

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *Oxford Instruments / Resonance Instruments Maran Ultra* at our website: *Click HERE*

RI NMR PROGRAMMING MANUAL (MARAN ULTRA)

RI NMR Programming Manual :

MARAN ULTRA HARDWARE v1.5 7/1/99 (software release 2.0).

Windows 95 is a registered tradename of Microsoft Inc.
Turbo Pascal is a registered tradename of Borland International Inc.

Copyright Resonance Instruments Ltd 1999.
This manual may be reproduced with permission.
Author : TBB.

Comments, errors and omissions should be reported to :

Product Support,
Resonance Instruments Ltd,
Unit 13, Thorney Leys Business Park,
Witney,
Oxfordshire,
OX8-7GE.
United Kingdom.
Tel. +44 (0) 1993 700442.
Fax. +44 (0) 1993 700363.

E-mail. support@resonance.co.uk

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

INTRODUCTION

The RI NMR Programming Manual (MARAN Ultra Hardware) provides information on :

i) Programming NMR pulse sequences using RI NMR.

ii) Programming macros for use with RI NMR.

A general overview of the RI NMR software, including information on loading and running pulse sequences and macros may be found in the RI NMR Users Manual. Users wishing to find information regarding supplied RI NMR pulse sequences and macros should refer to the RI NMR Pulse Sequence and Macro Manual.

Programmers should note that RI NMR may run on two different hardware types - and that the pulse programming language for each hardware type is different. The two types of hardware are :

i) MARAN Ultra Hardware

For benchtop MARAN Ultra NMR spectrometers, imaging and spectroscopy consoles.

ii) MARAN Hardware

For MARAN systems (pre-MARAN Ultra) running RI NMR.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

1. PULSE PROGRAMMING

1.0 Introduction

The set of procedures and functions which define the events and timing of an NMR experiment are known as a pulse sequence or pulse program. A typical pulse sequence specifies a number of RF pulses, phases, timing and acquisition delays. When the pulse program is compiled, the pulse sequence is loaded into the spectrometer hardware for execution.

1.1 Specifying the Default Editor and Compiling the Pulse Sequence

The default pulse sequence editor must be specified in the RI NMR software before pulse sequence editing and compilation can take place. The RI NMR Developers Kit is supplied with Borland Turbo Pascal 7.0 for pulse sequence compilation. Users should ensure that the correct path (usually c:\tp\bin\turbo.exe) and correct default pulse sequence directory (usually c:\program files\resonance\rinmr\seq\source) are specified under the Tools menu of the RI NMR software before attempting to write and compile pulse programs.

1.2 Pulse Program Skeleton

The pulse sequence must be defined as a Pascal procedure called Sequence. The following listing can be used as a skeleton and is stored on disk in the c:\program files\resonance\rinmr\seq\source directory :

```
PROGRAM MyPulseSequence;
{$I COMPILE.INC}

USES
{$I UNITS.INC}

VAR
{Define local variables here}

PROCEDURE Sequence;

BEGIN
{Enter pulse sequence here}
END;

BEGIN
Run(Sequence);
END.
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

1.3 Global Pulse Sequence Variables

A set of global, pre-defined variables exists for use in the pulse sequences. All the pre-defined variables may be specified using the RI NMR data acquisition software.

RI NMR PARAMETER NAME	INTENDED USE
P90	90 degree pulse length
P180	180 degree pulse length
SI	Size of acquisition buffer
TAU	Duration
DEAD1	Probe dead time
DEAD2	Filter settle time
RD	Experiment recycle delay
DW	Sample frequency
NS	Number of scans
PH x (x=1-5)	Phase program
D x (x=1-9)	Delays
P x (x=1-9)	Pulses
C x (x=1-5)	Counters
FW	Filter width
NEch	Number of echoes
G x (x=1-9)	Gradient amplitudes
GX	X gradient global scalar
GY	Y gradient global scalar
GZ	Z gradient global scalar
IG x (x=1-9)	Incrementing Gradients
SH x (x=1-5)	Shaped RF Event Identifiers
RFA x	RF Event global scalars
GSH x (x=1-5)	Shaped Gradient Event Identifiers

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

1.4 Resonance Instruments Pulse Programming Conventions

Pulse programs written by Resonance Instruments follow a convention to facilitate the operation of pulse sequences by users. It is recommended that pulse sequences written by programmers adhere to this convention.

Phase Programs

The first event (which is nearly always an RF pulse) is assigned phase list PH1.
The ADC phase list is defined as PH2.
The second event phase list is defined as PH3.
The third event phase list is defined as PH4 etc.

Delays

Tau is used as the time delay between consecutive RF pulse centres (for example in the CPMG pulse sequence).

In other pulse sequences where a duration is specified from the end of one RF pulse to the beginning of another, a Dn variable is used for the duration (for example the SOLID pulse sequence, CPMAS sequence).

RF Pulses

P90 is used where a 90 degree pulse is required.
P180 is used where a 180 degree pulse is required.

Gradients

The first gradient event is assigned amplitude G1.
The second gradient event is assigned amplitude G2 etc.

ZeroTime

Zerotime is defined as the start of the first RF event.

It should be noted that certain pulse sequences supplied with the instrument, such as DROPLET, do not adhere to this standard in order to preserve backwards compatibility.

1.5 Information Files

In order to facilitate the use of pulse sequences, programmers may define an information file which describes the parameters used in the pulse sequence and their function.

The information file should be a ASCII text file (created with Windows 95 Notepad or a similar text editor) comprised of a list of the parameters used in the sequence and their description. For example, the information for the pulse sequence such as FID might be as follows :

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
P90    90 Degree Pulse (us)
Dead1  Probe Dead Time (us)
Dead2  Receiver Dead Time (us)
SF     Spectrometer Frequency (MHz)
O1     Offset from SF (Hz)
FW     Filter Width (Hz)
DW     Dwell Time (us)
SI     Size of Acquisition Buffer (points)
NS     Number of Scans
RG     Receiver Gain (%)
RD     Relaxation Delay (us)
PH1    90 Degree Pulse Phase List (rec: 0213)
PH2    Receiver Phase List (rec: 0213)
DS     Dummy Scans
RFA0   RF Amplitude (%)
```

The information file should be placed in the C:\Program Files\RI NMR\Seq\Bin directory and given the extension .info. The information contained in the file will be displayed in RI NMR when the user views the acquisition parameters. Note that if no information file is created, RI NMR will use the default information file, default.info, when the acquisition parameters are displayed.

1.6 The Duration Procedure

Pulse programs are comprised of a series of Duration procedures. The Duration procedure takes two arguments. The first argument is a variable of type LONGINT or a constant which defines the length in time of the Duration procedure. The second argument (action argument/function) tells the hardware what actions to perform during the Duration procedure. For example :

```
Duration(10*US,0)
```

This command instructs the pulse sequencer to do nothing for 10 microseconds. In addition to the constant, a time multiplier (either US, MS or S) must be specified.

The pre-defined variables may specified as arguments to the Duration procedure, for example:

```
Duration(D1,0)
```

This command instructs the pulse sequencer to do nothing for D1 microseconds. D1 may be altered interactively from within RI NMR.

