An automated pulse programmer for NMR experiments

J. Phys. E: Sci. Instrum. **20** (1987). Printed in the UK

An automated pulse programmer for NMR experiments

D W Dubro, T H Nuij and J M Pope
School of Physics, The University of New South Wales, PO Box 1, Kensington, NSW, Australia 2033

Received 24 December 1985, in final form 9 June 1986

Abstract. A new pulse programmer designed and built for an NMR spectrometer is described. Constructed from TTL circuits with 2 kbyte of on-board memory, the pulse programmer provides 100 ns pulse width resolution with a 10 MHz clock. It is specially designed to be interfaced to any computer with parallel I/O lines. Output consists of four independent channels, four trigger lines and a dedicated acquisition line. Operation is independent of the host computer to ensure precise timing.

In conjunction with the hardware, a complete software system, with a number of innovations, has been written in FORTRAN to control the pulse programmer and manage the NMR experiment.

1. Introduction

The continued rapid development of pulsed NMR as a technique for detailed studies of condensed matter places increasing demand on pulse programmers and data acquisition systems. Recent developments, which have led to new pulse sequences and consequently new demands on pulse programmer performance, have included 2D NMR methods, multiple quantum NMR and magnetic resonance imaging.

A number of pulse programmers have appeared in the literature (see, e.g., Doherty 1982, Gianotti *et al* 1980, Saint-Jalmes and Barjhoux 1982, Shenoy *et al* 1976, Taylor *et al* 1974) which consist of a series of counters or registers and provide a simple multipulse sequence. However, such pulse programmers are likely to prove inadequate for the ever increasing number of complex pulse sequences now employed, ranging from the phase cycling of standard sequences to 'binomial' excitation and pulse combs used in selective excitation and solvent suppression (Hore 1983, Morris and Freeman 1978), to multi-pulse line narrowing sequencies (Mehring 1983) and to 'depth' pulses employed in spatially resolved spectroscopy (Bendall and Pegg 1985, Pope and Eberl 1986, Shaka *et al* 1985).

To achieve longer pulse sequences, pulse programmers have been designed with their own on-board memory (see, e.g., Adduci and Gerstein 1979, Caron and Herzog 1978, Cosgrove *et al* 1980) using TTL circuitry to achieve the maximum speed. As pulse programmers become more complex, microprocessors can be used to simplify the interaction of the pulse programmer with the user (Adduci and Gerstein 1979, Pratt and Ackerman 1980). Adduci and Gerstein (1979) used a microprocessor as the basis for a stand-alone pulse programmer which had a pulse program editor plus ROM for storage of commonly used pulse programs. Such a system is not necessary if the pulse programmer is interfaced directly to a computer.

2. Description of the new pulse programmer

All the pulse programmer designs we have examined had limitations which we felt were unnecessary. Some were not completely under software control; some were limited in capacity and flexibility and others (Dart *et al* 1980, von Os and Veeman 1979) required special computer interfaces.

A new pulse programmer design is described herein which is wire wrapped from standard TTL circuits. It has its own on-board memory capable of holding 2048 steps. This size is adequate to provide extremely complicated pulse sequences without looping. Program flow inside the pulse programmer is determined by instructions stored in this memory and decoding is done by the hardware in one clock cycle.

The pulse programmer was interfaced to a computer by means of (5 × 8) parallel I/O lines so that it is easily disconnected and does not depend upon one particular bus system. It can be attached to any computer with a minimum of eight parallel lines. Programming the pulse programmer is carried out with a set of four commands and software is easily written in any high level language to control it.

A block diagram of the pulse programmer is shown in figure 1. It consists of: A, a 32-bit down counter; B, output control logic; C, memory (48 bits wide by 2 K long); D, program counter; E, interface control logic.

The pulse programmer recognises four different types of instructions: 1, time; 2, acquire; 3, stop; 4, repeat; which are stored in the high eight bits of memory.

