the formula.

$$m = \frac{1}{n}\sum_{k=1}^{n}\left|a_k + ib_k\right| = \frac{1}{n}\sum_{k=1}^{n}\sqrt{a_k^2 + b_k^2}$$

where $n$ is the number of data points in the window and $a_k$, $b_k$ are the A-channel (in-phase, 'real') and B-channel (out-of-phase, 'imaginary') components of the signal.

The report generated by the procedure has the following format:

```
    TAU          Magnitude
-------------------------------------
 2.800e–01       1.300e+02
 2.615e–01       1.474e+02
...
...
```

Because of the 'rectification' feature of the procedure, the resulting magnitudes may be somewhat contaminated by anomalous noise contribution. This is negligible for data with large signal-to-noise amplitudes. When the S/N ratio is low, however, the contribution of the 'rectified' noise to the resulting values may become significant (even with zero signal, we obtain a non-zero value). The net result of this systematic bias is a *reduction* of the estimated relaxation rate. The analysis of this problem is a matter of a separate Application Note.

Like for WAV, the arrayed parameter whose values are listed in the first column is the one whose acronym was specified in BACR (nDim parameters table) during acquisition of the data. Its values are re-calculated using the stored BINI, BEND and BGRD parameters. When there is just a single block and therefore no BACR parameter, the first column heading is *(null)* and the corresponding value is 0.

When the original data have no 'imaginary' part (FTM≠0), the WAM procedure defaults to WAV. In many applications, this makes it possible to use the same procedure to generate valid intermediate data for all acquisition modes (quadrature phase detection, single-channel phase detection and diode detection).
When graphic output has been specified, the WAM procedure produces the DisDat graph (original, untreated data) plus the Multi-block graph plotting the window 'magnitudes' against the arrayed-parameter values.

#### 4.4.3.6 Data Window: Signed magnitudes (WSM)

This evaluation procedure is like **WAM** (data-window absolute magnitudes) except for an *a-posteriori* attempt to determine the sign of the signal. In particular, this becomes a necessity when, upon stepping of the arrayed parameter, the signal changes sight (a typical example is the IR sequence).

In the case that the data were acquired using quadrature phase detection, the estimate of the actual sign of the signal is carried out as follows:

1) First, the data block wi th the largest absolute magnitude is chosen as 'reference'.
2) For any other block, one calculates the signal correlation coefficient for the two blocks, i.e.,

$$c = \sum_{k=1}^{n} w_k (a_k + ib_k)(a'_k + ib'_k)^* \Big/ \sum_{k=1}^{n} w_k ,$$

where the apostrophe indicates the reference data, the asterix denotes complex conjugate and wk are suitable positive weight factors (at present, wk=1). The discriminating quantity is the *real part* of c - when it is positive, the block is considered 'in-phase' with the reference data and assigned a positive value; otherwise, it is considered to be negative.

Notice that, since both 'real' and 'imaginary' components of the signals are used, the procedure is totally insensitive to the receiver phase (mis)adjustment.

It shall fail and make incorrect sign estimates, however, when there are large signal phase variations between the blocks. Though this is not likely to happen (barring a hardware defect), a situation of this kind may be 'simulated' by instabilities of the signal offset due to magnetic field variations, combined with the use of data points far from the RF pulse. Just after a pulse, the signal always starts with the correct phase, regardless of its offset from resonance. However, when there are field instabilities (like it might happen in FFC), the 'good' region may be masked by the system's dead time and/or by the use of data window points with excessively large indices.

Any sign mis-attribution is usually immediately evident from the generated multi-block graph (non-monotonous curve). If this happens, follow the above generic guidelines to correct the problem or, if possible, avoid using the WSM routine. In any case, if the multi-block graph is not monotonous, don't use the RSM procedure to evaluate the relaxation rate of the data.

The data generated by this procedure are subject to the same 'rectification' problem as in the case of the WAM procedure. In some cases, actually, the situation is better since, after the sign correction, some of the data points may be affected positively and some negatively (there shall appear to be a discontinuous 'step' upon crossing zero signal level). Consequently, fitting procedure such as the one estimating the relaxation rate (RSM) may end up with a less biased value.

When the original data were acquired using phase sensitive detection but have no 'imaginary' part (FTM≠0), the WSM procedure - like WAM - defaults to WAV and the two procedures become identical.

A somewhat different case occurs when the data are acquired in diode detection in which case all points are positive already at the detector level and the above procedure would be pointless. If such data refer, for example, to an IR multi-block experiment, the sign correction is still essential but can be attempted only as a simple guess. The applied algorithm in this case is the following:
1) Start with the first data block and proceed forward for as long as the data values keep monotonously increasing or decreasing. If the whole set is monotonous, the data are left unchanged and we have finished. Otherwise, denote the last monotonous block as b1 and
2) starting from the very last data block, proceed backward for as long as the data vary monotonously and denote the last encountered block as b2 (unless the data are very noisy, we shall have b1=b2).
3) Invert the sign of all blocks b for which b<(b1+b2)/2.
The report generated by the procedure has the same format as for the WAM procedure. Likewise, the graphic output of WSM also follows the same general rules, which apply, to WAM.

#### 4.4.3.7 Offset and Phase estimate

This evaluation procedure is applicable only to quadrature-detection data (**FTM** = 0). When applied to single-detection data, it issues a warning and does nothing more.