The different types of action arguments/functions that may be passed to the Duration command are listed in the following table :

Argument	Description	Example
0	Do nothing	Duration(D1,0)
RF(PHn)	Output on the first RF channel, use phase program PHn	Duration(P90,RF(PH1))

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

RF2(PHn)	Output on the second RF channel, use phase program PHn	Duration(P90,RF2(PH4))
FRF(PHn)	Output a fast RF pulse on the first RF channel, use phase program PHn	Duration(P90,FRF(PH3))
FRF2(PHn)	Output a fast RF pulse on the second RF channel, use phase program PHn	Duration(P90,FRF2(PH2))
ADC(PHn)	Acquire one pair of points (simultaneously), use phase program PHn	Duration(DW,ADC(PH2)
REC	Enable the receiver	Duration(DEAD2,REC)
TXEnable1	Enable the first RF channel	Duration(8*US,TXEnable1)
TXEnable2	Enable the second RF channel	Duration(8*US,TXEnable2)
CWn	Output an RF pulse on the CW channel, phase n.	Duration(P90,CW3)

More than one action argument/function may be passed using the addition operator, for example :

Duration(D1,REC+ADC)

More information on the use of specific action arguments is contained in sections 1.7-1.11.

1.7 Local Pulse Sequence Variables

Although it is often necessary to define local variables for use within the pulse sequence procedure (such as loop counters, calculated delays), only the global variables may be altered from within the RI NMR data acquisition software.

Local variables should be declared at the beginning of the pulse sequence procedure following the VAR declaration, for example :

```
PROGRAM MyPulseSequence;
{$I COMPILE.INC}

USES
{$I UNITS.INC}

VAR
  PreAcqTime : LONGINT

PreAcqTime:=(P90/2)-10*US;
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
PROCEDURE Sequence;

BEGIN
Duration(PreAcqTime,0);
END;

BEGIN
Run(Sequence);
END.
```

Programmers should note that constants used in expressions MUST have a time multiplier (either US, MS or S) associated with them.

1.8 Generating RF Pulses and Phase Cycling

RF pulses may be generated using the RF action function with the Duration procedure. The RF action function requires one argument which contains the phase list that the RF pulse will use. For example :

```
PROGRAM MyPulseSequence;
{$I COMPILE.INC}

USES
{$I UNITS.INC}

VAR

PROCEDURE Sequence;

BEGIN
  Duration(20*US,TXEnable1);

  Zerotime;

  Duration(1*US,RF(PH1)+TXEnable1);

  Duration(RD,0);

END;

BEGIN
Run(Sequence);
END.
```

The first Duration procedure takes the TXEnable1 action argument which enables the transmitter for 20 microseconds before the RF pulse is applied. The zerotime command defines the start time of the sequence for RI NMR - the data display will define time zero at this point when the XYZERO option is selected from the view menu of the RI NMR software (see the RI NMR Users Manual for more information).

The third Duration procedure outputs an RF pulse of length 1 microsecond of phase PH1 and then waits for RD microseconds. The entire program will execute NS times (the variable NS is set via

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

the RI NMR software).

The phase program PH1 is assigned in the RI NMR data acquisition software. PH1 may take the value 0,1,2 or 3, corresponding to each of the quadrature phases.

The final Duration procedure instructs the pulse program to do nothing for RD microseconds.

The programmer may increment the RF pulse phase by using the Next procedure, which takes one argument, the phase list to be incremented. For example, to phase cycle the RF pulse in the program listed earlier :

```
PROGRAM MyPulseSequence;
{$I COMPILE.INC}

USES
{$I UNITS.INC}

VAR

PROCEDURE Sequence;

BEGIN
  Duration(20*US,TXEnable1);
  Duration(2*US,RF(PH1));

  Next(PH1);

  Duration(RD,0);

END;

BEGIN
Run(Sequence);
END.
```

In the above listing the phase of the RF pulse will be changed according to the PH1 phase list every time a scan is executed.

Programmers familiar with the MARAN MS-DOS software should note the different handling of phase lists using RI NMR.

Programmers should note that two RF channels exist for spectroscopy sequences, RF and RF2. The receiver is locked to the first RF channel (RF).

1.9 Data Acquisition

Data are acquired by the spectrometer using a pair of analog to digital converters (ADC's). The fastest acquisition rate is dependent on the configuration of the hardware and is 0.1 microseconds (10 MHz) unless specified. Data are collected simultaneously in pairs. To acquire one pair of points the ADC action function is passed as an argument to the Duration procedure. The action function ADC also takes a phase list as a parameter, which specifies the acquisition phase in a

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

similar manner to the RF pulse phase. In addition to the ADC action function the receiver must be enabled by passing the REC action argument to the Duration procedure. The REC action argument must be specified before the ADC action function is used to allow the receiver to stabilise.

To accumulate a series of points a Pascal loop must be used :

```
PROGRAM MyPulseSequence;
{$I COMPILE.INC}

USES
{$I UNITS.INC}

VAR
  n :     LONGINT;

PROCEDURE Sequence;

BEGIN
  Duration(10*US,REC);

  For n:=1 to 512 DO
    Duration(2*US,ADC(PH2)+REC);

  Next(PH2);

  Duration(100*MS,0);

END;

BEGIN
Run(Sequence);
END.
```

The above program acquires 512 pairs of points 2 microseconds apart. Note that the receiver has to be enabled for 10 microseconds before the acquisition can take place and the local control variable n (a LONGINT) for the acquisition loop is defined at the start of the program. If more than one scan is performed, the data are accumulated according to the phase list defined by PH2.

1.10 A Simple FID Sequence

Enough of the features of the RI NMR pulse programming language have been described to allow a simple FID sequence to be written. The sequence is as follows :

```
Program FID;

USES{$UNITS.INC}

VAR
  n: LONGINT;

PROCEDURE Sequence;
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
BEGIN
  Duration(20*us,TXEnable1);

  ZeroTime;

  Duration(P90,RF(PH1)+TXEnable1);
  Duration(Dead1,0);
  Duration(Dead2,REC);
  FOR n:=1 TO SI DO
     Duration(DW,ADC(PH2)+REC);

  Next(PH1);
  Next(PH2);

  Duration(RD,0);

END;
```

1.11 Fast RF Pulse Generation

The pulse programming language has a facility for generating fast RF pulses. Fast RF pulses have two properties :

i) They can be used to switch phase instantaneously (useful for spectroscopy applications).

ii) They use less events in the pulse generator (useful for sequences in which large numbers of events are required, e.g CPMG sequences).

Fast RF pulses are generated using the FRF action function :

```
Duration(FRF(PH1));
```

1.12 Multi Channel RF Pulse Sequences

The MARAN Ultra Spectroscopy Hardware has a number of sophisticated features for high resolution spectroscopy applications. The following listing gives an example of a multi channel pulse sequence used for cross polarisation :

```
PROGRAM CPMAS;
{$I COMPILE.INC }

USES
{$I UNITS.INC }

VAR
  n : LONGINT;

PROCEDURE Sequence;

BEGIN
Duration(20*us,TXEnable1+TXEnable2);
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
ZeroTime;

Duration(P90,FRF2(PH1)+TXEnable1+TXEnable2);
Duration(D5,FRF(PH2)+TXEnable1+CW3);
Duration(Dead1,CW3);
Duration(Dead2,REC+CW3);

FOR n:=1 TO SI DO
  Duration(DW,ADC(PH3)+REC+CW3);

Next(PH1);
Next(PH2);
Next(PH3);

Duration(RD,0);

END;

BEGIN
Run(Sequence);
END.
```

The sequence for cross polarisation requires two RF channels, in this specific example one for the 13C excitation and one for the 1H excitation. A pulse sequence timing diagram can be seen in Figure 1.0.