The time instruction loads the output control logic with the proper values for the duration of counting by the 32-bit counter. This instruction offers the possibility of selecting four output channels and four trigger channels in any combination. A delay is simply generated by not selecting any of the output channels. Operating with a 10 MHz clock (either internal or external) and the 32-bit counter, the pulse programmer is capable of providing

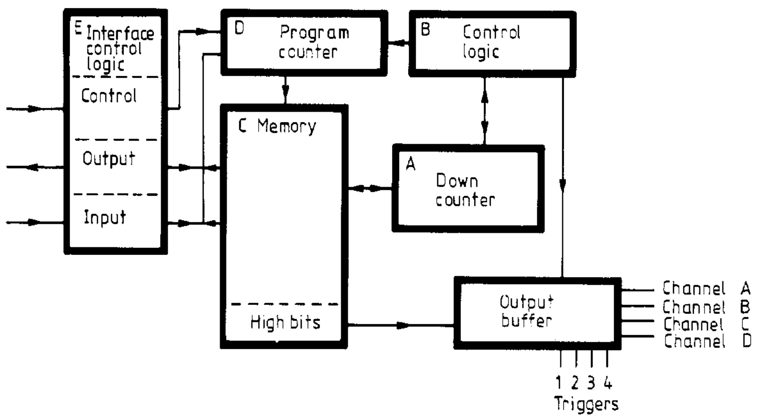

Figure 1. Block diagram of the pulse programmer. Copies of the complete circuit diagram are available. Enquiries should be directed to the last author.

timing durations over the range 100 ns to about 7 min in 100 ns steps. A feature of this pulse programmer, found in only one previous design (Cosgrove *et al* 1980), is that of a hardware protection circuit to guard against long RF pulses and excessive duty cycles. Protection values are conveniently selected by means of small plug-in modules.

The acquire instruction divides the 32-bit counter into two 16-bit counters, the first holding the number of clocking pulses for the ADC and the second containing the dwell time between clock pulses. The hardware executes this step by looping within a single step. The other output channels mentioned above may also be selected, but they remain high for the whole length of the pulse train.

The repeat and stop instructions determine the program flow. In the case of the time and acquire instructions, the pulse programmer simply continues with the next step stored in memory. The stop instruction halts program execution and sets a flag which can be tested by the host computer. The pulse programmer can be restarted either manually or by computer. A repeat instruction jumps back to the instruction stored at location zero in memory, which is the nominal starting point.

Once downloaded by the computer, the pulse programmer may be operated manually or by computer. When under computer control, the pulse programmer operates independently of the computer in the sense that control of the experiment passes to the pulse programmer for precise timing of pulsing and acquisition of the FID, and control passes back to the computer for accumulation and storage of the data. If control is ever lost, a manual reset places the pulse programmer in a 'wait' mode without corrupting the pulse sequence. Acquisition of the FID is clocked by the pulse programmer directly into the computer memory (by means of a commercial direct memory access board) while the computer may be carrying out other tasks without interference.

3. Description of the software

The software written to manage the NMR experiment and control the pulse programmer was written in FORTRAN. Access to the pulse programmer is carried out by means of peeks and pokes. The software consists of a series of program modules as depicted in figure 2. Each program in turn is written around a library of subroutine modules which may be easily modified to meet certain requirements, as for instance, alternate addition and subtraction to memory or an upgrade of the NMR hardware for quadrature detection.

The computer boots automatically to the MENU program which lists the various options available. All other programs, upon termination, return to the MENU. The various programs in the software system communicate through a series of files on the disc. All such files are text files which can be read visually by the operator on the screen or printed out. Comments (lines preceded by an exclamation mark '!') appear in pulse program and data files. Management programs have been written to ignore such comments unless they have unresolved parameters.

Running an NMR experiment begins by executing the TUNE program. This loads the pulse programmer with a simple pulse sequence used for tuning the spectrometer. It consists of a single pulse whose duration is controlled by a hand-turned potentiometer. The pulse time may be stored on the disc during tuning at the touch of a key. It is always assumed that the pulse time recorded is that of a 180° pulse. This value is later picked up by the SET-UP program to set both the 180° and 90° pulse lengths.