With quad data, it generates two real values for each FID window (i.e., one for each block). The returned values are the estimated frequency offsets of the signal from resonance and its estimated phase just after the RF pulse (time $t = 0$).

The report generated by the procedure has the following format:

```
    TAU         Offset cor.      Phase cor.
-------------------------------------------------
 2.800e-01      1.259e+03      -1.261e+01
 2.615e-01      1.139e+03      -9.465e+00
 2.429e-01      1.134e+03      -1.142e+01
...
...
```

where the offset correction is in Hz and phase correction is in degrees. They are called 'corrections' since they should be added to the acquisition parameters **F1** and **RPHS**, respectively, to acquire an in-resonance, in-phase signal. Such an action, of course, is meaningful only with freshly acquired data and it is exactly what happens upon execution of the menu command *Evaluate|Correct F1 using last data* in the case of single-block experiments (in multi-block experiments, the command first averages the offset & phase corrections over the blocks specified by **EWIB** and **EWEB**). With old data, the procedure can be used to check how far from resonance the signal was and how well was the receiver phase adjusted.

The arrayed parameter whose values are listed in the first column is the one whose acronym was specified in **BACR** (*nDim* parameters table) during acquisition of the data. Its values are re-calculated using the stored **BINI**, **BEND** and **BGRD** parameters. When there is just a single block and therefore no **BACR** parameter, the first column heading is *null* and the value is 0.

Every time the offset & phase correction routine is run, the results are used to modify the parameters **EDF2**, **EDZ2**, **EF1**, **EF1S**, **ERPH** in the *Eval|Quick Results* table. Keep in mind, however, that **EF1** and **ERPH** do not report the *corrections* but rather the actual *corrected values* of **F1** and **RPHS**, respectively. Also, in multi-block experiments, the *averages* of the offset and phase corrections over all blocks specified by **EWIB** and **EWEB** are used to calculate **EF1** and **ERPH**, respectively. The **F1** spread **EF1S** the difference between the largest and the smallest value of offset correction among all the blocks (it is a measure of the stability of the offset varies during the experiment).

#### 4.4.3.8 Offset and Phase estimate algorithm

Let [$z_k$], $k$ = 1,2,...,$n$ be an array of consecutive, complex data points of an FID, taken at times $t_k = t_1+(k-1)d$, where the values of $t_1$ and $d$ are known (they can be easily calculated using the absolute index of the first point, the acquisition delay parameter **ACQD** and the dwell time parameter **DW**). Explicitly, $z_k=a_k+ib_k$, where $a_k$, $b_k$ are the A-channel (in-phase, 'real') and B-channel (out-of-phase, 'imaginary') components of the acquired signal.

When projected onto the Cartesian complex plane, an ideal FID describes a spiral (with time as a parameter) starting at a point $z_0$ at $t$=0 and converging to $z$=0 with time going to infinity. The [mean] offset from resonance is in this context defined as the [mean] rate of variation of the azimuth $A(z)$ of $z(t)$. This is a quantity, which is easily estimated from the experimental data set (note that, since averages are involved, the algorithm tends to suppress the unavoidable experimental noise).

The azimuth difference (in radians) between two data points $z$, $z'$ is

$$A(z) - A(z') \equiv \Delta(z, z') = \sin^{-1}\left( \frac{a'b - b'a}{\sqrt{(a^2 + b^2)(a'^2 + b'^2)}} \right)$$

and the average offset (in Hz) for the given data set is

$$f = \frac{1}{2p}\left[ \frac{1}{d(n-1)} \sum_{k=1}^{n-1} \Delta(z_{k+1}, z_k) \right].$$

Once the offset has been determined, it is easy to back-extrapolate the azimuth of the first data point $z_1$ to time $t$=0 and thus determine the azimuth of $z_0$ which defines the overall phase (or 'receiver phase') of the signal. This step, however, is burdened by the noise present in the single data point $z_1$ (no averaging) and hence the phase estimate is much less reliable than the offset estimate and should be trusted only for signals with high signal-to-noise ratio acquired close to resonance. In any case, on most instruments, the receiver phase remains constant in time and fairly independent of gain and filter settings. Moreover, it does not influence the outcome of $T_1$ estimates so that its adjustment is in no way critical.

#### 4.4.3.9 Relaxation Rate Estimate (RAM, RSM, RAV)

These evaluation procedures are applicable only to multi-block data sets (minimum 3 blocks) in which the signal magnitudes depend exponentially on the arrayed parameter (usually TAU). When less than 3 blocks are present in the data zone, a warning is issued and nothing gets done.

The difference between the procedures lies exclusively in the employed intermediate set of multi-block data (see the pertinent sections for a more detailed discussion of their nature). These are:

- **RAM** uses data generated by WAM (data-window absolute amplitudes)
- **RSM** uses data generated by WSM (data-window signed amplitudes)
- **RAV** uses data generated by WAV (data-window averages)

Internally, the procedure first calculates the intermediate multi-block data set and then calculates the relaxation rate R1 (the inverse of T1) by fitting them to the formula

$$m_k(t_k) = a + be^{-rt_k}$$

and identifying R1 with the optimum value of r. Here the index k ranges over all the blocks specified by the evaluation parameters **EWIB** and **EWEB**, $t_k$ is the value of the arrayed parameter (specified by BACR) for the k-th block and a, b, r are the quantities to be estimated by the multivariate, nonlinear fitting algorithm.