The first Duration procedure enables both RF amplifiers. The second Duration procedure outputs a fast RF pulse on the second (1H) channel (FRF2). Note that both the RF amplifiers are enabled during the execution of this procedure and the RF argument is replaced by the FRF2 to enable fast phase switching.

The third Duration procedure activates the first (13C) channel for D5 microseconds and locks the second RF channel on with the CW3 action argument (for continuous wave operation, 3rd phase).

The fourth and fifth Duration procedures wait dead1 and dead2 microseconds respectively. The receiver is enabled during the dead2 duration. Programmers should note that the receiver channel is locked to the frequency of the first RF channel (FRF) and should also note that the second RF channel (FRF2) is on throughout the acquisition and dead time period.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

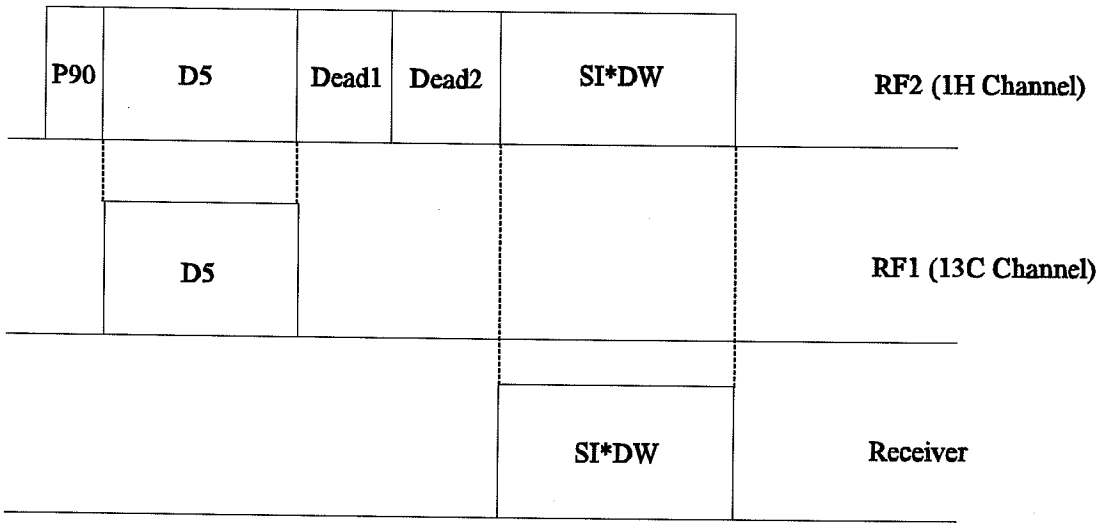


Figure 1.0 Pulse Sequence Timing Diagram for CP sequence.

1.13 Controlling Gradients

This section explains how to control the magnetic field gradient units. Programmers should note that not all MARAN Ultra Spectroscopy systems are supplied with magnetic field gradients.

All pulse programs which require gradients must specify the gradient unit under the USES section of the pulse program header:

```
Program GRADECHO;
{$COMPILE.INC}

USES
Grad,
{UNITS.INC}
```

Gradients are controlled using the Gradient procedure. This procedure takes a single argument which specifies which gradient channels to activate :

```
Gradient(GX(G1)+GY(G1)+GZ(G2))
```

RI NMR has two sets of parameters for controlling gradients. The first set G1-G9 allow the user to specify amplitudes for individual gradient pulses. The second set, GX, GY and GZ, are global channel scalars that multiply the output of all gradient pulses on each gradient channel by a specified amount.

The final amplitude output of a particular gradient channel is equal to the gradient pulse amplitude multiplied by the channel scalar. In the above example the amplitude of the output pulse will be G1*GX (and G1*GY, G1*GZ if specified). Note that all pulse amplitudes (G1-G9) are scaled by the current value of the channel scalar. Variables G1-G9 may take values between -32767 and 32768 and variables GX, GY and GZ may take values between 0 and +32767.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Once the Gradient procedure is called, the gradients are activated until turned off by another Gradient procedure :

```
Gradient(GX(OFF)+GY(OFF)+GZ(OFF))
```

To output a gradient pulse of 1 ms the following sequence of commands should be used :

```
Gradient(GX(G1)+GY(G1)+GZ(G1));
Duration(1*ms,0);
Gradient(GX(OFF)+GY(OFF)+GZ(OFF));
```

Programmers should note that due to the initialisation of the gradient hardware this series of statements produces a gradient pulse exactly 3 us longer than the 1 ms specified in the Duration procedure. In practice the shape and duration of the gradient pulse is dominated by the rise time of the gradient set / slew rate of the gradient amplifiers and the extra duration is insignificant. The extra time may be removed if required.

In the above example, the output on each of the three gradient channels is controlled using G1 from RI NMR. As mentioned earlier, the scalars GX, GY, GZ must also be set within RI NMR if gradients channels are to output correctly. For example if G1 is set to 100, GX is set to 32768, GY is set to 20000 and GZ is set to 0, only the X and Y gradient channels will output pulses.

An alternative pulse sequence could specify a separate amplitude for each gradient channel, using the following method :

```
Gradient(GX(G1)+GY(G2)+GZ(G3));
Duration (1*MS,0);
Gradient(GX(OFF)+GY(OFF)+GZ(OFF));
```

Now the X,Y and Z gradient channels can be controlled using the G1, G2 and G3 parameters. Using this technique GX, GY and GZ can be set to either 32767 or 0 to toggle between on and off gradients.

1.14 A Simple Gradient Echo Sequence

We are now in a position to define a simple gradient echo sequence for performing a 1D profile :

```
PROGRAM GRADECHO;
{$I COMPILE.INC}

USES
Grad,
{$UNITS.INC}

VAR
  n: LONGINT;

PROCEDURE Sequence;

BEGIN
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

```
Zerotime;

Duration(P90,RF(PH1));
Duration(Dead1,0);
Duration(Dead2,0);
Gradient(GX(G1)+GY(G1)+GZ(G1));
Duration(D1,0);
Gradient(GX(OFF)+GY(OFF)+GZ(OFF));
Duration(D2,0);
Gradient(GX(G2)+GY(G2)+GZ(G2));
Duration(D3,REC);

FOR n:=1 to SI DO
  Duration(DW,ADC(PH2)+REC);

Gradient(GX(OFF)+GY(OFF)+GZ(OFF);

Next(PH1);
Next(PH2);

Duration(RD,0);

END;