The next step in an experiment is to select a pulse program from a library of pulse programs which have been previously written and stored on the disc. These programs are written by the PPWRIT program in which the user (at this level of operation) has available to him/her eight types of steps:

PULSE A	DELAY
PULSE B	ACQUIRE
PULSE C	STOP
PULSE D	REPEAT

plus any of the four trigger lines. This expanded list of types of steps makes the programming operation conceptually easier for the user. It can be learned in five minutes. A time duration is associated with each pulse or delay; four kinds of duration are allowed:

(i) simply a fixed time, e.g. 5 ms;
(ii) a programmable time (P1, P2, etc) which is fixed at run time;
(iii) a 90° or 180° pulse whose length is determined on the day of the experiment during tuning;
(iv) a variable time (VT). This is a special type of programmable time, similar to (ii), but which is stepped through a *list* of values instead of a single value. The list is fixed at run time.

A short pulse program which illustrates many of the above features is shown in figure 3. Pulse programs have been designed to be easy to write, read and use. The optional comments are also displayed as prompts at the terminal to aid the operator in setting up an experiment.

The SET-UP program organises everything necessary to run a series of experiments. It asks for all relevant parameters such as

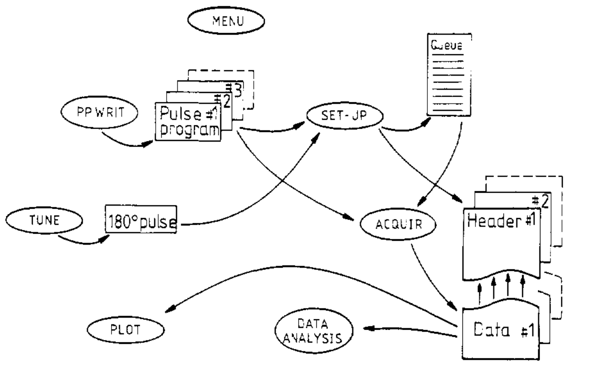

Figure 2. Organisational diagram of the NMR software. Disc files are represented by rectangular ikons; programs by ellipses and information flow by arrows.

```
1, PULSE A,     S180,
2, DELAY,       VT,
3, T1PULSE A,   S90,
4, DELAY,       0.10000E-04,
5, ACQUIRE,     ★★★,
6, DELAY,       P1,
7, STOP,        ★★★,
8, REPEAT,      ★★★.
!THIS IS 'PPT1', A SIMPLE PULSE PROGRAM TO MEASURE
T1
!180-VT—90 INVERSION RECOVERY PULSE SEQUENCE,
FIRST THE 180° PULSE
  1, PULSE A,  S180,
  2, DELAY,    VT,
!NOW THE 90° PULSE ACCOMPANIED BY TRIGGER #1
  3, T1PULSE A, S90,
!DELAY FOR RECEIVER RECOVERY
  4, DELAY,     0.10000E-04,
  5, ACQUIRE,   ★★★,
!RECYCLING DELAY TO ALLOW FOR RELAXATION OF SPIN
SYSTEM
  6, DELAY,     P1,
  7, STOP,      ★★★,
  8, REPEAT,    ★★★.
```

Figure 3. Example of a very short pulse program selected to illustrate the different options available in a pulse sequence. This program is shown first without, then with comments.

experiment name, date, dwell time, spectrometer frequency etc. It calls up the pulse program, resolves all undetermined parameters, prompting the operator when required with comments from the pulse program and sets the length of the 90° and 180° pulses. For each experiment, it writes a comment header file, recording all the details for future reference. It simulates the pulse sequence and informs the operator if preset values for pulse lengths or duty cycle are exceeded. This is in addition to the hardware protection which has been built into the pulse programmer. Finally, the SET-UP program places the name of the experiment onto a QUEUE.

To run the experiments, the ACQUIRE program calls up the QUEUE, then the header file and the PULSE program file, loads the pulse programmer, acquires the FID and appends the data to the header file. The program continues running experiments until the QUEUE is empty. Once collected, data files are left intact, but they are used as input files to the analysis programs which may be tailored to suit the demands of a particular experiment, e.g. multiple exponential fits, fast Fourier transform, etc.

To aid the operator, HELP files have been written which may be accessed from within the above programs to allow him/her to recall details of operation.

4. Summary

A hard-wired computer-controlled pulse programmer has been constructed and used with a computer to automate the acquisition of data on a previously manually-operated (Bruker SXP) NMR spectrometer. The pulse programmer is machine independent and can be connected to any computer with parallel I/O lines. In conjunction with the FORTRAN software written for it, the pulse programmer and the spectrometer form a data collection and analysis system which is very versatile, is easy to use and has provision for future expansion. A circuit diagram of the pulse programmer is available upon request.

The software written for the system allows great flexibility and a degree of 'user friendliness' which many commercial systems still often do not achieve. Among its features are a simple method of writing pulse programs which can be learned in a few minutes; HELP files which prompt the operator; the storage of experimental parameters with the data in easy-to-read text files; protection against high power RF pulses and a queueing system which allows a series of experiments to be set up, queued and then run without operator intervention.

References

Adduci D J and Gerstein B C 1979 Versatile pulse programmer for nuclear magnetic resonance
Rev. Sci. Instrum. **50** 1403

Bendall M R and Pegg D T 1985 Theoretical description of depth pulse sequences, on and off resonance, including improvements and extensions thereof
Magn. Res. Medicine **2** 91

Caron F and Herzog R F 1978 A new programmable timer designed for pulsed NMR
J. Magn. Res. **31** 357

Cosgrove T, Littler J S and Stewart K 1980 A computer-controlled pulse sequencer for pulsed NMR experiments
J. Magn. Res. **38** 207

Dart J, Burum D P and Rhim W K 1980 Highly flexible pulse programmer for NMR applications
Rev. Sci. Instrum. **51** 224

Doherty S P 1982 A simple and inexpensive pulsed NMR data acquisition system
J. Magn. Res. **37** 31

Gianotti A, Lutter R and Ziessow D 1980 A computer programmable digital pulse generator for Fourier NMR spectroscopy
J. Phys. E: Sci. Instrum. **13** 956

Hore P J 1983 Solvent suppression in Fourier transform nuclear magnetic resonance
J. Magn. Res. **55** 283

Mehring M 1983 *High Resolution NMR in Solids* (Berlin: Springer)

Morris G A and Freeman R 1978 Selective excitation in Fourier transform nuclear magnetic resonance
J. Magn. Res. **29** 433

von Os J W M and Veeman W S 1979 Programmable digital pulse generator for proton enhanced ^{13}C high resolution NMR in solids
Rev. Sci. Instrum. **50** 445

Pope J M and Eberl S 1986 On surface coils and depth pulses
Magn. Res. Imaging **3** 389

Pratt R G and Ackerman J L 1980 Pulse-pair concept in the design of a microprocessor-bassed programmer for solid-state NMR
J. Magn. Res. **41** 140

Saint-Jalmes H and Barjhoux Y 1982 Powerful timing generator using mono-chip timers: an application to pulsed nuclear magnetic resonance
Rev. Sci. Instrum. **53** 1

Shaka A J, Keeler J, Smith M B and Freeman R 1985 Spatial localisation of NMR signals in an inhomogeneous radiofrequency field
J. Magn. Res. **61** 175

Shenoy R K, Ramakrishna J and Srinivasan R 1976 A versatile and flexible digital pulse programmer for pulsed nuclear magnetic resonance
J. Phys. E: Sci. Instrum. **9** 779

Taylor D G, Booth S and Allen P S 1974 A flexible modular pulse programmer suitable for diverse nuclear resonance experiments
J. Phys. E: Sci. Instrum. **7** 105