The report generated by the procedure has the following format:

| BRLX | R1 | +/- | Zone | File |
|---|---|---|---|---|
| 6.001e-02 | 1.384e+01 | 4.081e-01 | 18 | <filename> |

Notice that the procedure lists the value of the parameter indicated by EACN (in this case BRLX) also. It is up to the user to indicate in EACN a parameter which physically characterizes each set of multiblock data.
Apart from generating the report, the procedure also uses the results to modify the parameters **EDF1**, **EDZ1**, **EACV**, **ET1**, **ER1**, **ER1D** and **ER1E** in the *Eval|Quick Results* table.

**Notes of caution:** The estimated R1 *error interval* (*e*) resulting from the fit of a single multi-block experiment is not to be confused with the r.m.s. *scatter* in R1 values (*s*) when the same experiment is repeated many times. Two situations must be distinguished:
a) The decay is truly mono-exponential (i.e., any deviation from mono-exponential behaviour is much smaller than the experimental noise). In this case one expects a strong correlation between *e* and *s*, with *s* exceeding *e* due to experiment repetition statistics and additional error sources (statistics alone predicts a factor of about 2). When data accumulation is used, both e and s are expected to diminish with the square-root of the number of scans.
b) When the decay is not really mono-exponential, the 'fitting error' *e* contains a systematic component which may be quite large and which is independent of the number of scans. The apparent scatter, however, reflects only the *reproducibility* of the measurements which is the same as in the former case and tends to zero as the number of scans is increased. Anomalously high fitting errors and 'normal' dispersion curve scatter invariably indicate that the decays are not mono-exponential and need to be handled by an external data-evaluation software.

Keep also in mind that the *reproducibility of the error estimate* is much poorer than the error itself. This is a normal statistical phenomenon; an error estimate of, let's say, 2% may be easily burdened by an order-of magnitude higher uncertainty so that repeated experiments may show fitting errors *e* scattered over an interval ranging from 1% to 3%.

#### 4.4.3.10 Relaxation Rate Estimate Algorithm

Let $[m_k]$ and $[t_k]$ be, respectively, the arrays of average data-window magnitudes and of the arrayed parameter values (the index $k$ ranges over all considered blocks, assumed to be $n$ in number, $n>2$). These data are to be fitted by the theoretical formula (the hypothesis)

$$m_k(t_k) = a + be^{-rt_k}$$

where $a$, $b$, and $r$ are some as yet unknown parameters. This requires a non-linear, three-parameter, least-squares fit in which one minimizes the quantity

$$Q(a,b,r) = \sum_k \left[m_k(t_k) - \left(a + be^{-rt_k}\right)\right]^2$$

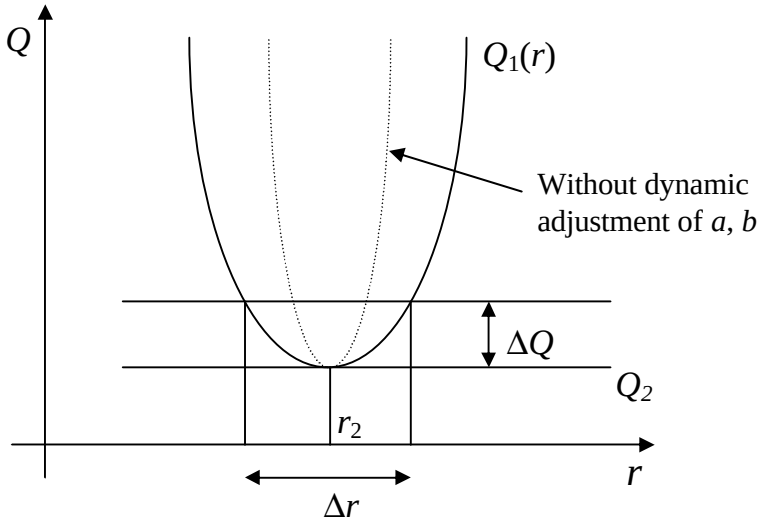
with respect to $a$, $b$ and $r$.

It is convenient to split the task into two distinct parts.

**1.** Assuming the value of $r$ to be fixed, the formula is linear with respect to $a$ and $b$. The optimal values of these two parameters are therefore easily determined using standard linear correlation formulae. The resulting 'optimal' values of $a_1$, $b_1$ and $Q_1$ are then non-linear functions of $r$, i.e., $a_1=a_1(r)$, $b_1=b_1(r)$ and $Q_1=Q_1(r)$, with $Q_1(r)$ being approximately quadratic around its absolute minimum.

**2.** Using the function which calculates $Q_1(r)$ for any value of $r$, its minimum value can be determined numerically using some standard method (in our case the Brendt's algorithm with simple interval bisection). Clearly, if the minimum of $Q_1(r)$ occurs at $r = r_2$ then $Q_2=Q_1(r_2)=Q(a_1(r_2),b_1(r_2),r_2)$ coincides with the 'absolute' minimum of $Q(a,b,r)$.