BEGIN
Run(Sequence);
END.
```

A timing diagram for the sequence may be found below. Note that the second gradient pulse should be negative for correct rephasing of the gradient echo. This is performed by specifying negative values for G1-G9 and GX, GY and GZ.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

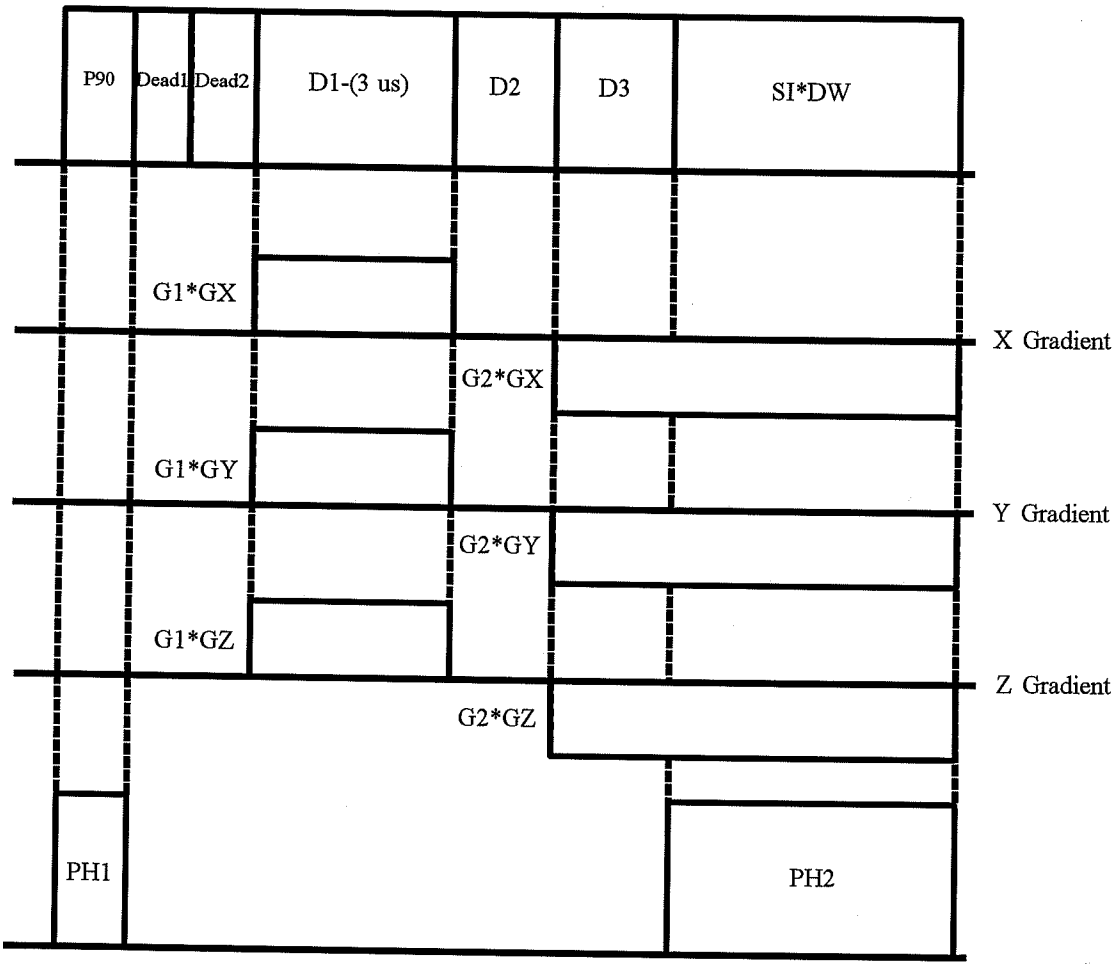


Figure 1.1 Timing Diagram for the GRADECHO Sequence

1.15 Incrementing Gradients

Incrementing gradients are necessary in order to conduct imaging sequences. An incrementing gradient may be specified by using the I prefix in the gradient procedure :

```
Gradient(IGX(G1));
```

G1 holds the current value of the incrementing gradient (note that this may not be accessed by the programmer). The number of times the gradient is to increment is held by the variable IG1 which is specified by the user within the RI NMR data acquisition software, along with GX which should be assigned the maximum value of the incrementing gradient. From the parameters IG1 and GX, the software automatically calculates the necessary gradient values for each phase encode step.

Thus the value of the gradient scalar is automatically divided into IG1 phase encode steps such that the IG1th/2 value will be zero. This removes the need for the user to specify a start gradient value and gradient increment. The user simply specifies the maximum value of the phase encode gradient through the gradient scalar and the number of phase encode steps required.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Gradients may be incremented using the NextGrad procedure :

```
NextGrad(IG1);
```

which increments the gradient IG1 to its next value. Sequences with incrementing gradients also require a loopback command to inform the hardware to return to the beginning of the pulse sequence and re-execute. This is performed using the LOOP procedure which uses the following syntax :

```
Loop(IG1);
```

which instructs the hardware to execute the pulse sequence IG1 times.

The following extract from a single point imaging pulse sequence shows how to use incrementing gradients as part of a pulse program :

```
Duration(10*US,0);
Gradient(IGX(G1)+TxEnable);
Duration(D1,TxEnable);
Duration(P90,RF(PH1)+TxEnable);
Duration(D2,0);
Duration(DW,ADC(PH2)+REC);
Gradient(IGX(OFF));
Duration(RD,0);
Next(PH1);
Next(PH2);
Loop(NS)
NextGrad(IG1);
Loop(IG1);
END;
```

The sequence starts by turning on the incrementing gradient IG1 and allowing the X gradient to stabilise for a period D1. The gradient is applied continuously during the 90 degree pulse and afterwards for a period D2, when a data point is acquired.

This process is repeated for NS scans for each value of incrementing gradient using the Loop(NS) procedure. Note that the Loop(NS) procedure is implicit in pulse sequences which do not contain incrementing gradients (such as the GRADECHO or FID pulse sequences). In pulse sequences which contain incrementing gradients the hardware requires information on when to loop for scans and when to loop for incrementing gradients. In pulse sequences with incrementing gradients the hardware has to be told explicitly when to loop for gradients and when to loop for scans.

Users should note that the above program is an example listing only. Additional procedures concerning data bin allocation are necessary in order for the program to function correctly.

1.16 Shaped RF Events

Shaped RF events may only be produced by MARAN Ultra systems equipped with special shaped RF event hardware.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

All pulse sequences using shaped RF events must specify the RFSHAPE unit under the USES section of the program header :

```
PROGRAM Shaped90;
{$I COMPILE.INC}

USES
RFShape,
{$I UNITS.INC}
```

Shaped RF events are generated in the pulse program using the ShapedRF procedure, which takes four arguments :

```
ShapedRF(Time, Phase, Identifier, Amplitude, Enable);

e.g ShapedRF(P1, PH1, SH1, RFA1, TxEnable1).
```

The arguments are as follows :

Time : The time for which the shaped RF event is generated. Note that according to the RI pulse programming convention the variables P1-P5 are used for shaped RF events. P90 and P180 are used for hard RF pulses.

Phase : The phase program for the shaped RF pulse.

Identifier : The identifier (which associates a shape file to the shaped RF event) to use.

Amplitude : An overall amplitude scalar for the shaped RF event. Note that RFA0 always corresponds to the amplitude of hard RF events generated with the RF action function. According to RI pulse programming convention, the first shaped RF event should be designated RFA1.

Enable : Enable the RF amplifier.

To produce a shaped RF event a shape file containing a description of the shape must be generated. The shape file should be an ASCII file containing two columns of numbers, the first column contains the amplitude of the shaped RF event and the second column contains the phase of the RF event. The numbers should and range from 0 to 1 for amplitude and 0 to 2π for phase. Once the file has been created it must be stored in the c:\Program Files\Resonance\RI NMR\shape directory.

WARNING : IF THE SHAPE FILE IS NOT PLACED IN THE CORRECT DIRECTORY IT WILL NOT LOAD.

The programmer must specify an identifier parameter (SH1-SH5) for each shaped RF event, which allows the user to associate a shape file with the event.

To identify a file with a particular shaped RF event within RI NMR type :

```
SHn filename (SH1 gauss256, in the above example)
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

at the command line.

In addition to the shape file, the overall amplitude of the shape may be scaled via the amplitude argument. The pre-defined global variables RFAn should be used for this purpose.

The phase program is handled in the usual manner. Finally, the Enable argument allows the relevant RF amplifier to be enabled (i.e either TXEnable1 or TXEnable2).

The following program provides an example of shaped RF pulses :

```
PROGRAM Shaped90;
{$I COMPILE.INC }

USES
  RFShape,
{$I UNITS.INC }

VAR
  n,i : LONGINT;

PROCEDURE Sequence;

BEGIN

ZeroTime;

Duration(10*us,0);
Duration(D1,TXEnable1);
ShapedRF(P1,PH1,SH1,RFA1,TXEnable1);
Duration(Dead1,0);
Duration(Dead2,0);
FOR n:=1 TO SI DO
  Duration(DW,ADC(PH2)+REC);
Duration(RD,0)
Next(PH1);
Next(PH2);
END;

BEGIN
Run(Sequence);
END.
```

1.17 Shaped Gradient Events

Shaped gradient events may be generated only by MARAN Ultra systems equipped with special shaped gradient event hardware. Programmers must specify the Grad unit in any pulse programs that use shaped gradient events. Shaped gradient events may be generated using the ShapeGrad function, which takes the four arguments :

```
Time:=ShapedGrad(Timebase, Identifier, Channel, Lines);
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The arguments are as follows :

Timebase : The duration per point in the shape file (example D1, 100*US).

Identifier: The identifier which associates a shape file with the shaped gradient event (example GSH1).

Channel : The gradient channel which will output the shaped event. Multiple channels may be specified using the + operator.

Lines : Any other control lines that are to be activated during the output of the event.

The ShapedGrad function returns the time for which the shaped gradient event occurs (number of points in gradient shape file multiplied by timebase).

The ShapedGrad function takes 18 us to execute (once the ShapeGrad function is called it takes 18 us until the shaped gradient is started).

To produce a shaped gradient event a shape file containing a description of the shape must be generated. The shape file should be an ASCII file containing a single column of numbers which describes the amplitude of the gradient event at a particular time. The numbers will be scaled automatically to use the full dynamic range of the hardware. Once the file has been created it must be stored in the c:\Program Files\Resonance\RI NMR\shape directory.

To associate a file with a particular shaped gradient event within RI NMR type :

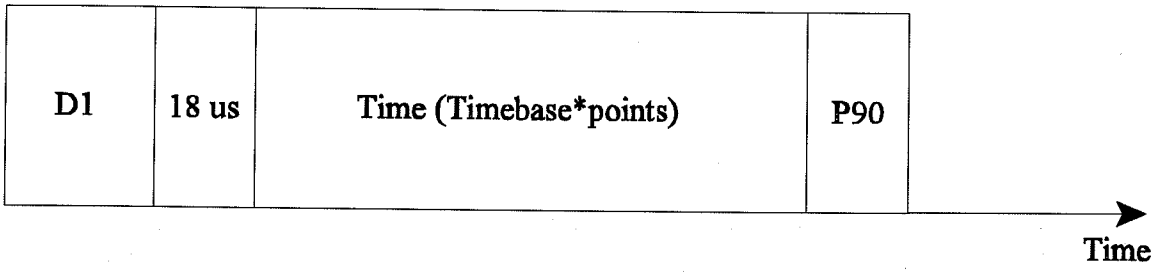
```
GSHn filename (for example GSH1 sinusoid)
```

at the command line.

Note the final output amplitude of the shaped gradient event will be proportional to GN (where n=X,Y,Z) * GX (where x=1-5) * value in Shape file.

The following programs provide examples of the use of shaped gradient pulses :

```
Duration(D1,0);
Time:=ShapedGrad(100*us,GSH1,GX(G1),0);
Duration(Time,0);
Duration(P90,RF(PH1));
```



Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

During the duration Time the shape described by GSH1 will be played out on the X channel.

```
Time=ShapedGrad(D1,GSH2,IGY(G1),0)
Duration(123*us,0);
Duration(1*us,ADC(PH2));
Duration(Time-((123+1)*us),0);
Duration(1*us,ADC(PH2));
..
..
NextGrad(G1);
```

The above program will play out the shape identified with GSH2 on the Y channel. Each point will last D1 microseconds. The amplitude of the gradient shape will change with each scan, from negative to positive, the number of incrementing steps is defined in RI NMR by IG1.

123 microseconds after the shaped gradient has started a single data point is acquired. Immediately after the shaped gradient event has finished a second point is acquired.

```
Time:=ShapedGrad(D1,GSH1,GX(G1)+GY(G2)+GZ(G3),0);
```

In the above case the same shape is played out on all three gradient channels simultaneously. The amplitude of each gradient (X,Y,Z) may be set differently.

1.18 Multi-Dimensional Data Acquisition and the Loop Command

The RI NMR pulse programming language has a flexible multi-dimensional data acquisition capability to cope with the demands of both imaging and 2D spectroscopy sequences.

To set up a multi-dimensional data acquisition buffer, the MultiAcquire procedure should be used. This takes the following syntax :

```
MultiAcquire(DIM1,DIM2,DIM3,DIM4,ACQUIRE);
```

where the DIM1,DIM2,DIM3,DIM4 are the dimensions of the data buffer and acquire is the name of a control procedure to be called after each DIM1 points are acquired.

The acquisition buffer should be set up using a separate procedure listed in the pulse program before the pulse sequence. For example :

```
Procedure SetAcquire;

BEGIN
MultiAcquire(SI,64,1,1,Acquire);
END.
```

In addition, any extra control variables that are required may be initialised in this procedure.

The above procedure initialises a multi dimensional acquisition buffer of dimensions SI * 64 * 1 * 1.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Every time DIM1 points are placed into the buffer the procedure Acquire (defined in a separate part of the pulse program) is called to allow the user to manipulate where in the data buffer the next set of data will be placed. The acquisition buffer is divided into DIM2*DIM3*DIM4 data bins, each of which holds DIM1 points. The data bin into which the next DIM1 points are placed is specified using the DataBin command which takes the syntax :

```
DataBin(Bin_number);
```

Where Bin_number is the number of the data bin where the next DIM1 points will be placed.

The following FID sequence illustrates the use of multidimensional acquisition buffers :

```
PROCEDURE Acquire;

BEGIN
IF ANS=NS THEN
  BEGIN
  ANS:=0;
  INC(Bin);
  END;
DataBin(Bin);
INC(ANS);
END;

Procedure SetAcquire;

BEGIN

MultiAcquire(SI,64,1,1,Acquire);
END.