What we have gained is the possibility to evaluate the function $Q_1(r)$ for any $r$ and, in particular, in the vicinity of the optimum at $r_2$ where we expect it to be approximately quadratic with the quadratic coefficient related to the confidence interval of $r$. Notice that, along the curve $Q_1(r)=Q(a_1(r), b_1(r), r)$, the parameters $a$ and $b$ vary dynamically so as to remain optimal for every value of $r$. This is essential since otherwise the error estimates for $r$ would be grossly over-optimistic.




Numeric values of the *confidence interval* are based on the least significant increment of $Q$. Assuming that the optimum value $Q_2$ of $Q$ is due entirely to random experimental errors (this is false for non-exponential decays!), the least significant increment $\Delta_a Q$ can be determined for any given significance level $a$ by means of the Fisher statistic with both degrees of freedom set to $n-1$. The confidence interval $\Delta_a r$ for $r$ then comprises the $r$ values for which $Q_1(r) - Q_2 = \Delta_a Q$ and the *probable error* $e=\Delta r/2$ is obtained, as usual, by setting $\alpha=0.69...$ This may sound complicated but it actually turns out that, in the quadratic case, the result is excellently approximated by the simple formula:

$$e = \sqrt{\frac{Q_2}{(n-1)Q_1''(r_2)}}$$

where $Q_1''(r)$ is the second derivative of $Q_1(r)$ which is easily estimated numerically.

# Appendix –A

## 1) Parameter Definition file (parameter.def)

The Parameters.def file defines the structure of the parameters set pertinent to your installation. It is a crucial element of the NMR32 package, which must be present in the same directory as the executable AcqNMR.exe file. When absent, the program shall issue a warning and terminate before completing its initialization.

The file is installation-specific. Though most of the parameters are common to all Stelar instruments and MUST be present, some of their values and properties may be specific for the particular instrument in question. Moreover, there are some optional parameters whose very presence are installation-specific and indicate the presence of a particular hardware.

The file contains plain ASCII text and therefore can be edited by any suitable plain-text editor. However, since the information contained therein is critical (in the sense that unqualified modifications could preclude proper functioning of the system and even endanger some of its hardware), Stelar discourages Users from editing this file, unless they are quite experienced and clear, any intended modifications beforehand with the Company. Users shall be held responsible for any negative consequences of violating this rule.

**In any case, before you incorporate any changes, make a copy of the original file so that you can restore it if anything goes wrong.**

Syntax rules:

---

* The file contains only plain ASCII text.
* Within the file, the significant section starts with a line beginning with
  ->Parameters:
  and ends with a line beginning by '->', usually
  ->End
  Anything before and after such a section is a comment.
* Each parameter definition takes up one line.
* The order in which the parameters appear may be relevant in some contexts (e.g., Updating).

* Extra white space (blanks, tabulators, empty lines) is ignored.
* Lines starting with a vertical bar '|' are ignored (comment line)
* Each parameter definition line contains the following items, separated by '|' :

| Item | Description | Note |
|---|---|---|
| Acronym | Brief reference name | 1-4 characters (usually capital letters and digits) |
| \|Full name | Extended name displayed in tables | any string |
| \|Display page | Index of parameters display table | 1- 9; 0 means don't display |
| \|Location | Location index within the page | 1-18; 0 means don't display |
| \|MainLocation | Location index in Main Pars page | 1-18; 0 means don't display in the Main Pars page |
| \|Flags | A string of letters which may include: | |
| | A | User accessible |
| | I | Integer number |
| | F | Floating point (real) number |

| | |
|---|---|
| s | string value |
| S | string in CAPITAL letters only |
| B | MultiBlock parameter |
| C | may be computed through a formula |
| V | Admissible as a variable in expressions |
| D | affects Data |
| E | Evaluation parameter, save only in parameter files |
| H | Hidden, unless enabled in a sequence |
| P | rf Pulse |
| p | affects pulser timing |
| T | pulser Timing delay |
| U | Update upon loading (unless hidden) |
| W | Write into parameter and data files |
| O | accepts Options |

|Value Default initialization value Initial default values must be mutually coherent

|Domain type Describes the input constants type:RLLlDN/otherModifier letters

| | |
|---|---|
| R | Range |
| L | List |
| L1 | Implies that only the listed values can be accepted |
| D | Double list |
| N/other | No constraint |
| Modified letter | These are optionalallowed with RLD |
| u | Round to nearest upper admissible value (up) |
| d | Round to nearest lower admissible value (down) |
| n/other | Round to nearest admissible value (nearest) |

|Domain script Defines the input constraint values: Syntax depends upon Domain Type.
type R: <min.val>,<max.val>,<step> All items are optional.
type L: <val>,<val>,...,<val> While editing, the list is in the pop-down window

|Comment/Options Any comment or, for parameters with the O (accepts Options) flag, initial value of the Options field. This value shall appear in the Parameter Options field.

Notes:
- Unlike full names, many of the parameter acronyms are pre-assigned conventional meaning and used within the executable file. In other words, if you change, for example, the full name or location or default value of F1, nothing bad shall happen. If you change the acronym F1 to something else, however, offset shall stop working!
- Some of the data may be generated and set-up during initialization, depending upon the type of installed hardware. Thus, for example, the standard Stelar filters have a double-list type of input domain constraint but the actual domain script is computer-generated during the filter device initialization, based upon parameters listed in the Hardware.def file. This is the preferential way of how the specified hardware can adapt its related parameters.

Here is a typical parameter-definition line:
SF |System frequency |2| 3| 6|ADWUF |9.21 |R |0.001,120,0.001|[MHz] Larmor frequency rounded to .001 MHz

## 2) Hardware Definition file (hardware.def)

The installation-specific Hardware.def file defines the hardware structure of the instrument.
It is a crucial element of the NMR32 package which must be present in the same directory as the executable AcqNMR.exe file. When absent, the program shall issue a warning and terminate before completing its initialization.

The file contains plain ASCII text and therefore can be edited by any suitable plain-text editor.

However, since the information contained therein is critical (in the sense that unqualified modifications would almost certainly preclude proper functioning of the system and possibly endanger some of its hardware), Stelar strongly discourages Users from editing this file, unless they are Stelar-trained to do so and have discussed any intended modifications with the Company prior to applying them.

**Users shall be held directly responsible for any negative consequences of violating this rule.**

In any case, before you incorporate any changes, make a copy of the original file so that you can restore it if anything goes wrong.

Syntax rules:

---

* The file contains only plain ASCII text.
* Lines starting with '->???:', where ??? is a device name, contain device definition scripts.
  All other lines are ignored (comment lines).
* Each elementary device definition takes up one line of script.
* The definition scripts for all devices listed below are required, even when the corresponding device is not installed (see below)! Additional devices may be required, depending upon the type of the principal units.
* The order in which the device script lines appear may is irrelevant.
* Extra white space (blanks, tabulators, empty lines) is ignored.
* Each script line contains one to two fields separated by a vertical bar:
  The first field is the TypeScript, the second is the InterfaceScript.
* Within each field there is a number of items separated by commas.

TypeScript items:

1. Device type (a decadic number).
   The type denotes particular type (make) of the device.
   When type=0, the device is absent and the rest of the line is ignored.
2. The consecutive items are called Type Configuration parameters.
   They may be either decadic (e.g.,13) or hex (0xD) numbers which shall be loaded into a parameter's type-configuration array. Their interpretation is device- and type- dependent and is normally explained on a comment line within the Hardware.def file which follows the particular device definition script line.

InterfaceScript items (when present):

1. Name of the interface device driver (void if there is none).
   The driver is a device carrying out the actual interface actions and its definition script must appear among somewhere in the file.

2. Name of an interfaced parameter (void if there is none).
   A device may not have more than one interfaced parameter.
   To interface more than one parameter, one must define owned sub-devices, each associated with one of the parameters.
3. Interface code size (decadic or hex number).
   This item is significant only if there is an interfaced parameter
   It specifies how many bytes of code does the interface actually use.
4. The consecutive items are called Interface Configuration parameters.
   They may be either decadic (e.g.,13) or hex (0xD) numbers which get loaded into a parameter's interface-configuration array.
   Their interpretation is device- and type- dependent and is normally explained on a comment line within the Hardware.def file which follows the particular device definition script line.

As an example, consider the device ObserveFilter whose definition script is

```
->ObserveFilter: 3 (SPM),31,100,470,30000 | Sbus,FLTR,1,5,2
```

The TypeScript "3 (SPM),31,100,470,30000" indicates that it is of type 3 (the contents of the following parentheses are irrelevant). In this case, the following type-configuration parameters indicate that it has 31 steps per range and three linear ranges starting with cut-off frequencies of 100, 470, and 30000 Hz, respectively.

The InterfaceScript "Sbus,FLTR,1,5,2" indicates that it is set by means of the Sbus device, its associated parameter is FLTR, the interface is done by setting a single byte which is to be written at the Sbus address 5, index 2. Should you analyze the Hardware.def file, you would see that the InterfaceScript of Sbus ("Aqm") passes all interface requests to the Aqm device which passes them further to the Computer device which does the actual job.

Following is the list of devices which MUST be defined in the Hardware.def file of any Stelar NMR instrument. Normally, however, many other devices (subunits of the principal units) need also to be defined. Thus any type of Aqm assembly (Acquisition and Control Manager) is likely to contain devices like ->Pulser:, ->SweepGenerator:, ADC channels, etc. This, however, goes beyond the introductory nature of this Chapter.

->Instrument:
->Computer:
->ComputerLpt:
->Sbus:
->Aqm:
->ObserveTransmitter:
->ObserveReceiver:
->Magnet:
->Lock:
->Decoupler:

## 3) Pulse Sequence Script

Introduction:
This section describes the pulser sequence programming language employed by the STELAR NMR32 data acquisition program. Like in the case of macro-commands, it is a User- and Application-oriented fourth generation language (4GL). However, since the purposes of the two languages are quite different, their syntax differs so much that, unfortunately, any similarity between the two types of scripts stops here.
The definitions and descriptions which follow are comprehensive but very compact. The interested reader is invited to print out and study portions of the file DefaultSequences.ssf supplied with the NMR32 package. It contains all the pre-programmed library sequences, many of which can serve as examples.