Procedure Sequence;

BEGIN

ZeroTime;
      Duration(10*us,TXEnable1);
      Duration(P90,RFA(PH1)+TXEnable1);
      Duration(Dead1,0);
      Duration(Dead2,0);
      FOR n:=1 TO SI DO
        Duration(DW,ADC(PH2)+REC);
      Duration(RD,0);
      Next(PH1);
      Next(PH2);
      LOOP(NS);
      LOOP(64);
END;
```

The above sequence performs NS*64 scans. Before each SI points are acquired, the Acquire procedure is called. This procedure first checks to see if NS scans have been performed (if NS scans have not been performed we are in the middle of a phase cycle and the data should be added

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

to the current data bin according to the phase lists). The test is performed by comparing NS to a temporary variable ANS. If ANS is not yet equal to NS, ANS is incremented and the data bin is left unchanged.

If NS scans have been performed, the data bin is incremented using the DataBin command (the value of the current data bin is held in the global variable bin) and ANS is reset.

The two loop procedures instruct the hardware to return to the beginning of the pulse sequence at the end of each series of scans. The first LOOP procedure instructs the hardware to return NS times, the second 64 times, resulting in NS scans being performed for each of the 64 data bins. Note that for a multi-dimensional acquisition LOOP(NS) must be stated explicitly. For a 1 dimensional acquisition LOOP(NS) is implicit within the pulse sequence and does not need to be stated.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2. MACRO PROGRAMMING

2.0 Introduction

RI NMR incorporates a sophisticated macro writing facility for batch program execution. The RI NMR software includes many macros, including those for performing T1 measurements and automatically setting the NMR frequency. Users may write their own macros and compile them using Turbo Pascal. Macros may be loaded, edited and run from the Tools menu of RI NMR. Programmers should refer to the RI NMR Users Manual for information on loading, editing and running macros. The default macro source code directories and default macro editor (this should usually be the Borland Turbo Pascal editor) may be specified using the Options menu of RI NMR. Programmers should refer to the RI NMR Users Manual for more information. Programmers should note that the RI NMR macro language is based on Turbo Pascal and all valid Turbo Pascal commands may be used in RI NMR macros.

2.1 Macro Skeleton and Compiling Macros

A skeleton macro resides in the default macro directory (c:\program files\resonance\rinmr\macro\src)

```
PROGRAM Test;
{$M 8192,0,0 }

USES
  Macro;

BEGIN
END.
```

Note that any macro programs must link in the Macro unit. The {$M 8192,0,0} is a compiler directive and must be specified. Macros should be compiled using Turbo Pascal in a similar manner to pulse sequences.

2.2 List Files

Macros are usually written to perform a series of similar experiments in which one or more parameters are altered. An example is the calculation of a T1 using the inversion recovery method. Several experiments need to be performed with different values of inversion recovery times. The RI NMR Macro Language allows the programmer to read in values from a list contained in an ASCII file. The list should consist of rows of numbers separated by carriage returns. For example the file for an inversion recovery experiment could be :

```
10000
20000
30000
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Lists may either be edited/created using the RI NMR List Editor (see the RI NMR users manual) or with any text editor. Lists should have the extension .lst and should be stored in the c:\program files\resonance\rinmr\list directory.

2.3 Macro Variables

Several pre-defined variables exist for use with macro programs. These are :

F1-F5 - command line variables.

Command line variables are specified when the macro is executed. For example, when the macro test is executed at the RI NMR command line with the following syntax :

test DATAIN DATAOUT

Then in the macro program test F1:='DATAIN' and F2:='DATAOUT'. Up to five command line arguments may be specified. Programmers should note that the command line arguments are pre-defined in the macro unit and are of the type STRING. Command line variables may only be used to specify filenames (input and output) and information regarding command line variables must be provided at the start of the macro (see section 2.7 for more information).

I1-I5 - indices

Indices point to an item in a list. A value of 1 specifys the first item in the list, a value of two the second item in the list etc. Five indices I1-I5 are pre-defined in the macro unit and are of type INTEGER.

2.4 Macro Commands

A series of macro specific commands exists. These are described as follows :

Command(Command: STRING)

Exits an RI NMR command, for example COMMAND('WR') writes the current data set to disk. The Command command may be used with several arguments, for example Command('WR '+F3+' Y') where F3:='MYFILE' is identical to typing :

WR MYFILE Y

at the RI NMR command line. Programmers should note that interpreter commands exist as well as the more usual commands that are entered via the command line. Interpreter commands are commands that are normally executed using the mouse in the windows environment (an example is the switch between acquisition and process modes). Interpreter commands are preceded by a ~. A full list of interpreter and command line commands may be found in the Appendix.

List (Filename : STRING,Index : INTEGER AssignmentList :STRING)

The list command allows the user to read in values from a list. Filename (this can be a command

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

line variable), index (an index variable) and assignment list specifies which acquisition parameters the value of the list are read into. For example if the list 'example list.lst' exists in the c:\program files\resonance\rinmr\list directory and takes the form :

10
20
30

The List command :

List('example list',3,'D1');

will assign the third item in the list to the acquisition parameter D1. Users creating their own list s (rather than with the RI NMR list editor) should note that the default pathname for the list command is c:\program files\resonance\rinmr\list). All lists must have the file extension .lst.

Ext (Index : INTEGER)

It is often necessary in macros to save a large number of data files to disk. The Ext takes an index as a single argument and prepares it for attachment to a file. The command Ext is passed to Command as an argument. For example :

```
I1:=10;
Command('WR '+'myfile'+Ext(I1)+' Y');
```

automatically attaches the extension .10 to the written data. Thus sequentially labelled files can be written as I1 is incremented.

Inc (Index : INTEGER)

The Inc command increments an index so that it points to the next item in the list. For example using the list 'example list' given above :

```
Command('~AMODE');
Command('Load FID');
I1:=1;
List('example list',I1,'D1');
Command('GO');
Command('WR '+'Output'+Ext(I1)+' Y');
Inc(I1);
List('example list',I1,'D1');
Command('GO');
Command('WR '+'Output'+Ext(I1)+' Y');
Command('~PMODE');
```

This macro selects acquisition mode, loads in the FID pulse sequence and reads in the first value from the list 'example list'. D1 is set to this value and the Pulse sequence is executed. After the pulse sequence has executed, the next command writes the acquired data to disk. Note the use of the Ext argument to add the index I1 as an extension to the filename. After this is repeated, I1 is incremented (increased to 2) and the next item on the list is read into D1. The second FID

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

is saved with the extension .2.

2.5 Additional Commands in RI NMR

RI NMR has a number of in built commands designed specifically for use in macros. These are:

```
Message (CANCEL) MessageText
```

Prints MessageText in a dialog box on screen and halts the macro until the user clicks on the 'OK' button. Addition of the word CANCEL brings up a 'Cancel' button which can be used to halt the macro.

Example :

```
Command('Message CANCEL WELCOME TO THE RI NMR OILSEEDS MACRO.
CLICK ON OK TO PROCEED, CANCEL TO EXIT');
```

```
Wait Seconds
```

Causes RI NMR to pause for seconds number of seconds.

Example :

```
Command('Wait 100');
```

2.6 Using Command Line Arguments

Macros must be informed of expected command line arguments. This allows the macro to prompt for command line arguments if they are not specified on the command line.

The command line argument information must occur at the beginning of the pulse sequence, before any other instructions take place. Two commands are used to specify filename information :

```
Infile('Browser Title','File Extension');
```

```
Outfile('Browser Title','File Extension');
```

Infile is used to specify input files (such as lists used by the macro), Outfile is used to describe output files (such as data files produced by the macro). Browser title is the title of the Windows 95 file browser that is displayed if the command line argument is not specified and file extension is the file extension of current files that will be displayed.