A pulser sequence script consists of a contiguous segment of plain, case insensitive ASCII text. Its beginning is marked by a text line starting (apart from any white space) with the keyword SEQUENCE, followed by the name of the sequence. The sequence script terminates with a text line starting with the keyword END SEQUENCE. Individual pulser sequence segments thus appear as text blocks of the form

```
SEQUENCE SequenceName

....    'sequence-script body
        ....
        ....
        END SEQUENCE
```

(attention: unlike all the rest of the script, the keywords SEQUENCE and END SEQUENCE must be in CAPITAL letters).

Any number of pulser sequences may be grouped together into a single pulser sequences file. The only additional requirement on such files is that, in order to be recognized by the software, their first line must start with the discriminator string

```
STELAR Script File
```

and, preferably, their file-type extension should be "ssf" standing for "Stelar sequences file". Apart from the discriminator condition, there are no other limitations. Any text which does not appear within a SEQUENCE....END SEQUENCE text bracket is simply ignored - a fact which may be exploited conveniently for comments and/or descriptions of arbitrary extension.

Stelar supplies one pulser sequences file named DefaultSequences.ssf which should be located in the same directory as the executable file. The software, however, includes mechanisms for searching and selecting pulser sequences located in text files anywhere in the system (including a network). Users are encouraged to write their own pulser sequences and store them in their own pulser sequence files.

Basic Rules:

There are some simple rules which hold everywhere within the body of a pulse sequence script:

1. Extra white space is ignored.

Things like strings of blanks and/or tabulators, for example, have the same meaning as a single blank. A new line is recognized by the LF (line-feed) character. CR (carriage return) is treated as a blank and so are all other ASCII control characters.

2. Apostrophe (') starts a comment extending up to the end of the line.

3. Sequence header.

Pure comment lines immediately following the SEQUENCE declaration line constitute the sequence header which is displayed in the preview box in experiment/sequence selection dialogs. It may (and should) contain a brief description of the sequence and instructions for the final User on how to use it.

4. Empty lines are ignored.
The only exception to this rule is header termination since an empty line is sufficient to terminate the macro header.

5. The script is structured as a series of instructions or commands.

Longer commands are usually written one per line, in which case they do not require a special terminator (the line-feed itself is a valid terminator). Short commands, however, are often packed several per line in which case they must be terminated by semicolons.

Script Section:
The body of a pulser sequence script is divided into two sections: the sequence-proper section, followed by a setup section. The end of the sequence-proper section and start of the setup section is marked by the keyword #SETUP. The general structure of a pulser sequence script is therefore

```
SEQUENCE SequenceName
....
        ....    'sequence-proper section
        ....
        #SETUP
        ....
        ....    'setup section
        ....
        END SEQUENCE
```

The two sections are scanned by the software at different times and for different reasons. When a new experiment is specified (parameter EXP), the syntax of the corresponding pulser sequence script is checked (both section). Should an error be detected, the script is rejected and the EXP parameter is not allowed to change. Otherwise, the script is loaded into a RAM buffer and its setup section is scanned in order to initialize the values and/or access flags and option strings of any specified parameters.

The initial values and option string of the initialized parameters need not remain intact forever. If the parameter is User-accessible (or defined by means of a formula including other User-accessible parameters), its value may be changed at any time by the operator. At the moment of pulser programming, the actual, current value of each parameter must be used.

It is during pulser programming that the sequence-proper section is scanned by the software in order to generate the proper sequence of pulser steps.

Notice that, while the setup section is scanned only once (just after changing the parameter EXP), the sequence-proper section is scanned every time the pulser needs re-programming which happens quite often. However, this does not imply re-opening its source file since the sequence script is stored in RAM.

Setup section:
The setup section of a pulser sequence script is scanned just once when loading a new experiment (parameter EXP). The order in which its individual items appear is not particularly important (except in special cases). Each of its commands consists of

1. A parameter acronym, followed by the following optional items:
2. Parameter User-access configuration consisting of one of the following characters:

| | | |
|---|---|---|
| = | enable | User access |
| | : | disable User access |
| | space | don't modify |

3. Initial value to be set when the sequence is first loaded (remember that, subsequently, the value of the parameter may still change). A plain numeric value may be specified either directly as a number (any format) or as a string constant (e.g.,"23"). Values may be initialized also to expressions; in this case, they must be enclosed in parentheses and follow the expression-value syntax rules. String values must be "enclosed in quotation marks".

4. Initial settings of pulser parameter options. This item, if present, must be enclosed in square brackets. When the parameter is a pulser interval, the contents of the brackets are copied into its options field when the sequence is loaded (again, this does not preclude subsequent changes). When any of the optional items is missing, its current value is left unchanged.
In order to understand properly the initialization process, keep in mind that, upon loading a new pulser sequence but prior to any script-driven initialization, the system takes the following actions:

a) Resets number of blocks (parameter NBLK) to zero.
b) Clears option specifications of all pulser interval parameters.
c) Restores original display & access flags of all parameters according to the original specification in the file Parameters.def.
d) Hides all pulser interval parameters.
During the initialization process, the "hidden" flag of every encountered parameter (including the ones appearing within expressions) is automatically removed. The rule is that when a parameter is used, it should be displayed, regardless of whether it is User-accessible or not (unless, of course, its display location in the file Parameters.def is null).

The #SETUP section includes the parameter ENDS with the initial settings of the receiver phase cycle almost always.

Example of a #SETUP section:

```
D0 :(RD)
PW = 90 [p(x,-x,y,-y)]
ENDS [p(x,-x,y,-y)]
```

Sequence section:
The sequence section of a pulser sequence script is scanned every time the [virtual] pulser is programmed. The chronological order in this section's items is essential since it defines the sequence of individual pulser steps during the pulse sequence execution.

Each sequence section command is

1) either the acronym of a pulser interval parameter
2) or a special pulser sequence directive, preceded by the # character.

ad 1) A parameter acronym defines a pulser step whose duration and active pulser channels are defined by the current value of the specified parameter and by its options.
Within this general framework, however, some parameters are associated with special default actions listed in the following Table

| **Parameter** | **Action** |
|---|---|
| RF pulses (PW,P1,P2,...) | Turns ON the T-channel (transmitter) |
| D0, ACQD and STIM | Turns ON the R-channel (receiver) |
| D0 | Sets pulser into ARMED state (idle but with active output) |
| PW | Resets the time origin (automatic #TIME0 directive) |
| STIM | Triggers a sweep (S-channel ON for one step) |
| STIM | Sets XOFF (time-scale offset of the sweep) |

Note: Each sequence should start with the pre-scan delay D0 which should not appear elsewhere. The D0 settings are also effective while the instrument is in the "updated" but "idle" state.

Sequence Directives:
The pulser sequence directives may appear only within the sequence section of a pulser sequence script. They are distinguished from NMR parameters by being preceded by the # sign and interpreted by the system during pulser programming and used to implement special actions such as loops, pulser-controlled data-acquisition (ADC strobes), etc.
A related special feature are the pulser programming registers distinguished by the # sign, followed by a numeric index (#0, #1, #2, ..., #99). These can be assigned numeric values, operated upon and, more generally, appear as arguments in expressions (together with constants and numeric system parameters).
The following directives are currently implemented:

| **Directive** | **Action** |
|---|---|
| #<index> = <expression> | Assignment of a pulser programming register |
| #<label>: | Target location for the GOTO <label> directive.<label> is any user-defined string. |
| #GOTO <label> | Unconditional pulser program jump |
| #IF <relation> <command> | Conditional execution of a sequence <command>(a pulser interval parameter or a directive).<relation> is a construct of the form<expression><relation operator><expression>,where <relation operator> is one of <,=,>,<=,>=,<>,><. |
| #TIME0 | Sets the time-axis origin for subsequent data points |
| #ADC | Generates an ADC strobe. When at least one #ADC is present, the experiment type (EXPT) is set to 1(explicit data-points timing array) instead of 0 (regular timing grid). |
| #SWEEP | Generates a sweeper strobe. |
| #SETUP | Ends SEQUENCE section and starts SETUP section |

## 4) Working with the parameters DAAM and DAAP

The new parameter DAAM (Data Acquisition and Accumulation Mode) introduces the potentially rich topic of various data acquisition modes. At present it admits two possibilities:

| | |
|---|---|
| BASIC | which is the mode used up to now. |
| MUTE | which completely suppresses monitoring of data during acquisition, except for the scans counter which remains active. |

In both cases, the second new parameter DAAP (Data Acquisition and Accumulation Parameter) is used to control the rate at which the PC checks the progress of acquisition. Its value is in milliseconds and its recommended default value is 200 (allowed range is 10 to 2000).

The difference between the two modes lies in different timing when scan needs to be repeated very fast and/or the data blocks are large.

To understand the effect, consider the various situations which may occur:

### (i). BASIC mode

A. When the theoretical time interval between successive scan starts (repetition time) is larger than the software overhead related to accumulation and/or display-data transfer to PC, the time lost due to data processing is quite small since most of it is done in parallel with pulser & sweeper actions. The actual delay due to software in this case is roughly 18 ms per 1K of complex data points which is the time needed to move data from the ADC buffers to a core AQM memory buffer (we call the latter flash buffer). This is a plain transfer - all more sophisticated processing is done later.

For comparison, with the old AQM we were loosing about 120 ms/1Kpoints - always!
The new algorithm is much more efficient.

B. Since the full BASIC-mode processing cycle takes about 160 ms/Kpoints, things start changing when the theoretical repetition rate is smaller than this number. In this case, the processing cannot be completed in parallel and ends up delaying consecutive scan starts. The most unfavorable situation is reached when the theoretical repetition time is lower than 18ms/Kpoints. The processing then does not allow it to go below about 160ms/Kpoints - slower than the old AQM!

In case A, the parameter DAAP has no perceivable effect on overall acquisition timing.
Somewhat surprisingly, however, its effect is relatively modest even in case B. The reason is, of the 160 ms of processing, roughly 20 are taken up by the flash buffer transfer, 50 by data accumulation, 50 by refreshing a "scope image" on the AQM, and 50 by the actual transfer of the image data to the PC (all values except the last one are intended for 1 K of complex data points). Since the transfer to PC is asynchronous and has low priority, not all blocks are transferred so that last number must be proportionally reduced. The maximum average overhead then lies between about 150 and 170 ms/Kpoints and its variation with DAAP is relatively small.

In FFC one normally uses small block sizes. For example, with a block size of 64 points, the minimum overhead is about 18*(64/1024) = 1.125 ms/block, while the maximum is about 160*(64/1024) =10 ms/block. Since the recycling delay, polarization and relaxation intervals and

field switching times normally take up longer, all processing (except for the flash buffer transfer) is done in parallel and all the software overhead we have is just a bit over 1ms per block.

In classical NMR, however, situations may arise (not very often) when most of the time shall be spent in data processing by the AQM. In such cases the new AQM appears about 30-40% slower than the old one. These are the situations when to use the MUTE mode.

**(ii). MUTE mode**

In this mode, the "scope image" on the AQM is not maintained and no graphic data are communicated to the PC during acquisition. The User sees a blank screen and just the scan counter goes on at the rate given by DAAP. Once the accumulation has stopped, however, all the data are transferred to the PC and are displayed on the "scope".

This corresponds exactly to what we had with the old system. However, the maximum processing overhead is now only about 60 ms/1points - twice as fast as with the old device.

Moreover, due to the parallelism, it becomes effective only when the theoretical repetition time is lower than 60 ms - for larger repetition times the contribution is only 18 ms/1Kpoints.

This is the absolute minimum for the AQM board - and it has nothing to do with AQM to PC communication.

**Practical recommendations**

One does not need any particular speed while working "manually" - setting up parameters, checking on things, etc. It is only needed in long accumulations and/or in automations.

So prepare your acquisition using the BASIC mode and, if you need to go at maximum speed, switch to the MUTE mode only before launching the automated acquisition.

If what you plan is an acquisition controlled by a macro and generating lots of data zones, you can always write the macro in a way so as to display the data of the preceding zone while a new one is being acquired. Alternatively, if monitoring the input is what you want, you can always hook up a classical scope to the analog inputs.

## 5) Available pulse sequences and Marcos

There are several pulse sequences available in the "DefaultSequences.ssf" file. When you select a particular pulse (**EXP**) from the default file, a detailed description of that pulse is displayed on the window. However, here we list the pulse sequences available for the user.

**Pulse Sequences**

| | | |
|---|---|---|
| ANGLE.FFC | : | nutation angle determination. |
| PP | : | Simple pre-polarized sequence |
| PP/S | : | balanced, multi-block pre-polarized sequence |
| PPX/S | : | Balanced, multiblock, pre-polarized sequence without TAU-range defaults |
| NP | : | simple non-polarized sequence |
| NP/S | : | balanced, multi-block non-polarized sequence |

| | | |
|---|---|---|
| NPX/S | : | Balanced, multiblock, non-polarized sequence without TAU-range defaults |
| PP1E | : | Pre-Polarized sequence using an echo for signal detection |
| PP1E/S | : | balanced, multi-block version of PP1E |
| NP1E | : | Non-Polarized sequence using an echo for signal detection |
| NP1E/S | : | balanced, multi-block version of NP1E |
| IR | : | Simple Inversion Recovery sequence |
| IR/S | : | balanced, multi-block Inversion Recovery sequence |
| PPXRING | : | pre-polarized sequence with ringing suppression |
| PPXRING/S | : | balanced, multi-block version of PPXRING |
| NPXRING | : | non-polarized sequence with ringing suppression |
| NPXRING/S | : | balanced, multi-block version of NPXRING |
| IRXRING | : | inversion recovery sequence with ringing suppression |
| IRXRING/S | : | balanced, multi-block version of IRXRING |
| PPUB/S | : | unbalanced, multi-block pre-polarized sequence |
| NPUB/S | : | unbalanced, multi-block non-polarized sequence |
| PP1E | : | Pre-Polarized sequence using an echo for signal detection |
| PP1E/S | : | balanced, multiblock version of PP1E |
| NP1E | : | Non-Polarized sequence using an echo for signal detection |
| NP1E/S | : | balanced, multiblock version of NP1E |
| PPCPMG | : | Pre-Polarized sequence with CPMG signal detection |
| PPCPMG/S | : | multiblock version of PPCPMG |
| NPCPMG | : | Non-Polarized sequence with CPMG signal detection |
| NPCPMG/S | : | multiblock version of NPCPMG |
| IR1E | : | Inversion-Recovery sequence with echo detection |
| IR1E/S | : | multiblock version of IR1E |
| IRCPMG | : | Inversion Recovery sequence with CPMG detection |
| IRCPMG/S | : | multiblock version of IRCPMG |

**Macros**

| | | |
|---|---|---|
| Profile | : | to acquire a FFC NMRD profile using PP/S and NP/S sequences |
| ProfileX | : | to acquire a FFC NMRD profile using PPX/S and NPX/S sequences (with BINI,BEND,BGRD predefined by User). |
| Profile1E | : | to acquire a routine FFC NMRD profile using echo acquisition (PP1E/S and NP1E/S) |
| ProfileCPMG | : | to acquire NMRD profile using PP and NP sequences with CPMG signal detection. |
| ProfileIR | : | to acquire NMRD profile using IR sequence |
| ProfileIRCPMG | : | to acquire NMRD profile using IR sequence with CPMG signal detection |
| Evaluation | : | to evaluate T1 with user defined EWIP, EWEP, EWIB and EWEB on an already collected data file (Off-line evaluation). |
| InspectNmrPar | : | to inspect a NMR Parameter (including hidden parameters) |