The command line argument information must be ordered in the same sequence as the command line arguments are expected, for example if one input file and two output files are required :

```
Infile('Diffusion List','List');
Outfile('Data File','RIDAT');
Outfile('Processed Data File','RIDAT');
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.7 Using Logfiles

A log file can be used to log all output that is printed in RI NMR command history box to an ASCII text file.

Log files are created using the following syntax :

LOG filename (Y) (A)

This creates a log file called filename.00001.RiLog in the current logfile directory. The Y option forces the software to overwrite any existing logfile of the same name.

The A option instructs the software to append the information to an existing file rather than create a new file.

Once the log command is executed all output to the RI NMR command history box is automatically logged to the text file.

After each line is output a line feed takes place. For example if a log file is turned on and the user types D3 <RETURN> followed by SIZE the following output will be written to the log file :

23
1234.126

Where 23 is the current value of D3 and SIZE is the value returned by the SIZE command. To suppress the line feed an underscore (_) prefix should be used. _D3 followed by SIZE produces :

23 1234.126

To turn of logging to the file the END command should be used.

The following extracts from the getdiff macro (the source code is supplied with RI NMR) illustrates the use of log files :

```
Command('~PMODE');                    {Select Process Mode}
Command('LOG DIFFLOG Y');             {Create Empty Log File}
Command('END');                       {Close Log File}
```

The above commands create a log file called difflog.00001.RiLog which overwrites any existing file of the same name in the current log file directory.

```
Command('LOG DIFFLOG A');             {Open Log File}
Command('_D3');                       {Log Without Line Feed}
Command('SIZE 20');                   {Get Size Of Echo}
Command('END');                       {Close Log File}
```

The above commands open the log file for appending. The underscore (_) writes the current value

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

of D3 to the log file without a line feed, which ensures that the result of the SIZE command is written to the same line. This produces a two column log file of D3 vs. SIZE.

```
Command('XY FILE '+LogDir+'DIFFLOG.00001.RILOG');

                                        {Replot New Log File}
```

The final command uses the XYPLOT facility (see the RI NMR users manual) to plot the log file on screen to allow the user to monitor the progress of the diffusion experiment.

2.8 Example Macro - Inversion Recovery Sequence

The following macro facilitates the measurement of T1 by reading in several inversion recovery times and performing inversion recovery sequences. The listing may be found in the c:\program files\resonance\rinmr\macro\src directory.

```
PROGRAM T1;
{$M 8192,0,0 }

USES
  Macro;

BEGIN
InFile('T1 LIST','LIST');
OutFile('Data Files','RIDAT');

Command('~AMODE');
Command('LOAD InvRec');
I1:=1;
WHILE List(F1,I1,'D1') DO
  BEGIN
  Command('GO');
  Command('~PMODE');
  Command('WR'+F2+Ext(I1)+' Y');
  Command('~AMODE');
  INC(I1);
  END;
Command('~PMODE');
END.
```

The macro takes two command line arguments, the name of the list to be used (F1) and the prefix of the data files to be produced (F2).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3. VISUAL BASIC SCRIPTING

3.0 Introduction

In addition to the simple macro language described in Chapter 2, users may also control RI NMR via Microsoft Visual Basic scripting. Users may also use Microsoft Visual Basic scripting to control NMR data acquisition via alternative software (for example Microsoft Excel)

Microsoft Visual Basic scripting is supported in all RI NMR versions greater than version 2.0, and requires either Windows 98 or a Windows 95 software upgrade to function. Users should note that no further developments to the macro language will occur after RI NMR version 2.0.

It is not the purpose of this documentation to describe the Microsoft Visual Basic programming language, nor teach the user how to program in Microsoft Visual Basic. Users should refer to Microsoft Visual Basic programming manuals/ Microsoft on line documentation for this information (http://msdn.microsoft.com/scripting/vbscript/).

3.1 Installation

WARNING : MICROSOFT VISUAL BASIC SCRIPTING WILL ONLY FUNCTION ON RI NMR VERSIONS 2.0 OR GREATER. USERS WISHING TO UPGRADE TO MICROSOFT VISUAL BASIC SCRIPTING MUST CONTACT RESONANCE INSTRUMENTS FOR A SOFTWARE UPGRADE.

Users wishing to run Microsoft Visual Basic scripting on Windows 95 PC's must install additional components. These are provided by Resonance Instruments on the RI NMR Scripting Binaries disk. When a scripting capable version of RI NMR is installed (version 2.0 or later) the installation software will automatically prompt the user for a scripting binaries disk where necessary.

Alternatively, users wishing to install Microsoft Visual Basic scripting without re-installing RI NMR should place the RI NMR Scripting Binaries disk in the floppy drive and run the a:\setup.exe program from the Windows 95 Start Menu.

WARNING : IF IN DOUBT ABOUT THE INSTALLATION PROCEDURE PLEASE CONTACT RESONANCE INSTRUMENTS FOR ADVICE BEFORE INSTALLING MICROSOFT VISUAL BASIC COMPONENTS ON WINDOWS 95 SYSTEMS.

3.2 The RI NMR Script Editor

The RI NMR script editor may be activated by selecting Tools-Scripts from the RI NMR main menu.

The script editor allows users to edit and run scripts. All scripts present in the default scripts directory (usually c:\Program Files\Resonance\RINMR\scripts) are displayed. Users may switch between scripts by clicking on the appropriate tab at the top of the editor window. Note that all

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

new scripts must be saved in the \scripts (usually found in c:\Program Files\Resonance\RINMR) directory and be assigned the file extension .ris.

Scripts may be run from the editor by pressing the arrow key on the right of the toolbar.

Note that scripts may be run from the RI NMR command line by typing .Scriptname, in a similar way to running macros.

Users should note that RINMR searches for scripts before it searches for macros. This means that if a script and a macro exist with the same name the script will be executed, NOT the macro (for example .AUTOO1). To disable the scripting, users should set USESCRIPTING parameter in the RINMR.ini file (this can usually be found in the c:\Program Files\RI NMR\bin directory) to NO.

3.3 The NMR Object Description

An object is a related group of Visual Basic functions, subroutines and properties (which are collectively referred to as members of the object) . Resonance Instruments provides an object called nmr for controlling the RINMR data acquisition software through a Visual Basic script.

The following list describes the members of the nmr object :

General Members

nmr.Execute(command)

Sends the string command to the NMR object to be executed as a command line parameter. This subroutine should be used to execute any RI NMR command except the GO command.

Type subroutine
Parameters command (string)
Example `nmr.Execute("MAG")`

nmr.GO

Executes the RINMR GO command (starts a data acquisition). Control does not return to the script until all the scans are complete. If the alternative nmr.Execute("GO") is used, control returns to the script immediately after the scans started.

Type subroutine
Parameters none
Example `nmr.GO`

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

nmr.ReturnValue

RINMR commands which return a value set a property on completion. The nmr.ReturnValue property allows the value to be retrieved by the script.

Type read property (string)
Parameters none
Example
```
nmr.Execute("SIZE")
Datasize=nmr.ReturnValue
```

nmr.GetParameter(paramname)

Returns the value of an RINMR parameter.

Type function (string)
Parameters paramname (string)
Example
```
p90=nmr.getparameter("p90")
```

nmr.Minimize

Causes RINMR to be minimized to the Windows 95 taskbar. Use in conjunction with nmr.Restore

Type subroutine
Example
```
nmr.restore
```

nmr.StatusMessage(message)

Causes the specified string message to be displayed in the RINMR command box, allowing feedback from the script to the user.

Type subroutine
Parameters message (string)
Example
```
nmr.StatusMessage("Setting P90 to Start value")
```

Data Transfer Members

These members are used to read values from the data channels from RINMR back into the script. Three data channels exist, A, B and T, containing the channel A and B quadrature data and time values respectively. Single data points can be accessed using GetAChannel/GetBChannelData, or the entire array can be read using the Data, DataB and DataT properties.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

nmr.GetAChannelData(datapoint)
nmr.GetBChannelData(datapoint)

Return the current value of the indicated channel at the index corresponding to the integer datapoint..

Type function (single)
Parameters datapoint (integer)
Example
```
FirstADataPoint=nmr.GetAChannelData(0)
```

nmr.DataA
nmr.DataB
nmr.DataT

Uses a variant array to transfer an entire data channel.

Type read property (variant array)
Parameters None
Example
```
Dim A, B as variant
Dim MagFirstPoint as double
A=nmr.DataA
B=nmr.DataB
Mag1stPoint=sqr(A(0)*A(0)+B(0)*B(0))
```

Dialog Box / Form Members

Scripts can be programmed to display a dialog box which contains a form for user input at run time. The layout of the form may be defined by the programmer from within the script using the following members. The script should first call the required form creation members to set up the form, then call the nmr.ShowForm function. When control returns from the nmr.ShowForm function, the users input may be retrieved.

nmr.GeneralEdit(name, caption)

Creates an edit box for user input, along with the associated caption. When control returns following the nmr.ShowForm function (see below), the nmr.GetEditValue function may be used to retrieve input from the box. The edit box and its caption are positioned automatically by RINMR. To set the position of an edit box manually use the nmr.GeneralEditXY procedure.

Type subroutine
Parameters name (string), caption (string)
Example
```
nmr.GeneralEdit ("edit RG","Enter Starting Value")
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

nmr.GeneralEditXY(name,caption,x,y)

Creates an edit box for user input, at the position of the form given by the pixel co-ordinates integers X and Y. Two strings must be specified for the name and caption of the edit box.

Type subroutine
Parameters name (string), caption (string), x (integer), y (integer)
Example
```
nmr.GeneralEdit("editRG","Enter Starting Value",20,200)
```

nmr.SetEditValue(name,value)

Sets the initial value of edit box specified by the name string to the value contained in the value string.

Type subroutine
Parameters name (string), value (string)
Example
```
nmr.SetEditValue("editrg","20")
```

nmr.SetEditCaption(name,caption)

Allows the caption associated with an edit box to be changed after the edit box has been created.

Type subroutine
Parameters name (string), caption (string)

nmr.ParamEdit(name)

Creates an edit box with additional features designed for NMR parameter input. On creation of the edit box the edit text will automatically be set to the current value of the parameter specified by the string name. If the user closes the form using the OK button, the NMR parameter is automatically updated. If CANCEL is pressed, no update is performed.

Type subroutine
Parameters name (string)
Example
```
nmr.ParamEdit("P180")
```

nmr.ParamEditXY(name,x,y)

Allows the programmer to create and position an NMR parameter edit box.

Type subroutine
Parameters name (string), x (integer), y (integer)
Example
```
nmr.ParamEdit("P180",20,150)
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

nmr.GetEditValue(name)

Returns the current edit text for the edit box labelled by the string name. This function is used to retrieve user input after the nmr.ShowForm function has been called.

Type function (string)
Parameters name (string)

nmr.CreateCheckBox(name,caption)

This subroutine creates a standard Windows 95 checkbox control to enable a user to make a YES/NO choice. To set the default value of the checkbox use the nmr.SetCheckValue subroutine.

Type subroutine
Parameters name (string), caption (string)

nmr.SetCheckValue(name,value)

Allows the script to check/uncheck a previously created checkbox, to indicate a default or currently selected value. Value should be an integer, either 1 (checked) or 0 (unchecked).

Type subroutine
Parameters name (string), value (integer)
Example
```
nmr.SetCheckValue("checkbox1",1)
```

nmr.GetCheckValue(name)

After nmr.ShowForm, nmr.GetCheckValue returns a 0 or 1 corresponding to the status of the checkbox labelled by the string name.

Type function (integer)
Parameters name (string)
Example
```
boxchecked=nmr.GetCheckValue("checked")
```

nmr.CreateRadioGroup(caption)

A single radio group may be created per form to allow the user to choose between a number of mutually exclusive options. Use this function to create an empty radio group then use the nmr.CreateRadioOption to populate the group.

Type subroutine
Parameters caption (string)
Example
```
nmr.CreateRadioGroup("Choose a start value for P90")
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

nmr.CreateRadioOption(name)

Allows the radio group created by nmr.CreateRadioGroup to be populated with options.

Type subroutine
Parameters name (string)
Example nmr.CreateRadioOption("Option 1")

nmr.RadioOption

This property corresponds to the currently selected radio option. The array of options is zero based, the first option in the group corresponding to a radio option of 0. nmr.RadioOption is a read write property and can also be used to set the current value of the radio group.

Type read/write property (integer)
Examples
```
optionchosen=nmr.RadioOption
Nmr.RadioOption=0
```

nmr.RadioOptionName

Equivalent to nmr.RadioOption, but returning the caption corresponding to the chosen radio option rather than its order in the list. This is a read only property and cannot be used to set the current value of the radio option group.

Type read property (string)
Example
```
chosenname=nmr.RadioOptionName
```

nmr.DialogText

Adds a text string to the window containing the form for additional user information.

Type write property (string)
Example
```
nmr.DialogText="Enter Required Values"
```

nmr.ShowForm

Displays the form. OK and CANCEL buttons are automatically created and placed on the bottom of the form. This function does not return until the user closes the form, the return value indicating if OK (return value=1) or CANCEL (return value=0) was pressed.

Type function (integer)
Example
```
if nmr.ShowForm= TRUE then
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

nmr.ClearAll

This procedure clears all the script created objects from the form. It is advisable to use nmr.ClearAll after finishing with a scripted form (after all the required input from the form has been retrieved) to both free memory and prevent name conflicts with future scripts. For example if a second script tries to create an edit box with a name that already exists and nmr.ClearAll has not been cancelled an error will occur.

Type subroutine

3.4 Special Variables

A number of special variables have been added to RINMR for use with Microsoft Visual Basic scripting. These parameters are read only and are identified with a '%'.

%TEMPDIR	Current temporary directory.
%DATADIR	Current data directory.
%EXPORTDIR	Current export directory.
%ROOTDIR	Current root directory.
%LOGFILE	Current log file.
%LISTFILE	Current list file.
%HARDWARE	0=None; 1=Internal (MARAN); 2=Network (MARAN ULTRA).
%SEQFILE	Current pulse sequence.

3.5 The RINMR Standard Library

A number of Microsoft Visual Basic routines have been written and stored in a standard library (RIStdLib.ril).

Users may create their own libraries. Library files should be placed in the \scripts\lib directory and have the file extension .RiL.

DeleteFile(Filename)

Deletes the file specified by the string filename. Note the full pathname of the file must be specified. Wildcards are allowed.

Show(Message)

Displays a message in a dialog box. The script pauses until the user presses the OK button.

WhiteSpace(c)

Returns TRUE if c is a whitespace character. Whitespace characters are TAB, SPACE, CARRIAGE_RETURN or LINEFEED.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

LoadList1(name)

Loads the list file specified by the string name into List1. List files are ASCII text files containing a list of entries. Each entry is separated from the next entry by one or more whitespace characters. Note the full pathname of the list must be specified.

LoadList2(name), LoadList3(name)

See Above.

NextList1(x)

The next value from list1 is read into the variable x. The function returns false when no more entries exist in the list.

NextList2(x), NextList3(x)

See Above.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP