

## Chapter 1: Introduction and general instrument discussion

The Nu Instrument's Noblesse noble gas mass spectrometer is a state of the art instrument designed specifically for the analysis of noble gases and other species under static vacuum conditions. The instrument is a single focussing, 75° magnetic sector design, with the magnet radius being 240mm. Due to the non-normal entrance and exit poles, the dispersion at the collector plane is equal to double the magnet radius, corresponding to 480mm. Between the magnet exit pole and the collector are placed two miniature lens arrays, which are used to provide a field equivalent to an electrostatic quadrupole. This enables a zoom lens to be incorporated into the vacuum envelope, allowing for the dispersion to be altered from the figure quote above. This patented approach permits the user to undertake multi-collector analyses, without the necessity of having to move collectors as different elements are studied.

Various different configurations of collector design are offered, normally consisting of at least one faraday collector together with a number of discrete dynode multipliers. Further, to enable analyses of such elements as helium, where the minor <sup>3</sup>He isotope has to be recorded in the presence of the much more intense <sup>4</sup>He beam, we also provide a retardation lens assembly to improve the abundance sensitivity of one of the ion counting channels. To further increase the dynamic range of some of the detector channels, it is also possible to place a multiplier and faraday detector behind a slit, and use an electrostatic deflector to send the ion beam into one or other of the pair.

The standard source supplied with the instrument is based on the well-known Nier design, but has been optimised for working in a static vacuum environment. Great care has been taken in the design to optimise the source's efficiency, as well as its user friendliness and longevity of filament life. Although it, and its accompanying electronics, has been designed to work at up to 8kV acceleration voltage, most users will not take it to these voltages, since the gain in sensitivity above about 5kV will be found to be small. However, it may be found to be advantageous to work here if the ultimate resolution is required, since any aberrations will be reduced as the acceleration voltage is increased.

A getter pump is placed immediately behind the source, in order to minimise any impurity peaks levels during analysis. A spare port is also provided on the manifold entry block, to which a further pump or cold finger may be fitted. Two ion pumps are also provided as standard, one off the manifold entry block, which is fitted with a pneumatic valve, and a second off the collector block, fitted with a manual isolation valve. It is expected that this second pump will only normally be used when the instrument is not in use (and during bakeout), and samples will be evacuated after analysis using the source ion pump. Finally a 70l/sec turbo pump may also be fitted, together with a dry (scroll) backing pump. This will only normally be used to evacuate the mass spectrometer after venting, although a connection is also provided to enable this pump to evacuate the user's manifold assembly, if so required.

A dedicated software package is provided to enable the analyses of the isotopic ratio of the gases to be fully automated. The electronic suite provides spare outputs to control the user's automated manifold valves, and a separate software utility is provided to enable schematic representations of these manifolds to be drawn, so as to enable them to be incorporated into the main control suite. The software permits two

automatic sequences to be performed simultaneously, thus enabling samples to be prepared whilst the previous is being analysed, and hence optimising sample throughput. Further, it is also possible to control laser systems from the suite, providing that the laser can communicate via commands sent over a serial link. The main analysis suite performs time fits of the individual beams (extrapolating the data to time zero) using either an exponential or linear model. The time zero data can then be analysed automatically using a further program, to combine the data to obtain the required isotope data. The runs can also be combined with previous analysis results, this method, for example, allowing blanks to be simply measured. All the beam data is also available in CSV format, to permit the user to analyse the data using their own programs, if more complicated data manipulation is required.

The suite also enables the user to fully characterise the performance of the instrument, and to fully control it from the keyboard. The microprocessor in the system control unit within the instrument bench is also continually checking the status of all the trips on a 10 millisecond time rate, so as to minimise the possibilities of accidental faults causing any damage to the instrument parts. Further, this unit is also checking to ensure that the outputs of many of the units match the required set values, and if any such fault is detected, the user is notified via the controlling personal computer screen.

### **General mass spectroscopy theory:**

These paragraphs are not supposed to replace the many textbooks, which are available, and which provide a full background to the theory and practice of mass spectroscopy, but rather to give a short introduction relevant to the use of this instrument. As such it is a very specific selection of some of the more relevant points which are often required to appreciate the terms used to specify this, and similar instruments.

For a good, general, introduction to this field, perhaps the purchase of:

Mass Spectroscopy (second edition), by H.E. Duckworth, R.C. Barber and V.S. Venkatasbramanian. Cambridge University Press – available in paperback, 1990

would not come amiss.

### **Absolute basics:**

The primary function of the source of the mass spectrometer is to ionise the sample so that it may be transmitted from the source region towards the detectors. Once ionised, it is extracted from the source using electrostatic fields, and these fields impart a high forward acceleration on these particles, in the region where these fields act. There is little or no change in the transverse velocity of the ions, and one can imagine the initial  $2\pi$  solid angle cone of particle emission (at thermal temperatures), being narrowed, as the ions receive this increased forward velocity. The narrower this cone angle, the more likely is that the ions will pass through any physical restriction in the mass spectrometer path without being stopped by that restriction. This is the reason why most mass spectrometers use a high voltage to accelerate the ions, and gives rise to higher transmission (= sensitivity). However once this cone angle is small enough

for the ions to pass through without hindrance, there is little to be gained by increasing the acceleration voltage. Some instruments are designed and require very high voltages to optimise the transmission, whereas others use a lower value. It should be obvious, using the model outlined above, that sensitivity should increase as the square root of increase in acceleration voltage, until a plateau is reached when the beam is no longer restricted by apertures inside the mass spectrometer envelope. Since this is a “hard” way to increase sensitivity significantly, due to the square root relationship, it is better to design the instrument so as not to require extreme voltages.

The separation of the various particles according to mass is achieved by the magnet. There is the well-known sideways force on the ionised particle as they pass between the poles of the magnet (direction given by the “left hand rule”). If all the ions have the same energy, the heavier ones are not deflected as much as the lighter ones, permitting the separation by mass to occur. What is not so obvious is that there is also focussing of the beam as it enters and exits the field, in both the horizontal and vertical directions. This fact permits one to design a very simple mass spectrometer using a source slit, a magnet and a collector slit, and the cone of ions emanating from the source are brought back to a focus at the collector. If it wasn't for the fact that the magnet also has this built in lensing property, there would be no well defined image at the collector slit, and separation of the ions by mass would not be recordable.

If the velocity of the ions through the instrument is due solely to the accelerating potential, we have:

$$\frac{1}{2} mv^2 = zeE \quad (1)$$

where m is the mass of the ion, v its velocity after being accelerated by field E. z is the number of charges present on the ion and e is the charge of the electron.

Passing through the magnet, there is a sideways force  $Bzev$ , producing an acceleration  $mv^2 / r_m$  towards the centre of the circular path.

$$\text{Thus} \quad Bzev = mv^2 / r_m \quad (2)$$

$$\text{Which gives:} \quad r_m = \frac{\sqrt{(2 m E / ez)}}{B} \quad (3)$$

Or (approximately, for ions of unit charge)

$$r_m \text{ (cms)} = \frac{144 \times \sqrt{(m(\text{amu}) * E(\text{volts}))}}{B(\text{gauss})}$$

This is the basis of the “single focussing” mass spectrometers, which produce a reasonable focussed beam due to all the ions having a similar energy i.e. equation (1) defines the ion velocity accurately. With a gas source such as the one used on the Noblesse instrument this is indeed the case, but it will be worth while looking at other designs, where this may not be the case. If ions of the same mass have differing velocities, this is equivalent to them having varying energies. This can occur if the initial starting velocity spread is large, or may result from the initial ionization process producing an energy spread. Equation (3) tells us that the ions of the same mass no

longer experience the same trajectory through the magnet, but the value of the radius of curvature also alters. One can see this intuitively. Ions with higher energy will not be bent as much as those with a lower value.

To overcome this smudging of the image, we may introduce a second element into the ion optic path, which can affect the image position depending on energy, but not mass. This can then cancel out this effect. This new element is an electrostatic analyser (ESA), simply two curved plates which bend the ion beam around a second curve. If the voltage on the plates is  $\pm V$  and the plate separation is  $2d$ , the central field is  $V/d$ . Passing through the electrostatic analyser, there is now a sideways force  $zeV/d$ , producing an acceleration  $mv^2 / r_e$  towards the centre of the circular path.

$$\text{Thus} \quad zeV/d = mv^2 / r_e \quad (4)$$

Which can be combined with equation (1) to give:

$$V/d = 2E / r_e \quad (5)$$

Thus we have the required dependence of ion curvature with ion energy, but the mass term is absent. Also it can be shown that the electrostatic analyser also acts as horizontal lens element, focussing the source slit to an image point, just as the magnet did (Spherically, rather than cylindrical shaped plates can also focus in the vertical direction.) If we consider a simple combination of these two elements, we can envisage an electrostatic analyser producing an intermediate image point, with ions of higher energy being brought to a focus on the outside, and ions of lower energy on the inside. Careful thought will show that if we consider these intermediate image points as the source for the magnet trajectories, we may be able to compensate for the spread of focus position caused by the magnet itself. (HINT: consider the ions flying back from the final image point through the magnet, and see where they come from.)

Although not straightforward, it can be shown that with the careful choice of magnet and ESA radii, and the drift free path lengths, it is possible to compensate for this defocusing of the final image at the collector due to the initial energy spread in the ion beam. Such a combination of ESA and magnet is termed “double focussing”. Although such an arrangement is symmetric (i.e. it does not matter whether the ESA or magnet comes first), we are interested in the multiple collection of the ion beam (to compensate for instability in the source), and it is sensible to have the mass separation last, in front of the collector array. For historical reasons this is called the “*normal*” geometry, and instruments with the magnet before the ESA are of “*reverse*” geometry.

With such a simple description, it may be thought that it is relatively easy to produce a sharp image of the source at the collector. To totally disillusion you, this is not the case. In practice it is the job of the instrument designer to calculate as exactly as possible the paths of all the possible ion trajectories through the instrument, and see how well they are focussed at the final image plane (which is probably not even flat!). The smudging of the image is technically caused by aberrations, and it is usually possible to determine which of these aberrations are dominant in a particular design, and to try to minimise their effect. It is virtually impossible to remove them, one

merely tries to make them to be of a level such as not to adversely affect the performance of the instrument.

### **Mass resolution:**

This quantifies the smudging of the final image beam. There are many different definitions of this term, but basically it is given by:

$$\text{Resolution} = \frac{\text{Actual mass of peak studied}}{\text{Width of peak}}$$

$$\text{Or } R = \frac{M}{\Delta M} \quad (6)$$

The ambiguity comes in the measurement of  $\Delta M$ , and two common standards apply, the width at half of the peak height or at the 10% positions. If two equal gaussian peaks are separated by an amount equal to  $R(10\% \text{ definition})$ , then the “valley” between them should get to 20% of the baseline.

For precision intensity measurements, it is conventional to have the collector slit wider than the image of the source. This produces a “flat topped” peak (corresponding to that part of the scan when **all** of the ion beam is collected by the detector). This is a sensible regime to work, since the instrument will undoubtedly have minor instabilities, with the image of the source moving slightly across the detector. But since the collector slit is so wide, it still receives the entire beam. If the collector slit was narrower (or exactly the same size) than the source image, the recorded signal would be extremely sensitive to such instrument jitter.

If the instrument is set up to have flat-topped peaks, the measured resolution will be less than the theoretical limit of the design. This could be obtained by considering the sides of the observed peak, and it is left to the reader to work out how to do this.

### **Mass Dispersion:**

This quantifies the separation between adjacent masses in terms of a physical distance, the Dispersion length (D). Given this value (which is a property of the mass spectrometer design) it is a simple matter to calculate the physical distance between peaks at mass M and (M +  $\Delta M$ ). If this distance is (X) then:

$$\frac{D}{X} = \frac{M}{\Delta M} \quad (7)$$

We normally are interested in the case with  $\Delta M=1$ , since this will tell us the separation between adjacent collectors of a multiple collector array. Since D is fixed for the instrument design, this formula tells us that as the mass decreases, the separation between adjacent masses will increase. This is the reason for previous generation of mass spectrometers incorporating movable collector arrays. If one utilises a fixed collector array, one must either magnify or demagnify the final image

using zoom optics, to ensure that they are coincident with the ion beams. The problem then is not to introduce severe aberrations by so doing.

It can be shown that resolving power and mass dispersion are not independent, which results in most instruments having very similar values in practice, given the general physical size of a machine.

### **Abundance Sensitivity:**

This describes the tailing of one peak into its neighbour.

The simple theory given above assumes that once the ion leaves the source region, no further interactions occur and that its trajectory can be determined by numerical calculation. In practice the ions can undergo collision during their passage within the mass spectrometer. Consider an ion which has passed the magnet, and is calmly proceeding expecting to enter the collector ahead of it, set up to receive all similar ions of its particular mass. Suddenly it experiences a collision with the background gas. Since it is travelling much faster than this background neutral (remember it has been accelerated through a high voltage), it will probably suffer only a small change in trajectory, but it will no longer be incident in its original collector. If it arrives at one of the neighbouring collectors, we have recorded the ion at the wrong mass. This is main cause for the tailing from one mass onto its neighbours, and can be of large significance if we are trying to accurately measure a weak peak near to a very strong one.

If the ion beam collides with any part of the mass spectrometer inside surface, and continues in a forward direction, it will also have lost energy due to the interaction. This is another common cause for poor abundance sensitivity, but is really a symptom of poor instrument design. A more frustrating case is where a small hair (or such like) protrudes into the ion beam path. This will charge up, and deflect a small percentage of the total ion beam. The net effect is the main peak is unaffected, whilst it sits on a weak, totally out of focus, baseline. This again can look like a case of poor abundance sensitivity.

### **Detector Design:**

The most accurate (and precise!) data is obtained using the **Faraday collector**. This is basically a long thin receptacle, into which the ions pass, impinging onto the base of the bucket. Ideally, on impact, the ion charge will pass along the wire to the amplifier attached to the detector, and be recorded by the electronic circuitry. Of course life is not that easy.

Upon impact of the high energy ion, a cascade of events can start. The initial impact will eject secondary ions and electrons from the surface. (There is an entire analytic technique – SIMS – which relies on this effect as it's primary ionisation process.) If one of these electrons were to escape from the “bucket”, the net change in current would no longer be one, but two (since we have lost a negative charge). Similarly if an secondary ion were escape, the final recorded signal would become zero! To overcome these problems, the design of these originally simple devices have become quite sophisticated. The base material is specially selected to minimise the yield do

secondary species, and is now normally some form of highly porous carbon. The actual buckets themselves are made as long as possible, to minimise the value of solid escape angle. Some form of magnetic field is also often present, so as to spiral the electrons into the side walls of the bucket, and so increase the trapping efficiency.

From early work on detectors, it was noticed that many systems showed anomalous negative dips, at the sides of the recorded peaks. These were identified as coming from secondary electrons formed as the ion beam impinged on the detector slit (i.e. the metal shield in front of the actual detector device) and subsequently entering the collector mouth. This was especially prevalent as the ion beam struck the slit sides, even occurring if very thin slit material was used. This effect was overcome by placing an extra electrode (the suppressor) after the defining slit, held at  $-50$  to  $-100$ v. This repelled the electrons, and had the added bonus of helping to repulse those formed inside the bucket itself, back to the base. Care must be taken with such electrodes, to ensure that they are not seen by the ion beam itself, since any secondary electrons formed from such interaction would be “sucked” into the bucket by its positive potential with respect to this suppressor.

Where the Faraday detector is not sensitive enough (see below), various forms of ion multiplier are available. These rely on an avalanche effect, whereby the initial impact produces a small shower of (say) 5 to 10 electrons. These are accelerated down the device by the applied field, and when each electron impinges the surface again, another shower of 5 to 10 electrons occurs. This continues down the multiplier chain, until at its end, a pulse of  $10^6$  to  $10^8$  electrons is achieved. This then can be observed by “conventional” electron circuitry and recorded as a single event. The multiplier can be of various forms, discrete dynode devices have each surface which produces a gain increase as a separate entity, connected to its neighbours by a resistor chain, thus producing the voltage difference between stages. Continuous dynode devices rely on a thin layer of semi-conducting glass inside a curved tube to provide the amplification stages. The high resistance of this thin layer enables the voltage gradient to be maintained along its path.

Since there is a much larger current pulse towards the rear of these devices rather than at the entrance, this is the region where most problems can be found. With the continuous dynode device, the high resistance glass itself must provide the current to replenish the charge. Since many of these devices are quite small, the current carrying capabilities are limited, and the response at high-count rate often falls off. (This is on top of the dead time effect – see below.) Also it is often feasible for an ion to be produced by the electron impact, or for a background gas molecule to be excited by the electron cloud, as it passes. This ion can then migrate back towards the entrance of the device, under the applied voltage to the device. The impact of the ion with the surface can produce a secondary shower, delayed by as much as few microseconds from the original due to the lower velocity of ions relative to electrons. Whether this false event is counted will depend on the discriminator setting of the detection electronics.

Since these devices actually can run quite warm (their resistance is 3 to 30 Mohm and they work at 2 to 3 kV – the sum is for you to do!), it is best to switch on the multiplier supply some time before use, so as to “boil off” as much of the surface contaminant as possible. The large electron current in the rear sections can also

increase the local physical pressure, due to the ejection of surface species, which again can mean that this secondary pulse yield can vary until the device is fully conditioned.

**Detector noise:**

Faraday systems: It is conventional to convert the small detector current to a voltage by using a high value resistor, normally  $10^{11}$  ohm, across a high input impedance operational amplifier. There is a random noise associated with this high value of resistance (Johnson noise), given by:

$$\sigma(I) = \sqrt{(4kT\Delta f / R)}$$

- where k = Boltzmann constant ( $1.38 \times 10^{-23} \text{ JK}^{-1}$ )
- T = Absolute temperature
- $\Delta f$  = measurement bandwidth
- R = Resistance value

Thus for a 1 second measurement period there is a minimum noise current limit of  $4 \times 10^{-16}$  amps rms, and for a 5 second integration period this improves to  $1.8 \times 10^{-16}$  amps rms. This is the absolute limit of this technique, and assumes that all other sources of noise have been eliminated which can be a non trivial task.

Multiplier systems: Here one can use the discriminator settings to lower the possibility of counting false events. These are not only the secondary pulses mentioned above, but also any other spurious signal. Such pulses can arise from local breakdown along the resistor chain (a major problem with the smaller, more compact, devices where the field gradients are larger), events triggered by radioactive breakdown (a problem if “nuclear” studies are undertaken, or radioactive spikes are used). Noise seen in photomultiplier systems (such as the Daly detector), where thermionic emission from the photocathode is dominant, normally limits the noise to 1-2 cps. With other devices, a level of a few counts per minute is feasible. The limit here is probable defined by the level of cosmic ray activity, and may actually result in poorer performance at high altitudes. (We assume that the system works correctly and that the discriminator can be set to give a multiplier gain of about 90%.)

**Beam Noise:**

As well as any jitter in the observed beam intensity of to instabilities of the source itself (which should be minimised by good design), there is a more fundamental limit of to the observable measurement precision with weak beams. Since the beam consists of a stream of individual particles, there is an uncertainty in the number of particles in that stream, which may be estimated assuming that the stream has a Poisson distribution. Here, the well known formula:

$$\text{Counts} = n \pm \sqrt{n}$$

that applies to statistical errors in many situations involving counting of independent events during a fixed interval applies.



### **An simple example:**

Suppose we have a sample which will produce 20,000cps for a period of 100secs.

With a multiplier system we would expect the best measurement precision to be given by  $(\sqrt{2 \times 10^6}) / (2 \times 10^6)$ , i.e. 0.07%. This ignores any systematic errors due to the determination of multiplier gain etc.

With a faraday system, the current is  $(2 \times 10^4 \times 1.6 \times 10^{-19})$  i.e.  $3.2 \times 10^{-15}$  amps. The resistor noise is determined above at  $4 \times 10^{-16}$  amps for a 1 second integration period. This corresponds to a measurement precision of 12.5% per sec, or 1.25% for the 100 second total measurement.

In practice the calculated multiplier precision quoted is optimistic, since it is exceedingly difficult to measure the gain to this level (unless one designs an experiment whereby the gain is determined as one step in a two step measurement sequence). It should also be noted what occurs if the experiment is redesigned to give a ten times more intense pulse for a correspondingly shorter period. The Faraday detector precision now improves to 0.39%, whilst the multiplier limit is unchanged.

### **Ion multiplier dead time:**

The counters, which follow the ion multipliers, are triggered whenever a signal is seen which has an intensity larger than a certain, pre-set value (the discriminator setting). We trigger on the rising edge of the transition. If you consider what would occur if there was a second pulse following the first, an interesting problem arises if the time between the pulses is too short. At low count rates the state of affairs is quite simple, the first pulse decays and the second gives rise to a separate voltage pulse, which can be recorded as the transition from below the discriminator setting to a higher value occurs. However if the two pulses are very close together, the output due to the first pulse from the preamplifier may not decrease to a value below the critical discriminator setting before the second arrives. In such a case the counter will not be incremented by the second event, and the recorded pulse rate will be slightly lower than the true value, due to this effect.

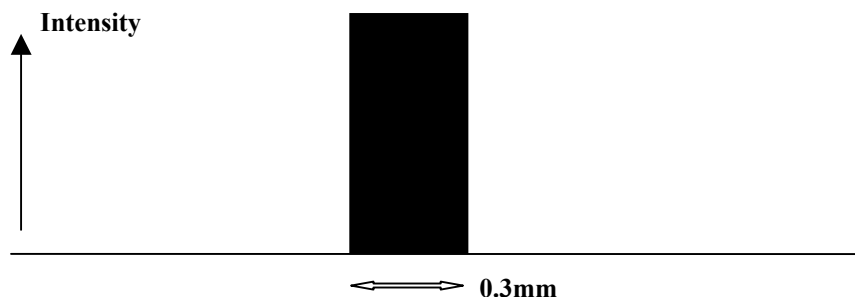
To allow for this loss, a mathematical correction is applied to the observed data, and the formula employed is:

$$\text{Corrected count rate} = \frac{\text{Observed rate}}{1 - \text{observed rate} \times \text{deadtime}}$$

### **Aberrations:**

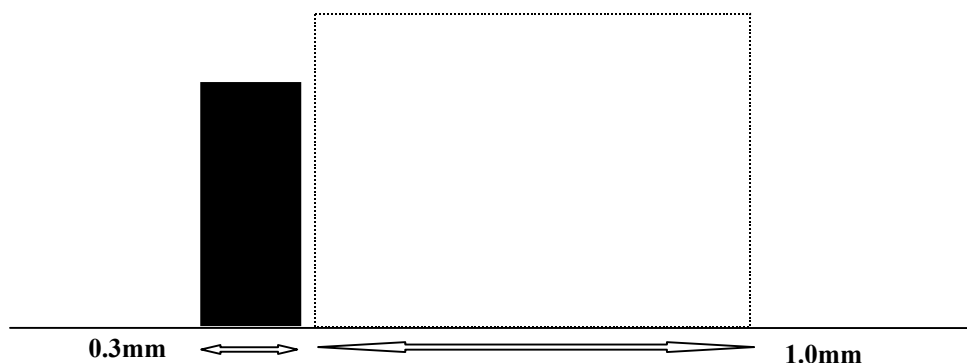
Simple perception of the trajectory of the ion beam through the mass spectrometer will give a idealistic impression of what is really occurring. From high school light optics knowledge, one expects the final image to be an exact copy of the source, but life is never so simple. It is actually a DISTORTED copy, and the degree of this distortion is a consequence of the aberrations introduced by the ion optical elements in the path through which the ions have to pass. Let us consider a simple example and

we will sketch out the beam intensity as we do various things to the beam. Firstly we have the source slit, which will define the intensity profile of the beam entering the mass spectrometer. We assume the simple case where the slit is uniformly illuminated, and it will then produce the following profile as we scan across the beam immediately after the slit:

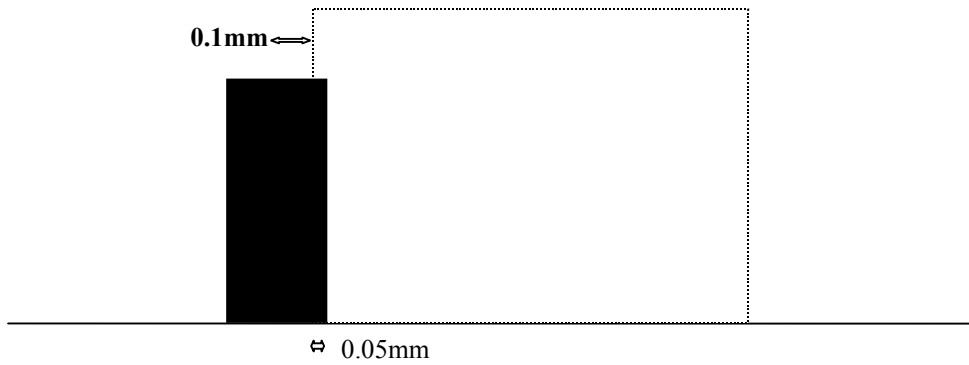


In this example we have assumed that the source slit has a width of 0.3mm.

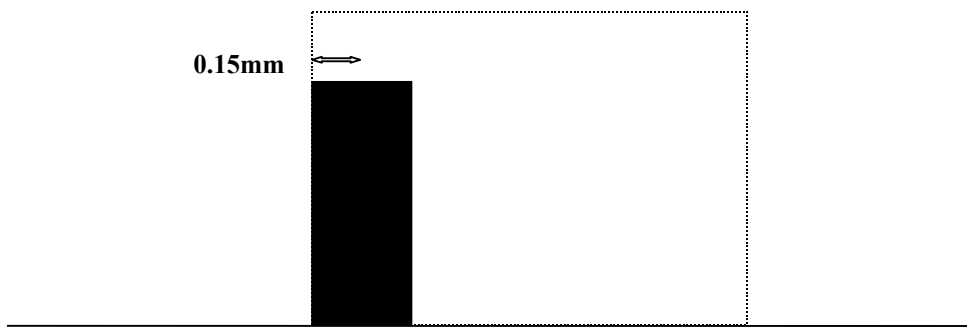
Let us now consider what occurs if this beam is scanned across a collector slit of (say) 1mm width, in the absence of any distortions. This is then the case for an ideal spectrometer:



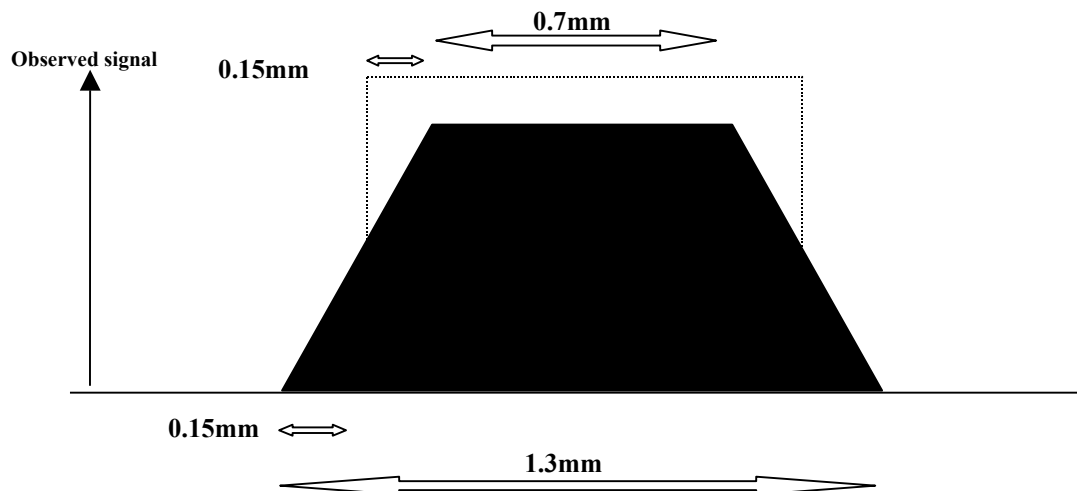
In the above sketch, we assume that the beam is just to one side of the slit - the magnet is set to a value, which is just too low to allow any of the beam to enter the collector slit. Nothing will be recorded on the detector. Now consider what the detector records as the magnetic field is slowly increased, and the image of the source slit is moved from left to right in the above diagram. When the centre of the profile is exactly 0.15 mm from the left hand edge of the collector slit, the detector is just on the verge of recording a signal. If the centre was 0.1mm from the left hand edge, the detector would record one sixth of the maximum possible signal. I illustrate this case below:



The detector does not record the maximum possible signal until the magnet has moved the profile so that its centre is 0.15mm past the left hand edge of the collector slit:



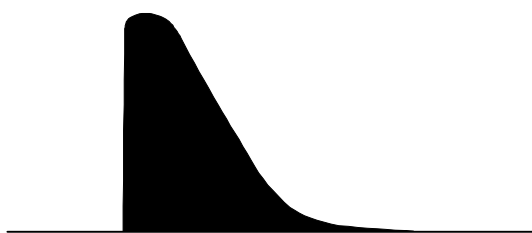
If we now consider what this process corresponds to for the RECORDED detector signal, it will be obvious that this will have the familiar trapezoidal form:



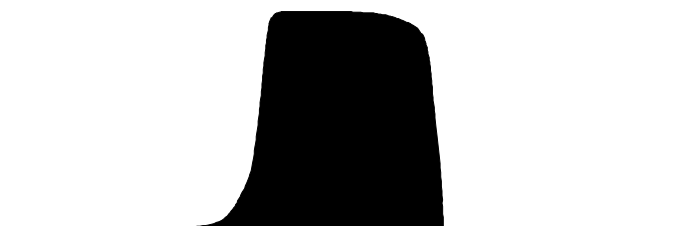
The peak flat region of this ideal peakshape is seen to be equal to the collector slit width LESS the source width, whilst the total width at the recorded peak base is equal to the SUM of these two. The width of the peak at the HALF height positions, is equal to the collector slit width.

In practice the beam, as it is incident on the collector slit, is not the ideal profile, but distorted. These distortions can be simply grouped into two types. The most annoying

(and perhaps common) is the asymmetric distortion, in which the final image shows a distortion to one side:

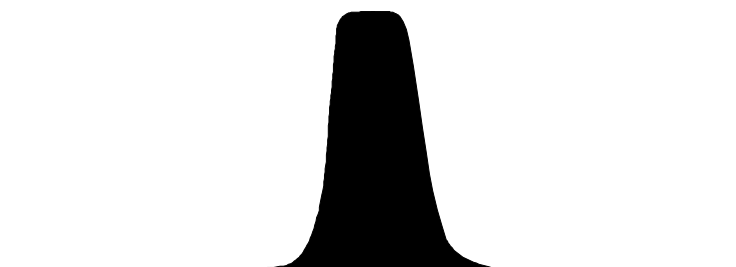


This is shown, greatly exaggerated above, where I have also started with a narrower source slit width than previously, to emphasise the effects. It is interesting to consider what the observed detector signal would be if such a beam profile was incident on the collector slit, since this is the observable. The result is something like:



which may well be a familiar shape to many users.

A symmetric distortion of the ion beam could produce an image of the source slit as illustrated below:



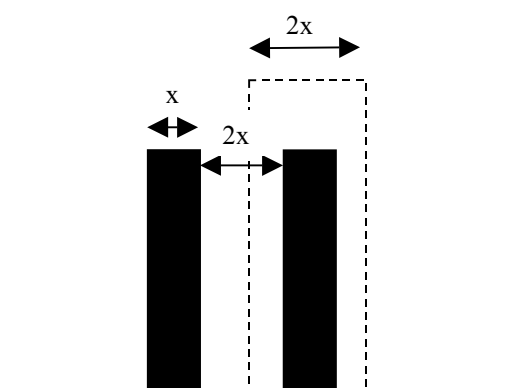
I leave it to the reader to obtain the recorded collector intensity as the peak is scanned across the collector slit, for such an incident beam profile.

Without going into detailed ion optical theory, I merely provide the rule that if there is a symmetric aberration, this can be corrected with the application of an even powered electrostatic field. The most common example of this would be the “normal” out of focus beam, when altering the zoom settings (a quadrupole or second order field distribution) will bring the beam into focus. The asymmetric recorded beam intensity can often be corrected by the use of a cubic term to the zoom elements, but this may only work with the axial beam, and may add other distortions on the off axes collectors.

## Pseudo High Resolution

It is also possible to perform high resolution studies without resorting to narrow collector slits, as long as the interfering species all lie to one side of the peak of interest. To distinguish this technique from the standard method discussed above, we have termed the name “Pseudo High Res”. However, do not be misled to think that this is in any way inferior to the more conventional analysis method. As we show below, this can enable peaks to be studied, which cannot be resolved by the standard approach, and possibly under higher sensitivity conditions.

To illustrate the method, consider the simple case where there are two adjacent peaks, of equal intensity next to each other, one being the peak of interest and the second an interference. We will ignore any aberrations in what follows.



The beams, as they hit the collector slit, are shown here, where we have two (obviously) equally wide beams, of width ( $x$ ), separated by a distance  $2x$ . In the first case we will assume that the collector slit width is also  $2x$  wide.

Consider what occurs as the beams are swept over the collector slit. When the first beam is  $x/2$  from the slit, the recorded signal will start to rise. As will be seen the sequence of events will be recorded as:

1. The signal starts to rise as the first beam crosses the left-hand edge of the slit.
2. The recorded beam then stays constant as the beam traverses the slit (as shown in the drawing above)
3. The beam drops to zero as the first beam leaves the right hand edge of the detector slit and the second beam just starts to enter on the left.
4. The signal starts to rise again as the second beam crosses the left-hand edge of the slit



5. The recorded signal stays constant
6. The signal then goes back to zero.

The observed signal is shown here, where we leave it to the reader to fill in the relative width of all the beams. Note that the two peaks are just totally resolved in this ideal example.

Consider the case where the collector slit is now much wider, and again imagine what occurs as the beam moves (from left to right) across the slit:

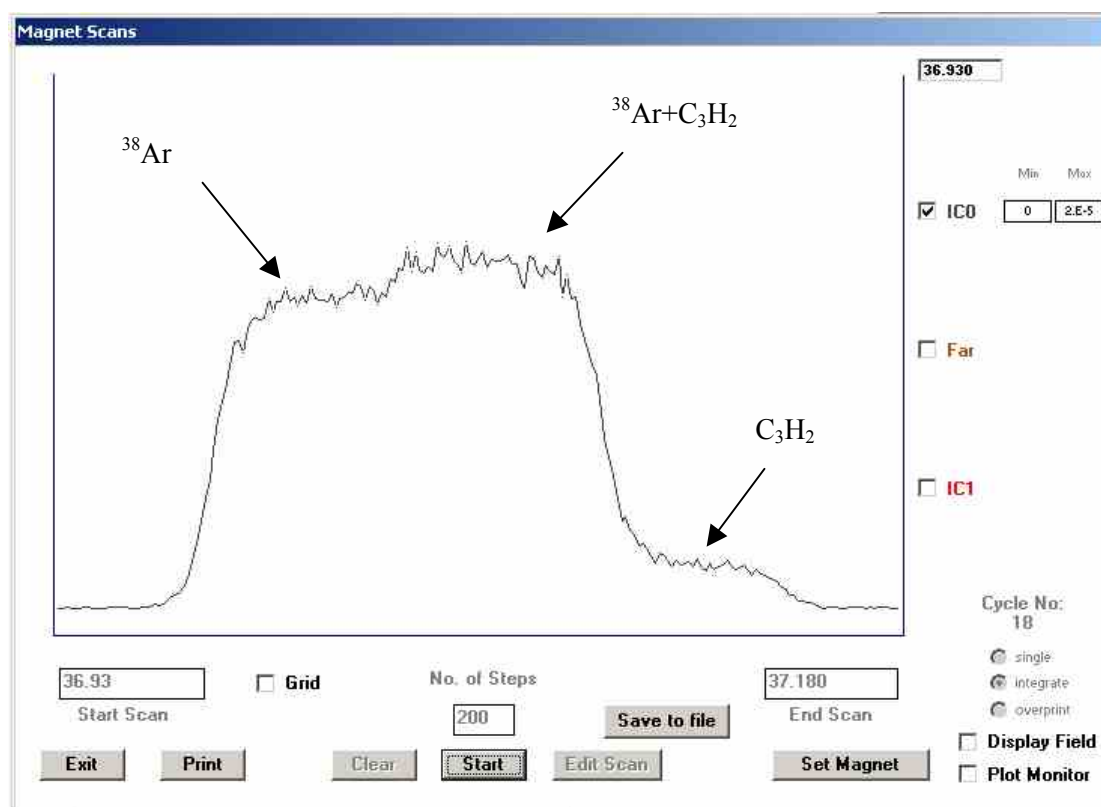
1. The signal starts to rise as the first beam crosses the left-hand edge of the slit.
2. The recorded beam then stays constant as the beam traverses the slit
3. The recorded beam then continues at the same intensity as the beam continues to traverse the (now wider) slit, until the second beam just starts to enter on the left.

4. The signal starts to rise again as the second beam crosses the left-hand edge of the slit
5. The recorded signal stays constant at a value equal to the sum of the two intensities
6. The signal falls to the --- we let the reader continue the sequence.

The important point between these two schemes however is point (3), where in the second case we have GAINED an extra region of peak flat. In practice we may not use this extra flat, since we can make use of the difference in two ways:

- we can “separate” two beams, which are closer together or
- we could use a wider source slit (increase the value of  $x$ ) and still resolve the two peaks.

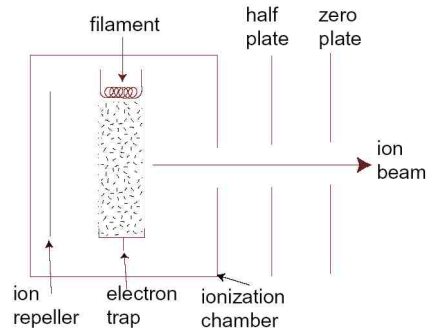
The first example gives higher effective resolution whilst the second increases sensitivity.



We illustrate this approach with a spectrum of  $^{38}\text{Ar}$  and the overlapping  $\text{C}_3\text{H}_2$  interference, obtained by using a (very) small air shot. The required resolving power required to separate these two beams is 717, and they are seen to be easily resolved, with plenty of flat to perform the Argon measurement.

### Source considerations:

The ion source supplied with the Noblesse mass spectrometer is a modified Nier design. The ionisation process is undertaken by the electron beam, produced using a heated tungsten filament. The electrons are sent through the source region and the

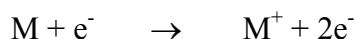


majority collected on the electron trap. We monitor the total emission, whilst stabilising the trap current. The difference in these two values are “lost” electrons, but in practice they may be traversing the source chamber region, but not be detected by the trap, since they are deflected by the strong electrostatic fields used to extract the ions. Due to the different charge on the electrons and ions, these fields have the effect of repelling the electron beam back into the chamber. If

nothing further were done to control the position of this beam, probably none would make it across to the trap. To minimise the effects of the extraction field, a transverse magnetic field is added, parallel to the electron beam direction. This causes the electrons to spiral in the field, and helps constrain the beam. Some people believe that the magnetic field also increases the path length of the electrons, as they transverse the excitation region, but a simple calculation, which we leave to the reader, will show that this is an extremely minor contribution.

The loss of some of the electron beam from being collected by the trap, makes it difficult to compare the relative sensitivities of different designs of source. Obviously some of these “lost” electrons can traverse the ionisation region and hit either the repeller or the inner walls of the source box. As such they can ionise sample atoms, contributing to the recorded ion signal. Because of this problem, a better way to compare sources is ion signal is to use the total electron emission, rather than the trap current.

The electron beam ionises any particle in the source box by a reaction, which can be represented by:



The probability of this reaction depends on the relative energy of the incident electrons. Studies have shown that this probability rises from zero below the ionisation energy of the species M, then rises to a maximum at about 70eV (for most species) before slowly falling again as the relative energy is increased.

Behind the electron stream is placed a small electrode, called the “repeller”. Again it is conventionally thought that this helps to repel the ions out of the source chamber, but since the voltage applied (for maximum observed ion signal) is usually negative, it will be seen that this simple explanation is superficial. In practice its effect on the electron beam cannot be ignored, whilst it is also helpful in compensating for the large field gradients caused as the extraction field penetrates the ion box through the ion exit slit.

The source box region is maintained a positive high voltage (3 to 8kV) with respect to the rest of the mass spectrometer. This draws out the positive ions from the source region, and accelerates them in the forward direction. The design of extraction lenses used in the source fitted consists of a set of extraction electrodes immediately after the source block region (shown as the “half plates” in the diagram above), followed by an earthed slit (“zero plate” in the diagram) further downstream. The half plates have a

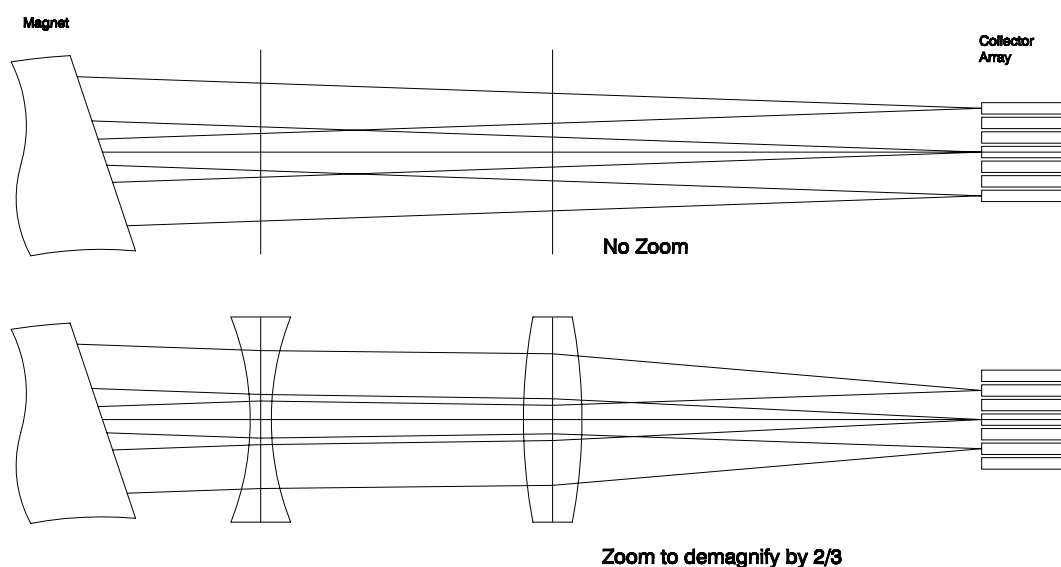
dual role. They are used to provide the extraction voltage to pull the ions out of the source block, whilst since a small difference in voltage between the two halves can also be applied (hence their name), beam steering in the horizontal direction can also be achieved. It can be shown that the field produced by the half plate and the other elements, focuses the ion beam to a narrow “virtual image” in the region of the zero plate, which acts as the source image for the mass spectrometer. This “virtual image” is in fact extremely narrow, and is in practice normally narrower than the actual slit width employed in the zero plate. If you are interested, its width may be estimated by measuring the width of the slope of the recorded peak shape, as indicated above.

Since the extraction of the ions from the source box depends on field penetration through the ion exit slit, the tuning tends to depend on the absolute value of the half plate voltage to the source block potential. Thus as the absolute HT of the source block is altered (say from 4kv to 6kv), it will be seen that the absolute value of source HT value less the half plate voltage remains approximately constant. For this reason we do not refer to the half plate voltage as a percentage of HT, as do other manufactures, since again this is not a useful parameter.

Finally, beyond the zero plate (and not shown in the above diagram), we have placed a pair of vertical lenses. These provide both steering in the vertical plane as well as some vertical focussing. It can be shown that the sensitivity of the source does not increase much as the source HT is increased above about 4kv. If the ion beam was over-filling the aperture in the flight tube (which defines the horizontal and vertical acceptance angle of the mass spectrometer), it would be expected that the signal should increase as  $HT^{1/2}$ , as discussed earlier. Since the gain in signal is less than this theoretical value, it will be seen that the design nicely matches the mass spectrometer optics, and it will not be surprising that the vertical lens has little effect.

### The Zoom Optics:

The provision of the two quadrupole lenses which together act as a zoom lens, (similar to a zoom lens on a camera) can alter the image magnification at the detector





array, which means that, for this instrument, the dispersion is no longer a constant term. The operation of the lenses is shown above, in which we consider two cases of a set of ion beams incident on an array of detector elements.

Firstly exact coincidence with every third collector of the array is assumed without any required “assistance” from the zoom lens. In the second example the zoom lens is activated to produce a demagnification of  $2/3$  so that the beams are now coincident with every second collector. Although the user may just utilise a lookup table to find the settings required for any particular element, it is perhaps worth while in having some feel for the process involved.

Comparison with conventional light optics will probable assist here. To produce a demagnification of  $2/3$  of an image would result in the acceptance angle at the image plane increasing by 1.5 (angle x magnification is constant). Thus if we have to increase the acceptance angle on the detector array, we should try to open out the beam at the first lens and then refocus it back down with the second. This is achieved in light optics with a concave lens followed by a convex one, as shown above. To open out the ion beam we must attract the ions away from the centre path by the first lens, which means (for a positive ion beam) that we apply a negative voltage to the outside plates of the first zoom lens array stack. To focus the beam back at the second stack we must repel the ions back towards the centre, and hence a positive potential should be applied to the outer plates of the second zoom element.

## Chapter 2: Electronic units; general discussion

A number of special designed and built electronic units are used on this instrument.

Communication between the controlling PC and any unit on the instrument is directed via the system micro (or system control), which contains a microprocessor card and two serial communication cards. Communication between this unit and the PC is via standard RS232 protocol, at 58.6k baud. The communication between units within the instrument itself is via RS422 protocol. The pin-out of the 9 way 'D' connectors have been chosen to permit simple ribbon cable wiring to be used and is given in the chapter describing these units in detail.

Each serial card has eight outputs; thus up to 16 devices can be connected before an extra card is required. The assignment of the outputs to the various units is, like all the assignments, configurable in software, and is discussed fully later.

The other units present in the instrument are:

- Twin channel digital voltmeter: Used to record the output from the Faraday collector(s).
- Ion counting unit: Provides up to four independent preamplifiers, discriminators and counters for the ion counting detectors.
- 1000v deflection units: Two are supplied as standard with each instrument. They each have 16 channels of programmable +1000v to -1000v output and are used for all deflectors, the voltages on the quad lenses, and the suppressor voltages for the Faraday array. Assignment of outputs is software programmable.
- Magnet slave: With the Hall probe and magnet control (mounted on the back of the magnet) used to control the field in the magnet. Is totally bipolar, although this feature should not be required here.
- Magnet control: Contains a precision D to A and hall probe amplifier, to control the magnet field.
- Utility Unit: Provides a 24 volt and +/-15 volt ring voltage for the instrument together with providing the interface between the various gauges and relays and switches used.
- Preamplifier Bin: Houses the Faraday preamplifier(s) and the connection circuitry for the multiplier units.
- High Voltage supply: Provides the source HT and half plate (extraction) voltage together with three independent multiplier high voltage supplies. A second unit may be present if a retardation filter is fitted to the instrument.
- Source supply unit: Contains all the filament supplies and related source output. Can be controlled either from the front panel or via the PC.
- Ion Pump Supplies: Two SPC units are fitted as standard, the pressures being read by the PC.
- Getter Supply: For activating, and reactivating the SAES getter fitted to the instrument.

The system control undergoes a code loop every 10 milliseconds, whereby the status of any input can be monitored. This process is used to provide the trips on the instrument, and as such extra trip wiring is not required since the serial highway is utilised for this process. This approach also permits the addition of extra trips or the modification of existing ones to be easily undertaken. **Nu Instruments cannot however be responsible if damage is caused by the alteration of this code. If you wish to alter this code please talk to us first!**

## Pneumatic Panel

The two automatic vacuum valves (Manifold to Mass Spectrometer valve and the Source Ion Pump valve) supplied as standard on the instrument, are controlled by compressed air. The solenoid valves to provide this control, and the regulator to supply them with a constant pressure gas supply, are mounted in the end of the source bay of the instrument sub bench. The recommended pressure for the regulator is 50 to 60 psi.

All the solenoid valves are fitted with LEDs to allow the state of the control to be easily seen. Mounted on a 5-station manifold, the assignment is as follows:

1 (Rear of instrument)	Manifold Valve
2	Source Ion Pump
3	Not Used
4	Not Used
5 (Instrument front)	Not Used

The distribution block, beneath the solenoid manifold, contains wiring for the two supplied outputs, and four spares. If these are required to control the users manifold valves, the extra solenoids can be added to the manifold or placed elsewhere.

## System Philosophy

The separate units are designed to be device (chip) independent, such that if the present generation of devices used become unavailable, a replacement unit can be easily designed. As such each unit is a small discrete entity, with a well-defined task, which communicates with its colleagues over a simple, industry standard RS422 serial bus. Since the signal levels for this bus are defined, but the wiring assignments for the connectors are not, we have chosen to implement our own, elegant solution here, such that the interconnecting cables can be produced using a ribbon connector, for reliability and simplicity. The pin out for this RS422 port together with all other connector assignments are given below, so that all the connections for the electronic unit connections are to be found in this chapter.

The serial bus used is a star configuration, with the controlling PC talking to the system micro via a RS232 port, and a series of independent serial ports then communicating with the individual units. The system micro provides an area of shared memory for each port, so that the data to be sent out to the remote module and the data to send back, can be accessed both by the serial drivers and the micro itself, enabling data transfer from and to the user's PC. Communication between the PC and the system micro is via a well-defined protocol (discussed later) and is implemented

in software, rather than rely on a hardware specific version. Simple commands are sent to the micro from the PC, which then decodes them and the communication with the separate units becomes transparent to the user.

We will discuss each unit separately, at a level so that the user can understand the role and capabilities of each device. We also list some of the commands which are employed to control each device, although this is in practice the high level language which is then decoded by the serial card drivers, and give details of the connections for each unit. We will not go into enough detail to enable the units to be serviced. It is the philosophy to provide redundancy in outputs together with reliability and if a unit should fail, we would expect to replace a board or the whole unit, rather than undertake field service.

### System Micro

This unit contains three main parts, a back plane into which a series of 4U high cards may be placed, the micro card itself and a couple of serial cards. The unit is powered by 240 v AC and is fitted with a third party switched mode power supply.

Micro Card: This fits into the left most slot (identified as slot zero in the assignments). This slot is unique on the backplane and must be used for this card. The present generation is fitted with a 68EC020 microprocessor due to its reliability and freedom from bugs. There are three ports on the rear of the card, a RS485 used for development work (9 way 'D' male plug), a RS232 port used as the connection to the PC (9 way 'D' female socket) and a 25 way 'D' female socket also assigned for development purposes. The connection cable to connect to the PC is wired as follows:

<u>PC Connector (9 pin female)</u>		<u>System Micro End (9 pin male)</u>
Pin 5	← →	Pin 5 via the shield
Pin 2	← →	Pin 2
Pin 3	← →	Pin 3
Pin 7	← →	Pin 7
Pin 8	← →	Pin 8

Pins 1, 4 and 6 are also joined together at each end, and pin 9 is linked to pin 5 at the PC end.

Serial Card: Two are supplied as standard with each instrument. Each card provides eight RS422 bi-directional ports using 9 way 'D' male plugs. Port 0 is on the bottom left of the rear panel, port 1 above it etc. The panel contains the port identification silk screened on it.

The assignment used for the RS422 connector is:

Connector number	Assignment
1	Rx+
2	CTS+
3	Earth
4	RTS+
5	Tx+
6	Rx-

7	CTS-
8	RTS-
9	Tx-

The software drivers for both types of card are, as described later, downloaded from the PC if required, and are not permanently resident on the boards.

### DVM Unit

Contains two precision analogue to digital converter channels. The unit is powered from 240v AC and contains a purpose built linear power supply to minimise noise pickup by the precision components.

The rear of the unit contains a RS422 port (9 way 'D' male plug) to connect to the system micro serial cards and a development port (9 way 'D' female socket), which is not available for general use. Connection to the A to D converter inputs is via a 37 way 'D' female socket. The assignments used on this connector are:

1	Channel 0 +
20	Channel 0 -
2	Channel 1 +
21	Channel 1 -
16 + 34	+5 volts
17 + 35	+15 volts
18 + 36	0 volts
19 + 37	-15 volts

The following software control is provided.

Writing an integer value (up to 256) to the address *nSamples* of the serial port driving the unit will produce an integration period equal to this value divided by 10, i.e. the smallest integration period is 0.1 secs.

Setting *Reset\_F* to "true" (all bits set) will reset the unit. The flag value will change to "false" after the command has been accepted.

Setting *Calibrate\_F* to "true" will cause the unit to undergo a self-calibration cycle on all channels. The inputs will be disconnected internally during this process.

Setting *Restart\_F* to "true" will cause the present integration to cease after the next 0.1 second time slot and a new integration cycle to commence.

After a new set or integration data is received by the system micro, the value stored in the memory location called *Sample\_No* will be increased by one. The values cycle around a 32 bit word, and so checking that merely the value has changed by unity over the previous value can cause problems.

The observed integrated beam values are stored in the array *V10* to *V11* in the shared memory in the system micro. The value returned corresponds to a 24 bit number for a +12.5 to -12.5 volt dynamic measurement range.

### 1000 volt unit

Provides sixteen independent programmable dual polarity 1000 volt outputs. Powered by the 24v supply, the unit has its own switched mode voltage converters. **The voltage inside this unit could be lethal and so it should not be opened for service work.** The units can be switched on and off from the mains distribution panel.

The 16 outputs are assigned by labels on the rear of the unit, output 0 being closest to the 24 volt power input, output 15 furthest. Also present on the rear panel is a 9 way 'D' plug which provides the RS422 connection to the system micro, and an earth stud.

The 24v connector has the following connections:

Pin 1	Ground
Pin 2	+24 volts
Pin 3	0 volts.

(Note however that pin 3 lies between pins 1 and 2 for this connector.)

The units may be controlled in a similar manner to the analogue outputs and inputs of the multiple IO card (see below). Thus sixteen analogue output channels *VO0* to *VO15* can be sent data to define the outputs, whilst the corresponding sixteen analogue input channels *VI0* to *VI15* read the actual output voltages via sensing resistors. The string sent to any channel is a integer of value ten times the required output (including sign). Thus a resolution of 0.1volt is possible. The corresponding values may be read back to monitor the outputs.

### **Magnet control units**

The magnet control consists of three separate parts. The Hall probe, which is placed in the centre rear gap of the magnet poles, has its own in-built heater and platinum resistance thermometer. This connects to the magnet control unit via a 9 way 'D' plug. The unit is mounted on the rear of the magnet yoke, so as to minimise the possibility of interference pickup of the small controlling signals, and contains, not only the Hall probe reading and temperature control circuit, but also a precision 20 bit DAC and associated circuitry to enable the magnet field to be set to the required value.

This magnet control unit is also connected to the magnet slave, which is mounted in a 19 inch rack section, via a 25 way 'D' to 25 way 'D' cable, which is wired pin to pin. (In practice less than 25 connections are required so not all pins need be connected) This cable carries power, digital and control signals to define the output requirements for the slave. Communication between the system control and the magnet control circuitry is via the RS422 connector mounted on the rear of the magnet slave unit. Other connectors at the rear of this unit are the mains in (240vAC) and the magnet power connectors. These are polarised and the positive connection (Red) should go to the upper of the two magnet coil connectors, with the negative output going to the lower.

The complete magnet control is bipolar, and is capable of providing a + or - 25amp current through the magnet coils. This design approach is necessary, not because there are negative ions to study , but because the voltage overhead will define the speed of

complete magnet system, and the negative drive capability will permit a fast decrease in magnet field, just as the positive overhead permits a fast ramp up in field.

The magnet may be set by sending a long (32 bit) word to the relevant port memory location corresponding to the DAC output required (a signed value). The calibration of the DAC to observed magnetic field is achieved by the PC control software, and is completely transparent to the system micro.

The slave unit contains an EEROM with the calibration constants necessary for unit running. The assignments of these constants are given below, in the form of local address and their approximate value:

Address	Value	Description
0	0	Default error flag state. Normally this should be set to 0.
1	2000	Scale factor for VCC calibration
2	7400	Scale factor for HV+ (+30V)
3	7400	Scale factor for HV- (-30V)
4	4010	Scale factor for LV+ (+12V)
5	4010	Scale factor for LV- (-12V)
6	4010	Scale factor for +15V analogue supply
7	4010	Scale factor for -15V analogue supply
8	2000	Scale factor for temperature sensor 1
9	2000	Scale factor for temperature sensor 2
10	2000	Scale factor for demand voltage
11	7400	Scale factor for Amplifier output voltage
12	7400	Scale factor for Buffer output voltage
13	7290	Scale factor for Buffer output current
16	550	Max. allowable VCC voltage, 550 = 5.50V
17	450	Min. allowable VCC voltage, 450 = 4.50V
18	1600	Max. allowable +15V analogue supply voltage, 1600 = 16.00V
19	1400	Min. allowable +15V analogue supply voltage, 1400 = 14.00V
20	1600	Max. allowable -15V analogue supply voltage, 1600 = -16.00V
21	1400	Min. allowable -15V analogue supply voltage, 1400 = -14.00V
34	8000	Max. allowable heat sink temperature, 8000 = 80.00 °C
35	4000	Max. allowable output current, 4000 = 40.00A

A utility is provided in the software to alter these parameters.

When in use, the unit performs checks to ensure that it is within a safe operating mode, and if it is not, it will limit its output or shut down, so as to protect itself.

## MIO Cards

There are a number of these “**M**ultiple **I**nput and **O**utput” cards placed around the instrument to control various units, each featuring sixteen of;

**16-bit analogue output.** Outputs are bipolar and can drive between +10v to -10v.

**16-bit analogue input.** Inputs are bipolar and can monitor between +10v and -10v.

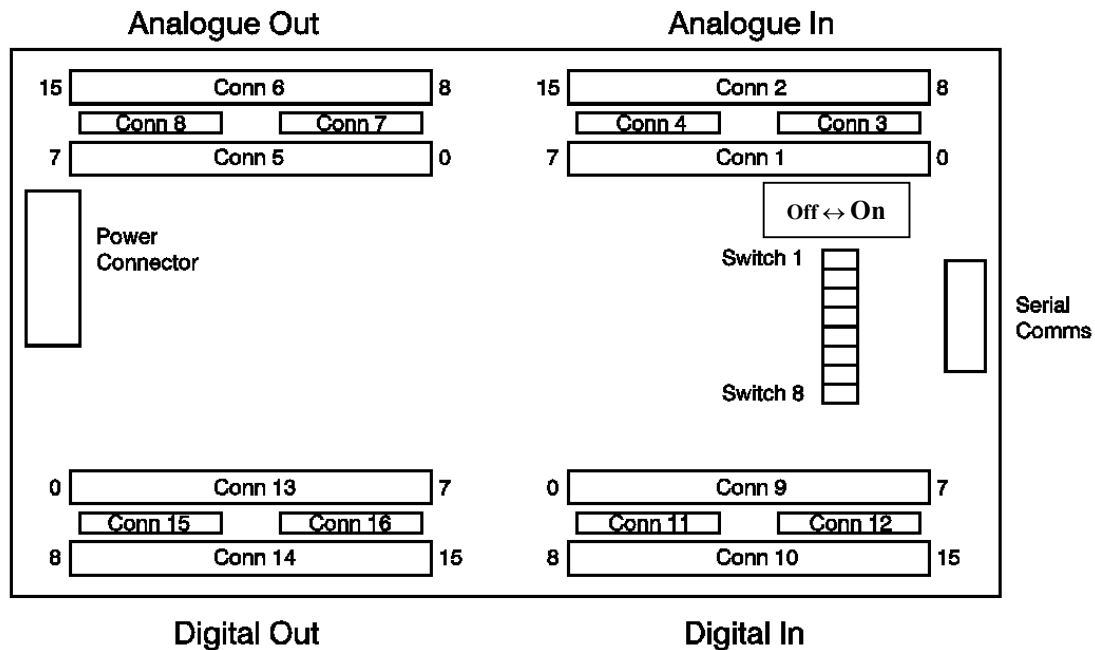
**Digital output.** These are FET devices that will pull down to (digital) ground when enabled.

**Digital input.** These are optically isolated inputs designed to switch at about 2.5 to 3 volts. These inputs are protected to approximately 50 volts.

The cards require a  $\pm 12$  to  $\pm 15$  volt supply for operation and have a built in +5 volt regulated output for board use.

<b>Pin 1</b>	- 12V supply
<b>Pin 2</b>	Analogue ground
<b>Pin 3</b>	Analogue ground
<b>Pin 4</b>	+ 12V supply
<b>Pin 5</b>	Digital ground
<b>Pin 6</b>	+5V output.

Connection for the serial control of the card is via a 10-pin IDC header. This is normally wired pin to pin (pin 1 to 1) to a 9 way 'D' male plug (pin 10 not connected) to enable a standard RS422 connection to the system control.



Outputs are in four banks along the long edges of the card. Each connector will service either eight inputs or outputs, and each connection has an associated earth pin. The socket positions are identified on the board. DO0 is the digital out port zero, DO7 the digital out port 7. Both types of output connectors are shown, although only one type is fitted on a single card.

In the top centre of the board is an 8-way switch bank.

### Switches

1-3 Define the address of the board in binary. Thus if switch 1 is on the address is 1 etc. The board when used on its own (i.e. is not used in a



- multi-drop mode) should have the address set to 1, otherwise it will correspond to the multi-drop address.
- 4-7 Defines whether the digital outputs are to be used in straightforward switch manner, or to control stepper motors, with groups of four outputs ganged together. Thus if switch 5 is set, it will define that the digital outputs 4 to 7 are to control a stepper motor, whilst the remaining ports act as conventional and independent switches.
- 8 Set the terminating resistor for the digital RS422 connection. Should be set if the board is used on it own, or if the board is the last in the line of a multi-drop set.

### Software control.

In the simple 16 channel independent control input and output modes, we merely have either to write (for control) or read (if used as an input) from the relevant micro memory location. These locations are labelled *AIO* to *AI15* for the analogue input monitoring ports and *AO0* to *AO15* for the analogue outputs. A 16 bit word is used such that the value zero corresponds to a zero input or output, 32767 to a 10 volt and –32767 to a –10 volt signal. The scaling is voltage thus the responsibility of the PC control software. To set a digital output, one must set the relevant micro memory address (*DO0* to *DO15*) to “true”, whilst a “false” value (all bit equal to zero) will turn the switch off. The corresponding values are reported for the digital inputs (*DI0* to *DI15*).

If the cards are used in a stepper mode the following commands are used:

- Disp\_0\_A* Sending a value of *X* (16 bit) to this address will cause the motor on the first port (corresponding to the digital output ports 0 to 4 in this example) of the first MIO card of a pair of a multi-drop set (i.e. card A) to move by *X* steps. This is a signed value, to enable control of the motion of the motors in either direction. The card supports half stepping, and as such for the 200 steps per revolution motors used, a value of *X* of 400 will result in the motors turning one complete revolution. The software driver will reset the value of this memory location to zero after it has downloaded the data to the card.
- Moving\_3\_B* Set to “true” by the driver if the motor on port 3 of card B (in this example) is moving, “false” otherwise.
- Stop\_F\_1\_A* Setting this memory location to “true” will cause the motor on port 1 of card A to stop moving.

As will be seen, there is no provision in this command set to keep track of the absolute positions of the stepper motors This is because it is a trivial task for the system control to undertake this during its 10milliSec code loop.

The MIO cards are updated by the system control at the 10milliSec rate, although the ports are updated in a cyclic manner to minimise system overhead. If the card is switched off, or otherwise disconnected from the system control, after reconnection the board outputs will be reconfigured to their original settings.

## Utility Unit

This unit house the 24 volt and +/-15 volt power supplies used to provide the power for valves, 1000 volt units and the various MIO cards within the instrument frame. It also provides a general monitoring and control point for many of the smaller devices on the instrument.

The two power supplies are wired to the 230v AC input, and connect their outputs via a 3-pin XLR socket for the 24V supply and 5-pin XLR socket for the  $\pm 15V$ .

24V 3-pin connector:

<b>Pin 2</b>	+24 volt
<b>Pin 3</b>	0 volt.

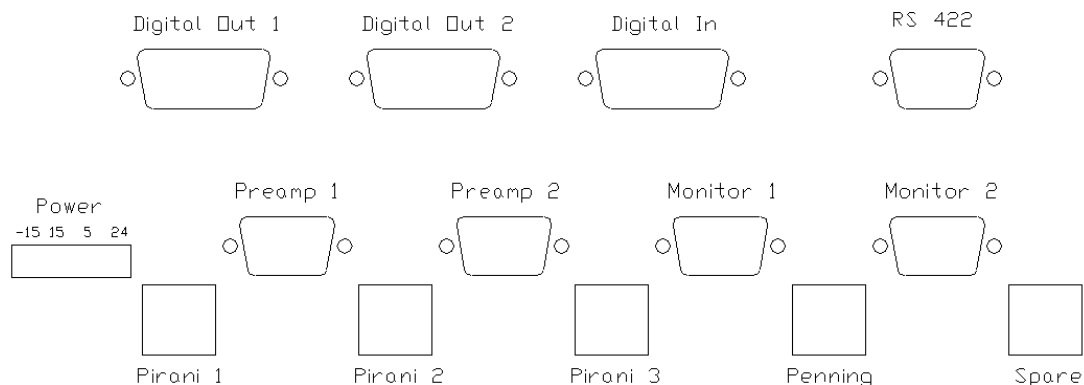
Note that pin 3 lies between pins 1 and 2 of the 3-pin connector.

$\pm 15V$  5-pin connector:

<b>Pin 1</b>	0 volt
<b>Pin 3</b>	+15 volt
<b>Pin 5</b>	-15 volt.

The unit contains a small PCB used to simplify the interconnection between a number of output connectors and the MIO card. Four LED's indicate the status of the power supply outputs, the 5v output being derived from the MIO board.

The five square 8-way RJ45 connectors are inputs for the active gauges on the instrument with one spare. The Penning connector provides a switched output. The pin connections of these sockets can be found from the Edward's manual.



**Preamp1+2** 9-way 'D' socket, provide supply to power a small remote operational amplifier device, and an input to record the observed signal. Connections are as shown:

<b>Pin 1</b>	+ Signal
<b>Pin 2</b>	- Signal
<b>Pin 7</b>	+15 volt supply
<b>Pin 8</b>	0 volt supply
<b>Pin 9</b>	-15volt supply.

**Monitor1+2** 9-way ‘D’ plug, provide some extra general analogue monitoring capability for the unit.

<b>Pin 1</b>	Analogue signal input 1
<b>Pin 6</b>	Analogue ground
<b>Pin 2</b>	Analogue signal input 2
<b>Pin 7</b>	Analogue ground
This pattern repeated through!	
<b>Pin 5</b>	+5 volt supply.

The assignments currently used are:

<b>Monitor 1</b>		<b>Monitor 2</b>	
<b>Pin 1</b>	Turbo 1 speed (+)	<b>Pin 1</b>	Unassigned
<b>Pin 6</b>	Turbo 1 speed (0)	<b>Pin 6</b>	Unassigned
<b>Pin 2</b>	Unassigned	<b>Pin 2</b>	Unassigned
<b>Pin 7</b>	Unassigned	<b>Pin 7</b>	Unassigned
<b>Pin 3</b>	Unassigned	<b>Pin 3</b>	Unassigned
<b>Pin 8</b>	Unassigned	<b>Pin 8</b>	Unassigned
<b>Pin 4</b>	Turbo Temperature sensor (sig.)	<b>Pin 4</b>	Unassigned
<b>Pin 9</b>	Turbo Temperature sensor (gnd.)	<b>Pin 9</b>	Unassigned
<b>Pin 5</b>	Turbo Temperature sensor (+Vs)	<b>Pin 5</b>	Unassigned

Two 15-way sockets provide digital output switching signals; one 15-way plug provides digital monitoring capability. The assignments of these connectors are given below, where **pin 9 to 15 is assigned to the digital earth.**

### Digital Out

<b>Digital Out (1)</b>		<b>Digital Out (2)</b>	
<b>Pin 1</b>	Manifold Valve	<b>Pin 1</b>	Unassigned
<b>Pin 2</b>	Source Ion Pump Valve	<b>Pin 2</b>	Unassigned
<b>Pin 3</b>	To Bake-out Enable Relay	<b>Pin 3</b>	Unassigned
<b>Pin 4</b>	Turbo Isolation Valve	<b>Pin 4</b>	Unassigned
<b>Pin 5</b>	Spare Valve 2	<b>Pin 5</b>	Unassigned
<b>Pin 6</b>	Spare Valve 3	<b>Pin 6</b>	Unassigned
<b>Pin 7</b>	Spare Valve 4	<b>Pin 7</b>	Unassigned

### Digital In

<b>Pin 1</b>	Monitor “Manifold In” valve state
<b>Pin 2</b>	Monitor “Source ion pump” valve state
<b>Pin 3</b>	Monitor 1kv units Power Switch
<b>Pin 4</b>	Monitor High Voltage Power Switch
<b>Pin 5</b>	Monitor “Turbo Isolate” valve state
<b>Pin 6</b>	Unassigned
<b>Pin 7</b>	Unassigned

## Filament Supply

This unit has been specially designed using the latest technology. The filament is heated using a high frequency AC current, and the emitted electron current monitored via the source trap, which is used as feed back to control the power being applied to the filament. The maximum output power may be set using the PC, a facility that is useful during instrument bakeout when the collimating magnets have to be removed, resulting in poor regulation if the standard trap current method is utilised. The unit also provides the voltages to float the filament from the source block, the repeller voltage and the (delta) half plate steering voltage. The actual voltage of the trap to the source block may also be altered, although it is not recommended that this be done.

**Please note that the voltages inside this unit are lethal, even if the source HT and half plate extraction voltages are set to zero, and the lid should not be removed.**

The actual source HT potential and half plate (extraction) voltage are supplied by the separate Supplies unit, and hence controlled separately.

The unit also monitors the drive power to the filament, and the total filament emission current, both being displayed on the front panel as well on the controlling computer screen. The actual power shown is the total drive power being supplied by the unit, this being made up of several parts:

The power dissipated in the filament supply unit itself (both the low voltage and high voltage sides)

Any losses down the connecting wires – which obviously can become quite considerable if long leads are used.

The power used by the filament itself.

As such, please be careful when comparing the recorded figure between instruments.

The input and output of the unit consists of 10kv rated SHV connectors. **Under no circumstances should the unit be run with the connectors disconnected.**

The system micro monitors the set and actual required outputs for this unit (as with most other units) and if these disagree, the controlling PC will flag a warning. This is done in case the unit is taken out of remote control (i.e. the PC can no longer control it) so as to indicate that any runs undertaken may not be at the settings displayed.

To enable the unit (assuming it is powered on by the “Source” switch on the mains distribution unit being enabled), double press the knob on the front panel. If the PC has been set to “remote control”, the display should flash “Remote”, otherwise it is under local control. If local control is enabled, the various setting may be altered by:

1. Selecting which value requires changing by turning the rotary knob, and then depressing the knob once. The selected value will the flash.
2. Change the value by turning the knob
3. The value may be set once the knob is depressed again.

To disable the unit, either switch off at the “Source” switch on the Mains Distribution Unit, or depress the rotary knob on the front panel for about 3 seconds.

## **Source supplies unit**

This comprises of a number of, industry standard, high voltage bricks, controlled using an MIO card. The input for this unit comprises of a  $\pm 24$  connector, supplied from the utility unit. To overcome earth loops, the  $\pm 12$  volt required to power the MIO card is derived via a dedicated DC-DC converter within the unit, rather than from the utility unit output.

The unit is fitted with two relays, which allow the source high voltages to be controlled independently from the multiplier voltages. These relays switch the 24v power to the relevant bricks, and are controlled via the digital output ports of the MIO. The analogue output control of the card controls the actual high voltages of the bricks, whilst these outputs are independently monitored via the analogue input ports.

## **Ion counting Unit**

Each unit may be fitted to monitor the output from up to four ion or electron multipliers. Each channel has its dedicated preamplifier board, whose input is via a 50ohm BNC connector, protruding from the units back panel. These preamplifiers plug into the unit motherboard, and so may be simply replaced if one should fail (a rare occurrence). The output of the preamplifier is then routed via a fast discriminator, and then into the counter array. The discriminator is completely bipolar and set via the controlling PC. Care is therefore required to insure that you set the correct polarity for the discriminators (if you should need to change them). For the multiplier arrangement used on the instrument, the discriminators should be set with a NEGATIVE threshold, corresponding to a pulse of electrons.

The unit monitors each output of each counter to see if the signals correspond to a value greater than a given pre-set value, defined by the user via the PC program or the code in the system micro. We conventionally set this value to be  $10^7$  cps, corresponding to  $10^5$  counts in each 10 millsec time slot. If this value is exceeded on any of the four channels a trip flag is set, which is used by the system micro to protect the multipliers (by deflecting off the beam). This is the only flag that is set without checking for validity (i.e. by seeing if it is repeated), and as such may be set by outside electrical interference, rather than a true excess beam. Once set the controlling PC program will indicate a trip, and show the multipliers as “protected” in the relevant window (see below).

## **Mains Distribution Unit**

This controls all the power to the instrument, as well as being the area from where the bakeout may be set. The input power is separated into two lines (Labelled “Instrument” and “Bakeout”), to enable the provision of a non-interruptible supply for the main instrument control and vacuum power, if required. The various regions of the instrument are switched separately, as indicated by the front panel labelling. Please note that the turbo will only come on if the rotary is powered (i.e. switched on), which may cause momentary confusion if you are using a separate rotary to back the turbo.

The bake out control is set from the front of this unit. The bakeout is split into three regions, source, flight tube and collector, each with its own thermocouple sensor and

power controller. The source and collector thermocouples are permanently fitted to the instrument through the bench, and connected to the unit at the side. The flight tube sensor, which should be fitted to the flight tube with tape when the magnet has been retracted, is connected via the unit's front panel.

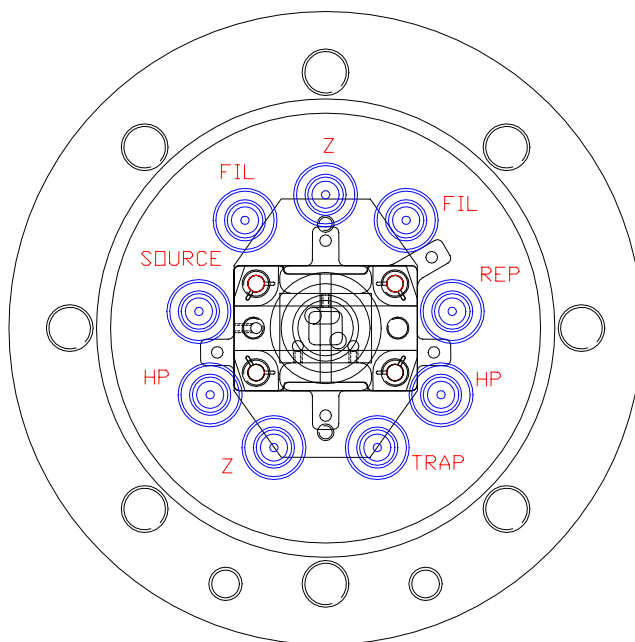
Although powered via the switches on the front of the unit, there is only any power to the heaters when the relevant output in the utility unit is set. This facility allows the PC to control when the bakeout is ended, thus allowing the system to cool down before the user arrives, if so desired (see below).

### Getter Supply

The use of this unit is described fully in Chapter 9.

### Source wiring connections

These are shown in the diagram, which shows the view from the source.



## Chapter 3: Micro Code description

The system control may be programmed by down loading of code from the PC. A complete set of tools, including a dedicated compiler, is provided for this purpose.

The downloaded code consists of a number of parts, first of which are the drivers for the various system control devices and outputs. Thus a driver exists for:

- The CPU card
- The MIO interface cards
- The HV1000 units
- The DVM unit
- The Ion Counter unit
- The Magnet Slave unit
- The Serial Output cards
- The SPC (Ion pump controller)
- The Filament unit
- A general serial communication utility

Other drivers may be added as necessary.

This approach enables updates to be undertaken without hardware modification and ensures new devices may be easily added to the instrument control suite at a later date without having to modify the system micro.

Next comes an assignment section, in which the various high level names for the various outputs and inputs, as recognized by the PC control software, are assigned to the various ports of the instruments. An example of this code is given below:

```
driver CPU is "cpu_v101.os"
driver RS485A is "Dd485a_8.os"
driver MIO is "Dmiov101.drv"
driver MIO2 is "Dmmio104.drv"           // multi-drop MIO
driver HV1000 is "Dhv_v100.drv"
driver DVMB is "Dvm_v303.drv"         // DVM
driver IC is "dionv100.drv"           // Ion Counter
driver MAG is "Dmagv200.drv"          // Magnet Slave
driver Para is "Dps_v100.drv"         // Diagnostics
driver IONPUMP is "Dspcv103.drv"      // SPC Ion pump Controller
driver SerialSend is "DSERV100.drv"   // General serial driver
driver Source_Supply is "Philv100.drv" // Source supplies
```

*// Now, define the slots*

```
slot 0 is CPU           // The main OS
{
```

```
vars
  Status is CPU_Stat
```

```

}

slot 1 is RS485A    // This is an 8-way serial card
{

vars
  Status is SS
  PortActiveFlags are PA_F
  PortErrorFlags are PE_F

port SIO0 is DVMS
{
vars
  Status is U0_Status
  nSamples is Num_Samples
  Reset_F is Reset_DVMS
  Restart_F is Restart_DVMS
  Calibrate_F is Calibrate_DVMS
  Sample_No is DVM_Sample_No
  VI0 is DVM0
  VII is DVM1
}
}

```

Etc etc.

The first part of this example (e.g. *driver CPU is "cpu.drv"*) permits the compiler to assign the required driver files for downloading to the system control. Next we assign the physical slots of the micro crate to their occupants (e.g. *slot 0 is CPU* , *slot 1 is RS485A*). This states that the first slot of the crate houses the CPU board, whilst the second houses a serial card.

Next are some status flag assignments, and finally in this example we assign the ports of the first serial card to identify what it is connected to (in this case port number zero is connected to the DVM unit).

Under the section labeled “vars” come the assignments, which relate micro addresses to PC understood, high level, names. Thus in the computer program controlling the instrument, variables such as “DVM0”, DVM1” are employed (they are the DVM reading of beam 0 and 1 etc) as well as variables such as “Calibrate\_DVMS” (used to recalibrate the relative gain of the DVMS). A fuller description of the commands associated with each unit is provided in the section describing the unit in detail. Note that the compiler is case sensitive and that *Dvm* is not the same as *DVM*.

A section where code can be written which is serviced every 10 milli seconds by the system micro then follows. An example is given below:

*code*



```

long Digital_state = 0      // contains all switch info
long Error_state = 0       // contains error status
//

Digital_state &= #40000000 //reset to zero apart from trip bit

Digital_state |= (To_Manifold and #00000001)
Digital_state |= (Source_Ion_Pump and #00000002)
Digital_state |= (Remote_Control and #00000004)
etc etc.

```

This section starts with the keyword “code” and in this example is followed by some assignments. The language is integer based and may contain long (32 bit) or word (16 bit) parameters. Since the processor is 32 bit, the long computations are actually accomplished faster than 16 bit ones. The parameters such as “Error\_state” may be directly accessed by the PC program code. The final part of the example illustrates the general format of the language, which has been optimized to permit rapid execution. The first statement

```
Digital_state &= #40000000
```

is stating that the parameter “Digital\_state” is equal to the original value of “Digital\_state” ANDed with 4000000HEX. This has the effect of setting bit 30 to one with all the rest set to zero. In the next statement the value of “Digital\_state” is ORed with the result of the AND operation of the state of “To\_Manifold” and bit 1. If “To\_Manifold” is set its value is true and all its bits are set to one, otherwise it is false and all the bits are zero. This operation therefore has the effect of setting bit 1 of “Digital\_state” if the manifold valve is open, and setting it to zero if it is closed.

This “code” part of the program is used to provide the software trips of the instrument as well as to undertake such tasks as setting the voltages of the individual plates of the lens arrays, from a single setting value sent down by the controlling PC.

## The Language

The following is supported (note that everything is case sensitive, and all the nouns and verbs are **lower** case):

**long** Name = xyz    The noun *long* assigns Name to be a 32 bit variable which may be accessed by the PC program. The variable is assigned a initial value of xyz at start up.

**word** Name1 = xyz    The noun *long* assigns Name1 to be a 16-bit variable which may be accessed by the PC program. The variable is assigned a initial value of xyz at start up.

**false**            Sets all bits to zero.

**true**             Sets all bits to one.

**:=**                The assignment, similar to Pascal. Can be read as “becomes equal to”

**if**                The condition which follows is checked. If it is true the following statement(s) are executed. The keyword “then” is not required.

**else**             used with “if” to determine those statements to be executed if the condition is false.

<b>and</b>	The bits of two parameters are ANDed together – if both are one the result is one, otherwise it is zero.
<b>or</b>	The bits of two parameters are ORed together – if either is one the result is set to one.
<b>not</b>	Produces the result of inverting all bits of the parameter which follows
<b>+</b>	arithmetic plus operation
<b>-</b>	arithmetic minus operation
<b>*</b>	arithmetic multiply operation
<b>/</b>	arithmetic division operation
<b>&lt;</b>	Less than
<b>&gt;</b>	Greater than
<b>{</b>	Start of section of code
<b>}</b>	End of section of code
<b>//</b>	Comment follows
<b>#</b>	Indicates that the number following is in hexadecimal nomenclature
<b>Tabc--</b>	The parameter Tabc is decremented by one each time the line is actioned in the code
<b>Tabc++</b>	The parameter Tabc is incremented by one each time the line is actioned in the code
<b>A  = B</b>	equivalent, but much more efficient, to <b>A := A or B</b>
<b>A &amp;= B</b>	equivalent, but much more efficient, to <b>A := A and B</b>

Since this code section is actioned every 10 milli seconds, it is strongly recommended that care be taken to ensure that efficient coding practices are followed. Please contact us for further advice if you wish to alter or add to this section.

The source code for this software, together with other micro code utilities, should reside in a sub-directory call **Ms-code** under the main directory containing the PC control software. This will allow the compiled code to be downloaded to the system control by the PC operating suite, without the resort to a browse facility being required to discover the whereabouts of these routines.

### Micro Code Utilities

The following are the minimum requirements for working with the micro code:

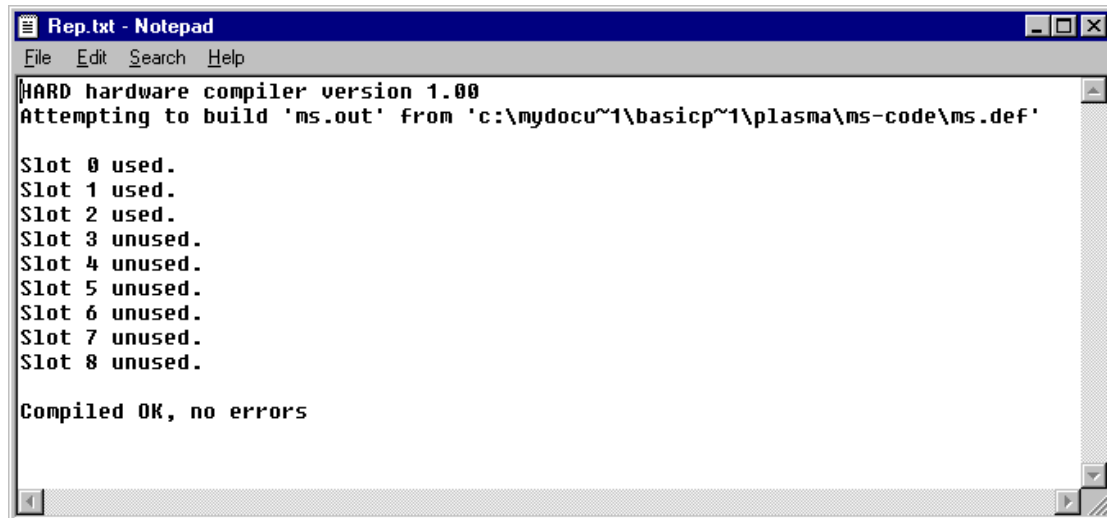
**Hard.exe** The compiler / optimizer to convert the source code into the compiled form necessary for downloading to the system micro.

**T.Bat** A batch file provided to automatically perform the compilation operation. Once the source has been saved (in Ms.def – see below), double clicking on T.Bat from within the Windows Explorer will perform the compilation.

**Rep.txt** This is a text file produced automatically by T.Bat to report the success or failure of the compilation process. The format of Rep.txt is shown below, and this can display may be produced merely by double clicking on the filename from within Windows Explorer, rather than using a text editor to view the file contents.

If an error is detected within the compilation process, this will also be reported inside Rep.txt. The line number of the (first) detected error will be shown, although like in all programming, one should always be aware that other errors may still be present

later in the code. These, if present, will be highlighted in subsequent compilation attempts.



```
Rep.txt - Notepad
File Edit Search Help
HARD hardware compiler version 1.00
Attempting to build 'ms.out' from 'c:\mydocu~1\basicp~1\plasma\ms-code\ms.def'

Slot 0 used.
Slot 1 used.
Slot 2 used.
Slot 3 unused.
Slot 4 unused.
Slot 5 unused.
Slot 6 unused.
Slot 7 unused.
Slot 8 unused.

Compiled OK, no errors
```

**Ms.def** Contains the source code.

**Ms.syn** A file produced on compilation of the source in which the addresses of all the parameters used in Ms.def within the system micro address space are shown. This file is used by the PC program to convert the high level names (*such as Protect\_Vacuum*) into micro software compatible addresses. It is accessed each time the PC code is started or the micro code is downloaded. Since the addresses may change as the source is modified (e.g. if the port assignments are changed), this lookup table enables such changes to be transparent to the user.

**Ms.out** The compiled code is stored here and it is this file which is downloaded to the system control.

**MS.cod** Also produced upon compilation – used in development.

**ABC.drv** A series of drivers which contain the code to control the actions of the various ports (see above). Also downloaded to the system micro.

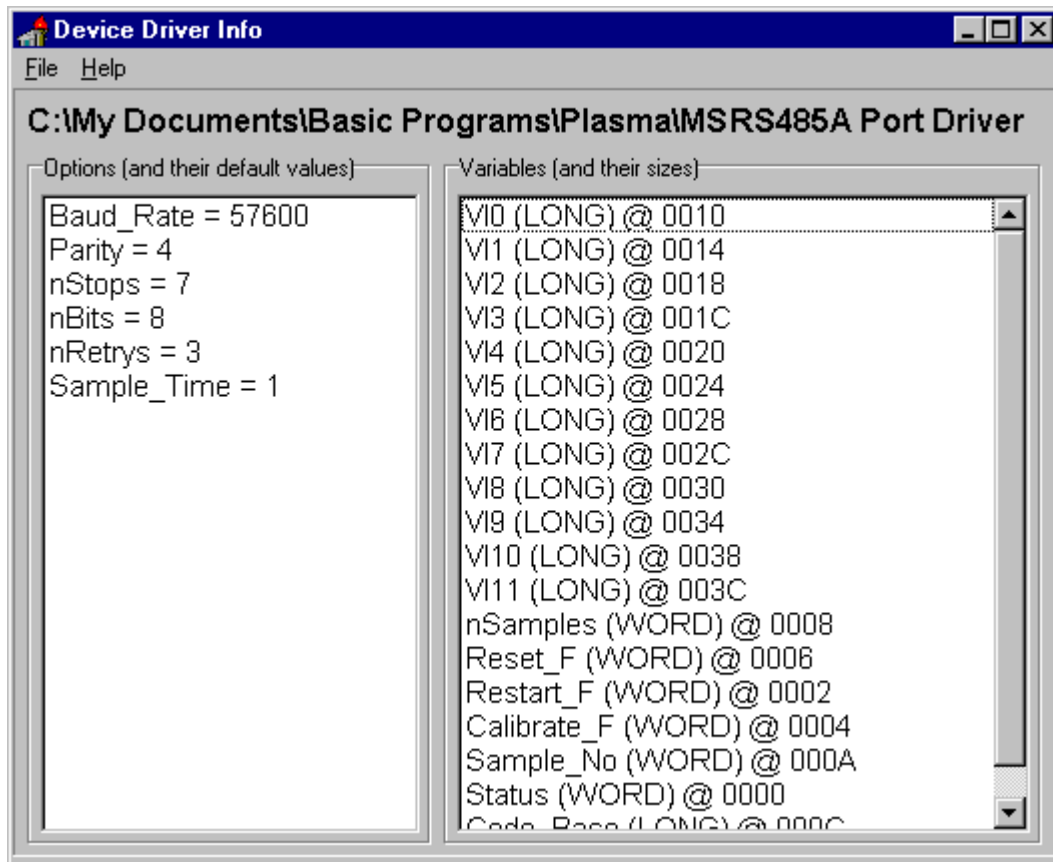
**Ddip1.exe** A utility provided to view the device driver parameters etc. Upon running this utility (double click on the file from within Windows Explorer), the required driver can be selected via the *File* menu. A sample screen is shown below for the 12 channel DVM driver and this corresponds to part of the assignment micro code which we used above. For convenience we repeat the relevant portion of the code here:

```
port SIO0 is DVMS
{
vars
  Status is U0_Status
  nSamples is Num_Samples
  Reset_F is Reset_DVMS
  Restart_F is Restart_DVMS
  Calibrate_F is Calibrate_DVMS
  Sample_No is DVM_Sample_No
```

```

    VI0 is DVM0
    VI1 is DVM1
}

```



The parameters on the left of the code section assignments refer to the driver variables (e.g. *Rstart\_F*) and they can be seen in the right hand window of the device driver utility. Also shown is the bit length for these parameters. Thus, in this example, *Rstart\_F* is a 16 bit (word) parameter whilst *VI0* etc are all 32 bit (long) parameters. Thus this utility will allow one to discover what variables are provided, their names (which must match exactly when used in the source code) and their bit length. The left-hand window shows the default values for the (given) control parameters used by the driver. These may be altered from within the source code section. Finally the title at the top of the window defines the driver under which the DVM driver must operate, i.e. the DVM driver sits on a RS485 port (software equivalent to RS422)

### To change assignments

Probably the only time in which the micro code will require changing is if a port or output assignment requires changing. (It is very unlikely that a slot reassignment will occur.) Before any changes are made to the Ms.def file, it is strongly recommended that a copy of the present version is made under a new name (e.g. Ms.old).

### To change ports

Suppose you wish to move the DVM unit from port 0 in the above example to port 5 (say). We presume that port 5 is at present unassigned. Simply change the 0 to 5 using

a text editor (e.g. Notepad in the Windows Accessories) and save the modified source code to file.

- Compile the code using the T.bat utility, and check in Rep.txt
- Exit the PC control software to ensure that the present running conditions are saved to the PC hard disk, since the present operating values stored in the micro will be overwritten when the Nu Noble program is exited. Restart the control program. This could be done at any time before the download is carried out, and if the run parameters have not been altered recently, may not even be necessary.
- Down load the software to the system micro using the relevant software menu provided (see software section below).
- Remove the serial ribbon from port 0 and plug into port 5. The instrument should now work as before.

### To change output assignments

Consider the following segment of code which defines the output assignments for one of the HV1000 units.

```
VO5 is Z_Plus  
VI5 is Read_Z_Plus  
VO6 is Z_Minus  
VI6 is Read_Z_Minus
```

Suppose (for some reason) output 5 has failed and you wish to use another (spare) port.

- First, simply change the address *VO5* to that of the new port *VO2* (say). Since in this example we also monitor the read back value (via the old *VI5* input, it is also necessary to change this address to the new one.
- Exit the PC software to ensure that the present running conditions are saved to the PC hard disk, since the present operating values stored in the micro will be overwritten when it is rebooted. Restart the control program.
- Down load the software to the system micro using the relevant software menu provided (see software section below).
- Remove the output lead from output 5 of the HV1000 unit and plug it into port 2. The old voltage, which was on *Z\_Plus* will still be present on the relevant plate. Any actions defined by the code section of micro program which referred to *Z\_Plus* will still function.

### System Micro Protection Routines

We have provided a number of intelligent software trips in the supplied Micro code, which we will now discuss. We will refer to actual variables used in the code to help navigate it, and you may find it useful to have a full listing of the source code available.

**Ion counting protection** This routine monitors the status of the “Over Beam” output provided by the ion counting unit. If it is set, it immediately changes the output on the z plates to deflect the beam away from the multipliers. The routine also alters the value of the long word “*Digital\_State*” (whose bit 30 is used by the PC

program to monitor for trips) and the long word “*Error\_State*”, so that the PC can decode what error has occurred.

The reader will also notice the use of the variable *Digital\_state*, whose bits are assigned depending on the state of various flags outputs. Thus, for example, bit 2 of *Digital\_state* defines the state of the source ion pump valve. This is done so that by reading only the one variable *Digital\_state*, the PC can know if any of (up to) 32 digital switches has changed (by merely seeing if the value of the *Digital\_state* has changed from its previous state). This is much more efficient in overall instrument time since the micro code overhead in recalculating the bit patterns every 10 milliseconds is much less than would be involved in transferring the 32 individual flag values to the PC as separate variables.

The use of the variable *Digital\_state* is slightly complicated by the requirement of the PC code to know the addresses of the various switches referenced. To speed up this control, the code assumes that the address of the n<sup>th</sup> item in the sequence is n words after the address of the “*To\_Manifold*” output. This is the reason that the outputs are referred to in the slightly odd, round about way. Firstly the addresses are defined by the list:

```
Digital1 := To_Manifold  
Digital2 := Source_Ion_Pump  
Digital3 := Remote_Control  
Digital4 := Bakeout  
Etc
```

This assures that the address are sequential, and in the correct order. The actual outputs are defined as:

```
DO0 is Digital1 // To manifold valve
```

in the assignment section at the top of the code.

Also in this part of the code section is a sequence of checks to ensure that the output of the deflector units (HV1000) and the high voltage bricks corresponds to the required set values. The status of these checks is only reported if the units are on, as will be seen by studying the code. Obviously a false error would be reported otherwise. Each output is checked to ensure that the measured output is within the preset variable “*limit*” of the set value. If it is not a bit is set in the flag “*Elect\_error*”, which can be read by the PC. The PC is informed of a problem by the state of bit 29 of “*Digital\_state*” which is set if the error is present for more than (approximately) 2 seconds continuously. These flags may be reset from the menu bar of the Nu Noble control software, as will be discussed fully below.

## Chapter 4: Software Overview

The Nu Noble control software is written using Microsoft Visual Basic 5. The program will run under both Windows 9x and Windows XP,NT and 2000 on PC fitted with a Pentium 350 or faster processor. Source code is supplied with the instrument and the user is at liberty to alter and / or customise the code as required. Although Nu Instruments can provide help to undertake such modifications, it cannot be held responsible for the performance of the instrument running under non-standard code.

As with all Visual Basic programs, the control software may be run in two modes, either from within the Visual Basic environment in an interpreted mode or as a stand-alone executable program. The former approach obviously requires the user to have a full working copy of Visual Basic 5 or 6 (Professional Edition) on their computer, and this is to be recommended if you feel that you may wish to experiment with the code. It also means that if a bug is experienced, the program will stop in a semi helpful mode and allow one to interrogate the variables to allow the cause of the problem to be found. When run as an executable, a bug will result in the program falling over in (what appears to be) a heap, with an incomprehensible error message.

Although the main control program is supplied and need not be altered by the user, the analysis routines will undoubtedly be changed and added to. These routines are completely separate from the main control software and are called only when required. A selection of analysis routines are supplied, and some will be discussed in detail below, and they will form the basis of user specific routines. These are written in a specially written compiler (to remove the requirement for the user to use Visual Basic compilers unless they so wish), which is interpreted by the main program and incorporated into it seamlessly.

There now follows a brief description of the software layout, which you may wish to skip on the first read through.

### Software Layout

For users with experience in programming only in DOS, or similar text based operating systems, Windows programming is a new experience! As well as having to worry about writing code to control the instrument and analyse data, one has to provide the Graphical User Interface (GUI), which is the part that the operator spends most time interacting with. Also, as discussed below, you can no longer be sure that processes occur exactly when you think it should, or even in the order in which you request them from the keyboard or mouse. Luckily most of the code for producing these Windows displays is now available from the Visual Basic environment and is mostly transparent to the author of the code. Many books (including the VB5 and 6 manuals!) describe how to write such code, and only the briefest overview is given here.

Windows programs are (obviously) based on a series of windows on the computer screen. Usually, but not always, unless the window is minimised, the user is not interested in the actions of those windows which have not been activated. Thus the code associated with a particular window is (normally) only loaded into memory when the window is activated, and is removed when the window is closed. Each

window can be thought of as an independent unit, although it will undoubtedly call general routines, which the operating system or the author provides. One window may only call on another if access has been permitted, and if the window is loaded. If it is not loaded, the operating system will automatically load it, in order for the code to be accessed, and this occasionally produces unexpected results when (in poorly written code), extra windows suddenly appear, apparently out of nowhere.

In a large program, such as the Nu Noble suite, there is a “top level” window and a series of subsidiary windows, which can be placed within it. The “top level” window is called the *parent* and subsidiary windows are termed *child* windows. The operating system ensures that the child is always within the parent area and it moves as the parent is moved. If the parent is closed, all the children also terminate. In the Nu Noble suite, the various child routines are activated using a menu driven structure. Each of these child windows has a *form* associated with it, on which the controls visible during program running were placed at the *design* stage of the program. Some of these controls accept user input, such as command buttons, text boxes and spin buttons, whilst others are just used to display information to the user. Every time one of the input controls is used, the operating system activates a section of code. Thus, for example if one was to click on a button labelled “Exit” the section of code to close the window should, hopefully, be processed.

The program also uses a series of timers. These produce an interrupt after a defined time period and causes a jump to specified section of code, without any user interaction. However (it appears) that you cannot always guarantee that a timer interrupt will be serviced and this can again produce some interesting results. Most of the time, however, the operating system is idle, (i.e. the program has serviced all outstanding actions), and when this mode is in operation, everything will run smoothly. If there is too much for the system to undertake within the time available, (and this can be important if a slow processor is used, or you play games on it), the system will just grind to a halt (crash). The most likely place for this to occur is if the program is written such that a routine (inadvertently) calls itself, and is therefore in an infinite loop. Another possibility is if too much traffic is being sent down the serial communication line to the system micro such that it takes longer to send and receive the reply than the period allocated.

Within the Nu Noble program are a large number of independent routines to perform the various control actions required. The more general routines are contained within basic *Modules* (e.g. Main\_Module.BAS and Decoder.BAS) whilst the more specific routines are contained within the code associated with their *Forms* (e.g. Serial\_form and Peak\_parallism). The layout may be seen by running Visual Basic and loading the Nu Noble code.

In the default mode of operation, little is occurring with the program. Every 0.1 seconds a timer is actuated which instigates a series of requests being sent by the PC to the system micro in order that the value of various parameters may be displayed. Depending on which windows are present on the desktop, various different types of data may be required, such as ion beam intensities, vacuum pressures and valve status (open or shut). It may also be necessary to send data from the PC to the system micro; for example if a valve is to be opened the required command must be transmitted down to the system micro for subsequent transfer to the relevant electronic unit



(utility unit in this example). Being a multi-tasking program, it is possible for all these windows to be working asynchronously. This could result in the request for data from one window interrupting an existing request, making it difficult to keep track of which reply corresponds to which question. For this reason the requests are all stacked up and sent out in an orderly queue in a routine called “General\_Read\_Loop”. For a similar reason one cannot guarantee that a command will be immediately sent out when the user clicks on the relevant button. For this reason the control may feel “spongier” than in non-multitasking suites and we have to take more care when collecting time critical data.

### **Directory structure**

The directory layout used by the program has a number of constraints applied to it by the program. The main directory containing all the code and other associated files can reside wherever required, but as written this directory **must** contain the sub directory *Ms\_code*, which contains the micro code for the system micro. Once the program has been run once, the path for the main directory is stored in the system registry, so as to enable the operating system to find the required data files on subsequent boot up.

This approach should not produce any problems unless the user wishes to experiment with different versions of the code, since running a modified copy from a different directory will still result in the data files from the original path being accessed. Thus if gross changes of the code are to be tested in which the format of the data files are changed, it is safest to rename the directory originally used, so that it cannot be found on starting the modified version of code for the first time. The user will then be prompted to provide the path for the running suite, which will then be stored in the registry, overwriting the previous settings.

### **Running “Offline”**

To assist with program development, the code has been designed to run on a stand-alone PC, when not connected to a system micro. This is achieved in two ways, by setting the relevant flag in the “*Setup*” + “*Preferences*” window of the program, or by merely starting the program whilst disconnected from the System Micro. On starting the program, if there is no reply from the system micro (and a timeout occurs) it will set the “*Offline*” flag itself. If communication is disrupted at any other time of the program execution, a standard timeout (with a beep and timeout error reported in the monitor window) occurs, but the PC tries to remain on-line. The fact that the system is running “*Offline*” is reported on the Title Bar of the Nu Noble main window. Very occasionally communication with the system micro is corrupted on starting the program and so the system will boot up into this off-line mode. In such cases it is best just to exit and restart the program.

### **Constant and Data files**

To ensure the most flexible mode of operation, all constants used by the program are stored in comma separated ASCII format in separate files within the main directory. These files may be edited using a standard text editor (e.g. Notepad from the Accessories section of the Windows Desktop). They are read by the program on boot up, and the user may force a re-initialisation using the “*Setup*” + “*Constants*” menu.

These files are labelled in the format **ABC.dat** and some are described below. Not all the more obviously named are fully described:

**Collector Data.dat** Contains the information which allows the program to define the collector array present in your instrument. Is split into two main sections, for the faraday(s) and ion counters. Has information describing the actual position of the detectors (in millimetres from the axial position) and the addresses of the reading channel.

**Deadtime.dat** Contains the ion-counting multiplier dead times, in nanoseconds. The first number is for channel zero etc.

**Deflect.dat** Contains the text description of the controls shown when the *supplies control* window is activated by the program (under the “Control” + “Supplies” menu tree). If these description require changing, they can be altered here, but please note that these descriptions must **exactly** match those in the MS.Def files (apart from a space being replaced by the underscore character), and so the change must also be done in this file. (See Chapter 3 on the micro code for instructions on how to alter this.) There then follows two dummy entries. The first may either be the word “DUMMY” or a random number. The “DUMMY” label is used by the program to indicate that the actual output value does not directly control a voltage, but rather defines a series of outputs. An example of this is the “Quad” settings, whose outputs define a SERIES of voltages applied to a set of plates of the lens array. As such a directly monitored output is not available. The main program uses the second entry to decide on how the plates are wired. If the second address is a minus sign, ‘-’, nothing further is done when the deflector value is altered. If a dummy value is present, this is the key to tell the program that this control operates a **pair of plates**. Within the Ms.def file there should be an name such as *Plate@pair*. This is the paired plate and the program will be able to find its address from the MS.sym file. Every time the value of the voltage on “Plate” is change by the user or program, the opposite signed voltage is automatically sent to this paired output.

The next number is defines the tab (one of three) on which the control appears in the supplies control window.

The final four values are the step size to be employed if the “Up – down” buttons are used, the maximum and minimum possible values to be output, and a conversion factor to convert the required output to the actual binary number sent to the system micro.

**Gain.dat** Contains the relative gains of the Faraday and ion counting detectors. The data is relative to that for the lowest numbered Faraday. The Faraday data values occur first, in ascending order, followed by the ion-counting channel values.

**Mag\_Cal.dat** Contains the data for calibrating the Hall probe. This is automatically saved after a calibration is undertaken. The first line contains the order of the fit used (note order 1 has 2 constants since the formula used in this case is  $y = a + bx$ ), followed by the number of data points used in the fit.

There then follows the constants calculated by the fit (these being used by the program to convert a mass value in amu to a field output value in DAC steps). The final data in this file are the raw data values used to obtain the fit, so that they may be edited at a later time from within the Nu Noble program, if required.

**Monitor.dat** This is a fairly complicated looking file, designed to enable the customisation of the monitoring capabilities of the program, being associated with many of the windows under the “Monitor” menu bar. Each entry defines which group the following monitor point is from (e.g. magnet, vacuum etc), followed by its text description. This is followed for the non-digital entries (digital controls can only have an ON or OFF value) by the units which the entry is measured in (e.g. volts, mbar) which will be displayed by the Nu Noble program. There then follows the constants necessary to convert the observed voltage reading from the analogue to digital converters to the required measurement. Two forms of conversion are catered for in the program, the simple linear fit:

$$Y = a + bx + cx^2 + d x^3 + \text{etc}$$

or a log fit, used mainly in pressure measurement:

$$\text{Log}(Y) = a + bx + cx^2 + d x^3 + \text{etc.}$$

The number of constants (i.e. one more than the fit order) and the values for these constants then follow.

The user should also be aware that although contained in this monitor file, those “Digital” controls can be both digital control outputs as well as inputs (e.g. “*Manifold Valve*” is an input whilst “*Remote Control*” is an output).

**Quad\_setting.dat** Contains the values for different axial masses for the Quad 1 and Quad 2 voltages, to ensure that coincidence is achieved. Data can be added automatically from the “Setup” + “Quad Values” menu window, and is used to estimate settings if the values are not known and by the “Optimise” routine in the magnet scan window, which can automatically set the zoom constants.

The third figure after each data set, is 1 if “normal” zoom conditions apply (i.e. the data can be used in a least squares fit to estimate new data points), and zero otherwise.

**Setting.dat** Contains the values of the control outputs the last time that the program was (properly) closed, i.e. from the “File” + “Exit” menu or the “Close” button on the parent window. To aid deciphering of this data, the text description is also included. These values are used whenever the system micro is rebooted, and may also be employed (but only if required i.e. the user is asked if this is required before this is done) from the “Setup” + “Constants” menu. If these values have not been updated recently, and a reboot is required (e.g. because a reassignment of ports is necessary), it is recommended that the user exits the program and restarts it, before the download, so as to update these data.

**Switch.dat** This file contains the information of the switch outputs used to control accessories to the instrument. The data in this file is used by the

program to construct the “Switches” menu by which these devices can be controlled. The format of this file is as follows:

1. A code flag. :
  - 0 = A “normal switch is being controlled – so just change the state of the digital switch
  - 1 = the output is talking to an “intelligent” device for example via a serial link.

If the code is zero the following describes the strings:

2. The name given to the output being controlled, as used in the Ms.def file. This data is checked each time the menu is called to ensure that the name is valid. This means that the file may be changed without rebooting the program, so that if a number of accessories are present, different versions of the file may be used depending on the required configuration, without a large number of switch names appearing on the screen.
3. The description to appear on the control bar when the switch is disabled (set off). Thus if the switch controls the firing of a laser, when the laser is off, the description could read “Start Laser”.
4. The description to appear on the control bar when the switch is enabled (set on). Thus if the switch controls the firing of a laser, when the laser is on (i.e. lasing), the description could read “Stop Laser”.

Otherwise the following applies:

2. The first part of the string must correspond to one of the “supported” commands, described later in the section describing the automatic sequencing. There then follows the number of required input parameters, an address followed by the list of these parameters.

The file is ended with “EOF”, and the program will ignore any data after this.

**Startup Parameters.dat** Contains all the values of parameters necessary to define the running state of the instrument. The values are written when the program shuts down, so enabling seamless operation of the instrument when the program is restarted. A description of most of the parameters is written to the file, to aid readability. If you change the instrument configuration, it will probably be necessary to change the corresponding entry in the “Startup Parameters.dat” file. To do this, exit the NU Noble program and then change the entry. If you do it whilst the program is running, your change will be overwritten when the program ends!

**Supported Commands.dat** Contains a list of all the commands which are interpreted by the automatic sequences. Used by the utility which allows the user to automatically write a sequence of commands to be followed (the automatic scripting program)

**Tau.dat** Contains the values for the measured constants from the Faraday Tau correction routines. The routine to fit the data is:

$$Y = A(1) + A(2)\exp(-x/A(3)) + A(4)\exp(-x/A(5)) .$$

Since the first term is merely the zero offset, although it is calculated in the non-linear least squares fit in the Nu Noble program, it is not needed to be stored in order that the software Tau correction be undertaken.

A series of files under the generic name "T\_beamX" are also present. These contain the observed raw data from the last Tau run, to enable off line testing to be undertaken.

**XXX.nsf** Contains the output from the "Designer" software package supplied with the software suite, and is the compacted information which is used by the program to produce the "vacuum" and "prep system" schematics. You will not be able to read this file!!!

### Analysis Files

A number of analysis related files are also present, which contain data in a comma separated ASCII format, and permit the Nu Noble program to undertake the required, user defined, analysis operations.

**Isotopic analysis files:** Has the "MDF" (Mass Data File) extension, and contain the information necessary to undertake the required analysis of isotopic intensities. Is created from input from the mass table window.

Each parameter is prefaced with a string, which identifies the meaning of the term.

Each analysis file should have associated with it an executable .ncc (Nu Compiled Code) file with the same basic name (i.e. *ABC.mdf* requires a file *ABC.ncc*) which contains the analysis maths for the run. It must reside in the same directory as the MDF file, and this is checked before the Nu Noble program starts each analysis. Since the exe file is produced using the NICE (Nu Instruments Calculation Editor), there will undoubtedly be source code file for the analysis, with a .crf extension (Calculation Routine File) in the ABC group.

**Batch and sequence files:** Contains the information for a series of sequences to be run automatically (held in a file with a .txt extension), or the single sequences themselves (with the .seq extension).

The sequences are a list of commands, which is conveniently created in the correct format using the supplied utility within the Nu Noble environment. These define a sequence of actions such as open or closing valves, talking to "intelligent" peripherals via serial interfaces, and running analysis programs. Branching, depending on the state of an analogue input is also supported. There is a whole chapter devoted to this facility, and we will not discuss it further here.

### Communication with the System Micro

Communication between the PC and the system micro is undertaken via the serial port of the PC and the RS232 port of the microprocessor card of the micro. The system

micro connector has been assigned to enable pin to pin wiring for the cable to be employed, but unfortunately, due to problems at the PC end, it has been found necessary to hard wire a number of the control lines to ground, to stop flags being set unintentionally. The wiring used is given in Chapter 3, Electronic Unit Description.

The setting of the port is:

Baud Rate:	57600
Parity:	None
Data bits:	8
Stop bits:	2.

### Software protocol

The communication between the PC and the system micro can be in two forms. The most vigorous has a software Acknowledge (Ack) (character 1B hex) after each data string is received correctly, or its inverse (a Nak) (character 1C hex) if data is corrupted. The characters used in this mode have to be removed from any string being transmitted. This in turn means extra calculations are required for each string sent and received. However it has been found not to be necessary to use this protocol, and other checks are used in order for the simpler method to be employed. If you require more information on this protocol, please contact us.

The strings sent between the two devices are split into up to five parts:

1. *A start of string byte.* The character A6 hex (116 decimal) is used. Every string must start with this character.
2. *A Command byte.* The following assignments are used:
  - ◆ 1: require to read a 16 bit data value from the address which follows
  - ◆ 2: a 16 bit data value is being transmitted
  - ◆ 3: require to read a 32 bit data value from the address which follows
  - ◆ 4: a 32 bit data value is being transmitted
  - ◆ 7 to 10: used to indicate that various micro code sections are being transmitted to the system micro – please contact us for more information.
3. *The Micro Address.* The address in the system micro where the data is to be sent, or read from. These addresses are contained in the MS.sym file discussed previously, and this is used to convert between readable “high level” names used in the PC code to the 4-byte addresses used for this data transmission.
4. *The Data.* If data is being sent, this next portion of the message contains that data. Thus if a 16 bit value is being sent, the next two bytes are filled with its value, if a 32 bit value is required then four bytes are necessary.
5. *A checksum byte.* This is calculated by summing all the previous bytes and ignoring all but the lowest byte value. The value of the checksum employed is

the byte which must be added to this calculated sum so as to produce 256. The code necessary to do this sum is:

$$\begin{aligned} Sum &= 256 - (Sum \text{ Mod } 256) \\ \text{If } Sum &= 256 \text{ Then } Sum = 0. \end{aligned}$$

Within the program as supplied, the software undertakes the following checks before the data is accepted:

1. The length of the replied message is correct. Nine bytes should be received for every 16 bit value requested and eleven bytes for a 32 bit value
2. The address in the request string exactly matches the address in the reply string
3. The check sum for the reply string is correct

If any of these checks fail, the request is reissued, with up to four attempts being permitted before a communication error is reported.

## Chapter 5: Basic Software Operation

### Housekeeping operations – The “*Protection*” menu



At the far end of the menu bar there are three controls under the “*Protection*” menu, as shown here. As mentioned previously, every 10 milliseconds the system micro checks to ensure that there are no parameters outside their set operation range. If an error is detected, it checks for another N cycles (anything from 6 up to 500 – five seconds worth of data) to ensure that it is a real error and not just a glitch. If it is valid a defined trip routine is performed (set by the micro software code) to protect the instrument and an error flag is sent to the PC. This error is reported in the monitor box error section (see below) as an integer number, which can be decoded as a bit message to discover the error. This is now done by the program itself, and the result will be shown on the top (now flashing) title bar.

The system micro is also checking that the measured voltages are equal to their set values (or more precisely not far different from them) for most of the electronic units in the instrument. If a difference is detected, an error is raised and the Nu Noble program notified. As there are so many possibilities here, we will not list them, but the interested reader can look at the code in the MS.DEF file to see how these trips are set. To aid the decoding of the error from the recorded bit pattern, the program will decode the message and flash the identifying message on the title bar of the program. Please note that we have specifically designed this error detecting to be “sticky”; in other words if the problem cures itself and the fault goes away, the warning message does not. This is because the detection process has been found to be so reliable that, once activated, there is normally something wrong, and it is recommended that the cause be found. Even if it is electrical interference from outside the instrument, it can still affect the performance when analysing very weak signals.

Once an error has been reported and noted, the flag may be reset by clicking on the “*Reset Trips*” control under the “*Protection*” menu. This should remove the trip message from the Monitor box and the flashing message on the title bar. If the message refuses to disappear, the fault still remains and the user must investigate further.

### The “Protect Instrument Settings” option

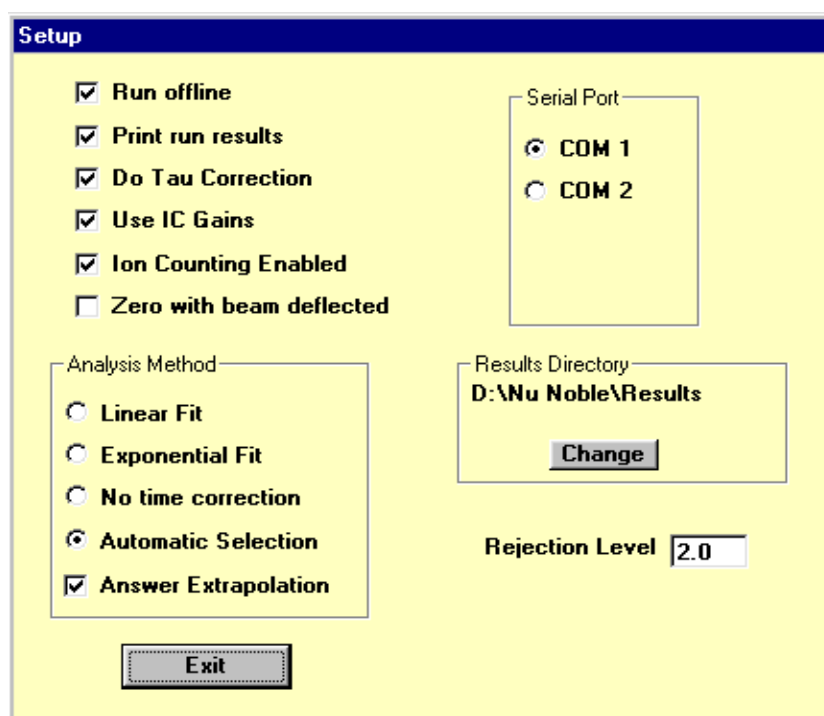
There are a number of instrument voltages which are rarely altered, and on which the instrumental response is quite sensitive. Examples of such parameters are the ion multiplier voltages and discriminator settings, the source trap current and the maximum power, which can be fed into the filament by the supply. Checking “Protect Instrument Settings” flag in the menu list will ensure that these (and a few others) controls may not be altered (they are greyed out), although their present set values are shown. If you do wish to alter any of these protected controls, it will be necessary to un-check the option, when all the controls will become active.



## The “Setup” menu

### Preferences

On clicking on the “Preferences” option the following screen is displayed. A series of Check boxes are present, which define how the program will run. It is not expected that these will be altered very often.



*Run offline:* Tells the program that communication with the system micro is not required, allowing the PC to run in a demonstration or debugging mode, disconnected from the instrument. If communication cannot be established with the instrument immediately on starting the program, this mode of operation will be used by default. If this is not required, try rebooting the PC program. When this mode is in use, the title bar of the Nu Noble program provides a visible feedback, since random numbers are being used for all data input, which can produce some interesting effects if unnoticed!

*Print Run Results:* Enables the printer to record the results of analysis routines. If not set, the results are still calculated and data stored to file, but no hardcopy output is provided during batch runs. In practice will probably not be altered.

*Do Tau Correction:* Enables the software Tau correction to be used in the analysis routines. In practice this should not be disabled since this will result in (small) errors occurring in the recorded data. The facility to disable this is provided merely to enable debugging to be undertaken if a problem is suspected with the Tau routines.

*Use IC Gains:* Enables the measured ion multiplier collector gains to be used in each measurement, if they have been measured and input into the “Gain.dat” file, and not left at the default value of unity! This option is useful if you wish to measure the gains, when disabling this option means that you don’t have to deconvolute the old gain value from the recorded beam intensity.

*Ion counting Enabled:* Enables the ion counting beams to be recorded by the operating system.

*Zero with beam deflected:* During the analysis routine, if this is ticked, during the zero cycle measurements, the beams will be deflected vertically at the source, so as not to pass down the flight tube.

Within the Serial Port *Frame* in the window are a series of *Option (or Radio) Buttons*. Only one of these buttons may be set at one time, if the setting is changed by enabling a new option, the present set one will be disabled. The settings within this *frame* define which communications port is to be used for talking to the system micro. In practice the setting need not be altered unless a new PC is to be used. In such a case, if the port is different from that shown, the program will boot up in the “*Off Line*” mode, and the assignment can then be achieved (either by reference to the PC manual or by trail and error) before the online communication is activated.

The second *Frame* identifies the directory where the raw data obtained during analysis runs together with the instrument run parameters at that time are stored. The directory can be altered by selecting the “*Change*” button, whence the directory tree will be displayed. After selecting the new directory, use of the “*Accept*” button will ensure that this path is used in future, whilst no change occurs if the “*Quit*” button is used.

The third frame identifies the analysis method to be employed during the data reduction program, and is discussed in depth in the relevant chapter.

The rejection level text box defines the value to be used by the analysis program when it rejects noisy data points. With the illustrated value of 2, approximately 5% of the data will be excluded from the data fits. If you wish to include more of the data, this value should be increased using standard statistical considerations.

The *Exit* button will close this window, when the new settings will be stored within memory. Storage to file only occurs when the Nu Noble program is terminated.

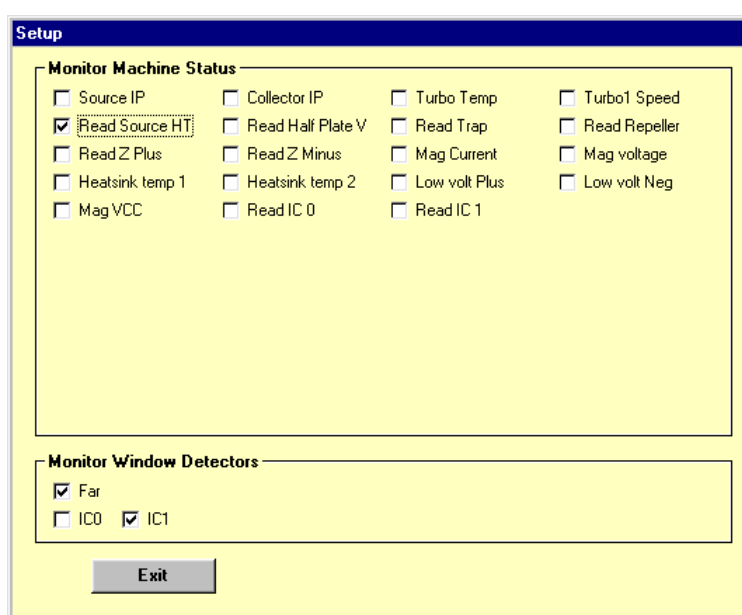
### **Monitor**

The monitor control produces the screen shown below for defining which instrument parameters are displayed in the monitor window and updated each measurement cycle. Up to three ion beams may be selected and up to four general monitor points can also be serviced here. All the monitor parameters can be accessed and displayed using other windows, so this approach does not limit the system monitoring, but is merely provided to enable the points of most interest to be easily accessible.

Detector selection: Up to six ion beams may be selected for display in the monitor window. In practice the total number of beams and “machine monitor

items” (see below) is six, and if more than three beams are required, it may be necessary to deselect some “machine monitor items. If the full complement have been requested for display, the program will not permit a further beam to be selected. To choose a new beam, it will be necessary to deselect one of those already enabled first.

Machine Monitor selection: Up to three items can be enabled from the list, and as previously hinted, if three have been selected it will be necessary to deselect one before a new selection is permitted. The text shown is picked up from the *Monitor.dat* file, and may be altered if required. The selection shown enabled in this example refers to the output from source HT brick. This could be used to monitor voltage breakdown in the source, for example. The output from the last displayed machine monitor can also be plotted in real time in the magnet scans window (see below).



The *Exit* button will close this window, when the new setting will be stored within memory. The assignment of the displayed ion beams are stored on closing down the control program, but the monitor selection information will be lost, since this is regarded as non-vital information.

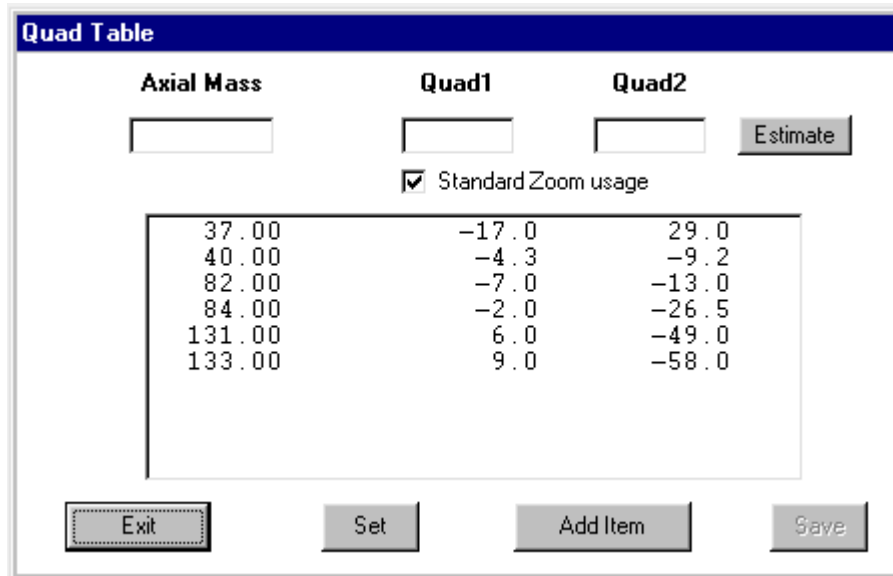
## Quad Values

This window enables the operating values for the two quadrupole lenses to be stored for future reference. The form of the window is shown below.

The window consists of two main areas, the top being the data entry area, whilst below it is displayed the data which has already been input (and hence stored to file) previously. The data consists of four parameters, the axial (or reference) mass at which the observed quad values were obtained, the observed values of the quad(1) and quad(2) lens voltage, and a flag to say whether this data correspond to the “natural” zoom setting of the instrument. To see what this means, consider the case of xenon, where, if we have an instrument with collectors set to record the Ar36,38 and

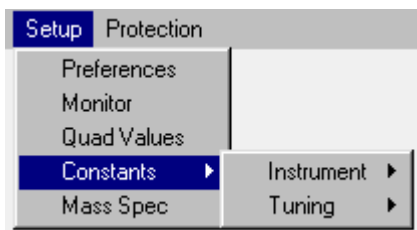
40 peaks, the “natural” zoom settings would have 6 amu between adjacent collectors. However it is useful in the case of xenon to have (say) 5amu between collectors for some measurements. This data should then be entered with the “Standard Zoom Usage” unchecked. This will ensure that this particular set of data is NOT used in the least squares fits of the quad voltages, within the program.

The utility allows one to estimate the zoom settings for masses not previously studied, by merely typing in the required axial mass, and depressing the “Estimate” button. The quad values may be downloaded to the mass spectrometer by use of the “Set” button.



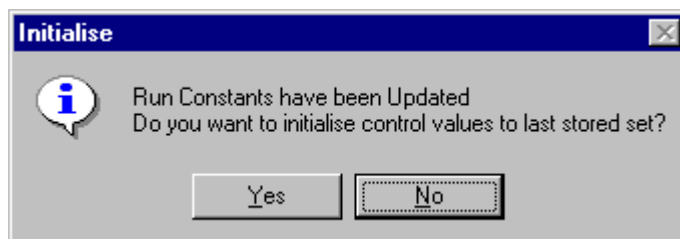
If one wishes to edit an entry, or to use the utility to download a previously determined value to the mass spectrometer, it is merely necessary to double click the relevant entry in the table, which will then load the selected values into the text boxes.

### Setup Constants



If activated by clicking on with the mouse, two options are displayed, as shown here. If the “Instrument” option is selected, one can reset the instrument to the complete set of running parameters, which was stored when the program was last shut down. These include the collector gain values, the magnet calibration constants and the Faraday Tau readings. This update is done automatically, but after this has

occurred, the user is given the opportunity to download the last saved HT and deflector settings to the system micro. The message box shown here is displayed.



occurred, the user is given the opportunity to download the last saved HT and deflector settings to the system micro. The message box shown here is displayed.

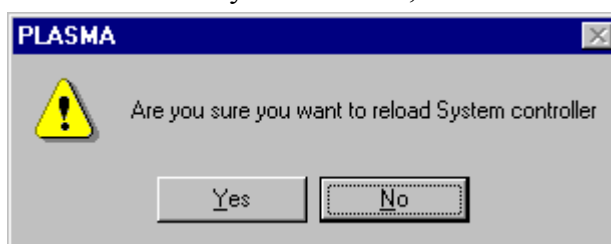
Whenever a message box is displayed, it must be acknowledged before normal control of the program can resume. Thus one of the two options (Yes or No) must be chosen before you can continue. Since the option of not downloading the parameters is the safest in this case, the program has been written to make that the default option, as shown by the command button being the more prominent of the two. As such hitting the return button of the keyboard will select this option, or the mouse (left) button can be used as normal. Selecting “Yes” will download these saved parameters.

It is also possible to save the present running parameters, via the “Save” option, and the print out a hard copy, via the “Print” option.

It is also possible to save and update the run parameters (not the gains etc here) for individual tuning conditions to and from named files. This option is useful, for example, if one is studying more than one element, where the different element source tunings can be stored in different files. The standard file window is opened on either of these options, allowing the user to select the required filename, and if the “Initialise” option is chosen, the new parameters read from the selected file are automatically sent to the instrument without the above message box appearing.

### Setup Mass Spec

This procedure will download the micro code to the system control, and as such is used if the mass spectrometer power has been disrupted, when the micro’s memory will have been wiped, or if modification to the code has occurred, for example if a reassignment of ports was necessary. Due to the drastic effects this can have to the unexpecting button pusher, a confirmation message box has to be answered before the routine is started. This is shown here.



Again, the message box must be responded to before any further inputs to the main program can occur. The default input is again defined to produce a “safe”, no-action, response by the program.

If activated the Nu Noble program will firstly down-load all the required drivers to the system control. The monitor box will display (in a slightly cryptic manner) the progress during this operation. If the download is successful, a second message box is displayed, reporting this fact, which again must be acknowledged. Since the default state of all variables has been predefined to be in whatever the save mode is, all voltages will at this time be zero and all valves will close etc. The values for these parameters saved to file, at the last occasion that the Nu Noble program was closed, are now downloaded to the micro. This results in a sequence of audible valve changes occurring during this download process, but this is quite normal and should be perfectly safe. It is advisable not to try to carry out any operations during this download and reassignment of operating parameters, since it cannot be guaranteed that they will be correctly decoded until the micro re-initialisation process is completed.

## File Menu

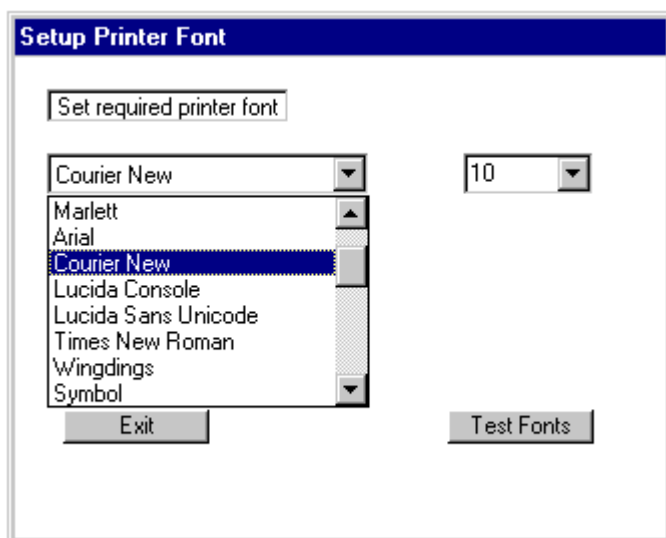
This menu provides a number of utilities of general interest for control of the printer.

## Print Setup

The first allows the printer format to be changed, which can be useful in the analysis routines for reporting the observed results. On clicking on this option the standard printer window appears. This window is the standard one provided by the Windows operating system for altering the printer properties. The form of this window can vary depending on the printer attached, although its use should be obvious.

## Printer Font

This routine allows one to alter the default printer font and size used in recording the analysis run results. The fonts available will vary depending on the printer and computer set-up, but it is recommended that a non proportional (i.e. fixed separation) font is used where each character uses the same width of printed output. This will ensure that columns of data line up with themselves rather than producing a jagged output. The window is shown below:



The names of the available fonts are held in the drop-down list, which may be accessed by clicking on the down arrow at the right hand end of the text window. This is shown activated in this example and as such all the available fonts are shown in the list form and one may be selected by double clicking on the required name. Using the “Test Fonts” button will print out example text for each of the available loaded sets. As mentioned above , a

non-proportional font such as “Courier New” is recommended.

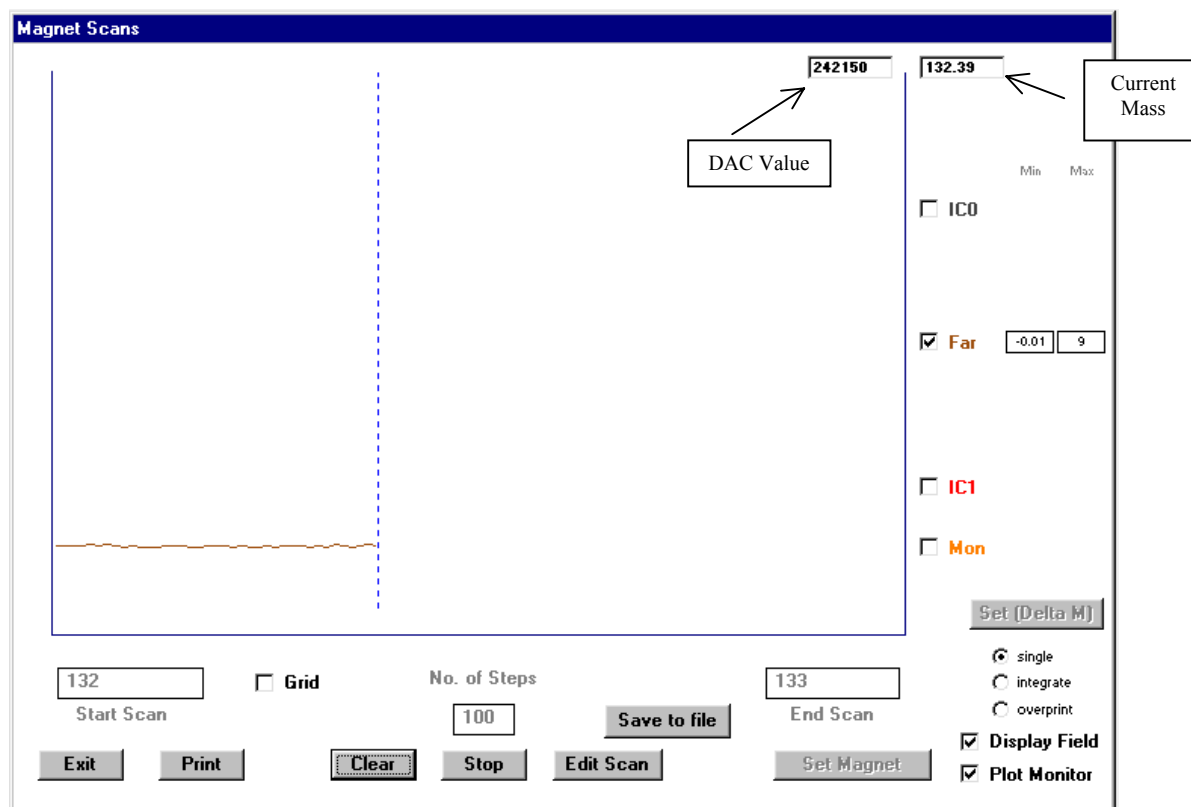
The font size may also be selected from the available options using the smaller drop-down text box. If a size greater than 10 is chosen, it may be necessary to use a landscape paper orientation in order to ensure all the analysis data is correctly written on each line.

The final option under the “File” menu family is “Exit”. Clicking here will close the Nu Noble program, storing to file the latest used values of the system operating parameters, as it does so. If exiting is done using the “Close” button at the right hand extreme of the title bar, the user is prompted to confirm that this is required. This

approach is adopted to ensure accidental closure does not occur if this button is clicked instead of its neighbour.

## Magnet Scans

This routine is one of the major ones used and it will be well worth being familiar with the various controls provided.



The window is shown above. On the right hand side are the check boxes to define which ion collectors are displayed. In this examples only the faraday is being monitored. The traces are colour coded to aid identification. If the “*Ion Counting*” option has been enabled from the “*Setup*” menu then the extra check boxes, corresponding to the extra possible multiplier channels, are shown corresponding to their actual physical placements (as specified in the “*Collector.dat*” file, see above). Any number of detectors can be enabled, although obviously the screen can get a bit crowded if too many are shown at once.

The scan range used is shown in the two text boxes at each lower extreme of the plot window. This range is divided into a number of discrete steps (in this example 100), and the scan process employed is to increment the magnet by a single step, wait for 0.1 second and then read the required beams, after which the magnet is incremented again. The values used in these three text boxes by be altered by clicking the “*Edit Scan*” command button, when the boxes will no longer be greyed out (disabled), and new values will be accepted. Clicking the “*Save Scan*” command button (the same button is now re-named) will permit these new values to be accepted, clear the screen and start a new trace. The program will protect against stupid values being accidentally input (e.g. a mass greater than 500) and if an error is detected the user

must correct the mistake before the text boxes revert to their standard “greyed out” mode. Scan ranges where the start and end values are the same (to allow the user to sit on the top of a peak for optimisation purposes) or the end is less than the start value (reverse scan) are acceptable.

Three modes of display are provided, as defined by the option buttons in the bottom right hand corner of the window:

- *Single trace:* the scan is overwritten each cycle
- *Integrate trace:* the scan displayed corresponds to the integrated average of the beam from the time the scan region was last cleared. The number of the present cycle will be displayed if this option is chosen.
- *Overprint trace:* the scan is overprinted each cycle. This mode of operation permits one to compare peak shapes as (for example) the zoom lens settings are altered and so find the optimised settings.

The “*Stop*” button (obviously) stops the trace at the present position. The caption then changes to “*Start*”. A number of buttons are disabled during the stop stage, to prevent clashes due to timer problems. When the scan is stopped the “Set Magnet” button is activated. Depressing it will cause the upper and lower values of the scan range to be set to the present values of the cursor, and the scan restarts, although obviously the magnet value does not change from the selected value. This facility is often used for beam tuning, where one wishes to sit on a peak top. If selected, the button caption is changed to “Restore Range”, and a second press will restore the previous upper and lower values.

When the scan is stopped, it is also possible to move the cursor (and obviously the magnet) by dragging it with the right mouse button depressed. Further double clicking the left mouse button at any two points of the scan will reset the upper and lower masses of the scan range to the values selected. This facility is often used to zoom in on a narrow peak to study its shape in finer detail.

The “*Clear*” will clear the display and restart the trace from the “Start Scan” value. If the integrate mode of display is active, the sequence reinitialises itself and starts afresh, i.e. all previous information will be lost.

The “*Print*” button will result in a bit image of the window being sent to the active printer. The code automatically ensures that the maximum print size is obtained, and prints the date and time of the dump at the paper top. It is not necessary to change the printer page orientation (using the “*Printer Setup*” menu) to ensure this page alignment.

The “*Exit*” button will close this application, although there is a slight (1 sec) delay before this occurs to overcome timer clashes. The button is disabled once the command is accepted to give the user feedback that things are about to happen.

The current value of the magnet setting (in atomic mass units) is displayed at the top right of the scan region. This is the normal display. However to calibrate the magnet DAC (Digital to Analogue converter), which is required so that when you say “jump to mass M”, the micro knows the DAC setting corresponding to mass M, one needs to know the bit setting being sent to the system micro. This may be displayed if the



“*Display Field*” check box is enabled, as in the example show above. In this case a second text window is displayed to the left of the mass value window, and this will report the current DAC bit setting.

The present values for the maximum and minimum of the plotted beams are shown to the left of the collector selection tick box. The values may be altered by double clicking on the required box, when a new text box will appear for the new required value to be entered. The value is accepted via the “ENTER” key, but if an error is detected (e.g. the new maximum value is less than the present minimum) the text box will not be removed, and a valid data entry will be required.

Clicking the “Grid” check box will cause faint horizontal grid lines to be drawn on the plotting area, which is useful for monitoring peak quality.

In the example shown, we have checked the “Plot Monitor” check box in the lower right hand side of the window. This has caused the extra check box labelled “MON” to appear below the ion collector boxes. If this check box is enabled, the program will plot the value of the last displayed instrument parameter displayed in the monitor window. This is useful if one suspects an intermittent fault with one of the instrument outputs, where glitches can now be plotted in real time.

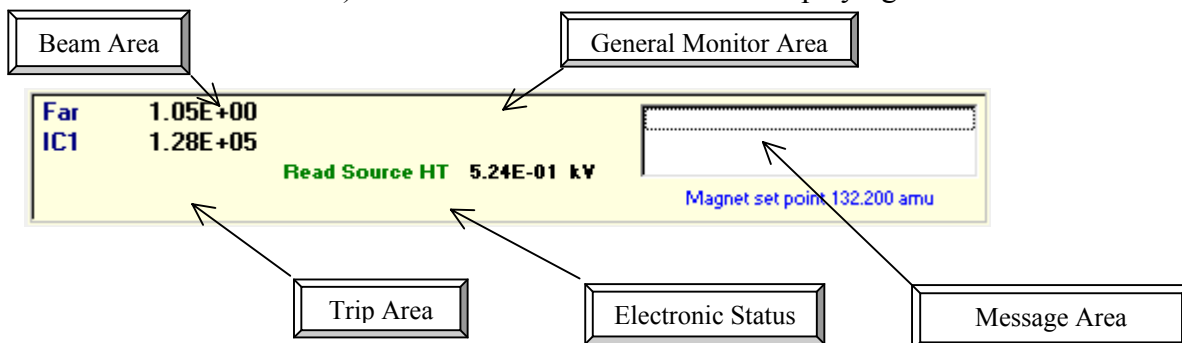
If only one detector is selected (as in the case illustrated) the save to file button is enabled. This will allow the last completed scan to be output to a selected file (in CSV format) for finer study offline.

Finally there are two options selectable when the scan is stopped, depending on the number of detectors enabled. If only one is checked, the button is labelled “Set (Delta M)” as in the example shown. This is used if one is analysing in a high-resolution mode, and requires the analysis program to sit on a plateau of a double peak. This option is used to tell the program how far from the half height position of the peak to sit during the analysis, since (obviously) the normal peak centre position has no meaning with such a profile. To use this facility, obtain a (good) peak shape of the required composite profile, and then stop the scan. Move the cursor (using the right mouse button) to the position at which you wish the analysis program to perform its measurement, and depress the (now activated) “Set (Delta M)” button. The program will calculate the offset from the selected point to the half height position and will report the value in a message box. If you feel that the value is correct, accept it, otherwise reject it and have another go. During analysis the “Peak Centre” program will determine the position of the half height and set the magnet at the stored offset value from this observed mass.

If more than one collector is selected, the button caption shows “Optimise”. With the scan stopped, this feature allows one to automatically find the quad values which will result in two selected beams being coincident. To operate, select the two beams required (if more than two were initially enabled), depress the start button and follow the instructions in the message boxes displayed. Since the program uses the stored quad values to produce a least squares fit of these data in calculating the optimum values, this facility will only produce a sensible fit if the quad setting are reasonably accurate.

## The Monitor Window

This will normally come up automatically in the bottom right hand part of the main control window, and stays in position as this window is scaled (as opposed to the other windows, which keep their absolute position and may be clipped if the main window is reduced in size). It contains four main areas for displaying information.



The values of (up to) three ion beams may be displayed on the right hand side of this window. The identification of the beams displayed may be set from the “*Setup*” + “*Monitor*” window as explained above. The display units is volts for the faraday detectors, which may be converted to the ion beam current by multiplication by the corresponding preamplifier board resistor value (usually  $1 \times 10^{11}$  ohms) and cps for the multipliers.

The General monitor area can display the values of up to four monitor points, again selected from the “*Setup*” + “*Monitor*” window as explained above. Since these values are updated serially, if four are selected for display, each value is only updated once every four cycles. For this reason we normally recommend that the minimum required are shown, as in the example here where the source HT output is selected.

The message area contains a series of information messages to provide a history of the machine operation. Only the previous 100 items are displayed, and if more items are present than can be displayed, a scroll bar will appear on the left of this area to permit the previous entries to be observed.

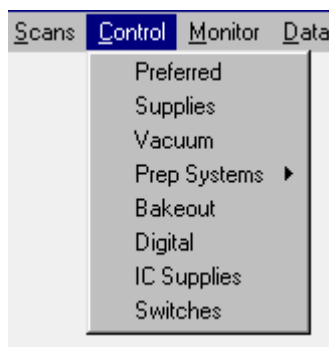
The trip area records the value of any instrument trips which have been detected by the system micro. After the fault has been rectified, this trip area may be cleared by use of the “*Protection*” + “*Reset Trips*” menu option.

To the right of this area, we report electronic errors, again using a bit code to indicate the faulty control. Since the observed output value will only be (even approximately) equal to the set value if the unit is on, the code only checks these setting if the unit is on and often if the vacuum protection is enabled. The reported error may be cleared using the “*Protection*” + “*Reset Trips*” menu option as above, and the fault investigated further using the independent monitor windows.

In the bottom right hand corner we display the current magnet mass, so that the user has a visual check that the correct peak is being analysed during analyses runs etc.

## Chapter 6: Software Operation – The Control, Monitor, Calibrate and Utility menus

### Control:



This section describes the use of the various windows employed in controlling the voltages and valves of the Nu Instrument's Noblesse mass spectrometer. On virtually all outputs, the system micro can provide independent monitoring of the actual (as opposed to set) voltages and this section describes how this information can be accessed. The family of windows under the "Control" button permit most of the required instrument parameters to be altered by the user, as is shown below. Clicking on the relevant menu will result in the required window

appearing in the workspace.

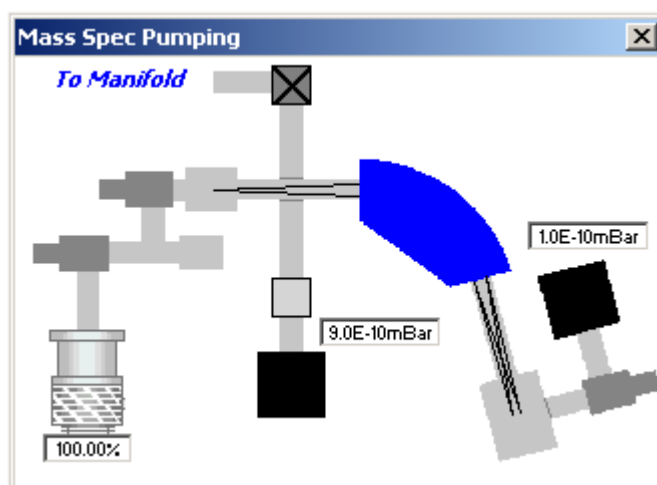
We will describe each window in detail below, in order of (probable) use.

### Preferred

This is a one-press shortcut to load the (most useful) of the windows accessible from this menu. It will load the Supplies, Vacuum, Digital and (if enabled in the Setup menu) the IC supplies windows.

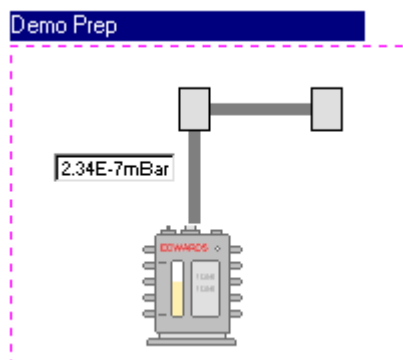
### The "Vacuum" window

This displays the window representing the mass spectrometer vacuum envelope, and displays the pressures of the two ion pumps and the turbo speed, as well as the states of the two pneumatic valves fitted as standard. The window also allows one to open and close these valves, by double clicking on the box representing the valve in the window. The box is dark, with a cross, if the valve is closed, and light if open. A single click over a valve will have no effect, for safety reasons.



The actual schematic was generated using the “Designer” software package supplied with the Nu Noble suite, and as such may be modified by the user, if so required. However, since the valve’s names are also hard coded into the software, please ensure that these names are not altered.

## Prep Systems



This option will appear if the user has created another schematic to represent part of the preparation system, and which is connected to the instrument.

We show the “Demo Prep” window, which is referred to in the “Designer” documentation, in which two valves, a pressure gauge and a rotary pump have been drawn. Due to the intimate coupling between the files generated by the “Designer” package and the Nu Noble software

suite, the valves may be operated in the same manner as with the main “Vacuum” window (double clicking), and the gauge readings are also updated continuously. The software “knows” of the presence of this second vacuum schematic from the relevant entries in the “Startup Parameters.dat” file. Please see the relevant documentation on the “Designer” package for more information.

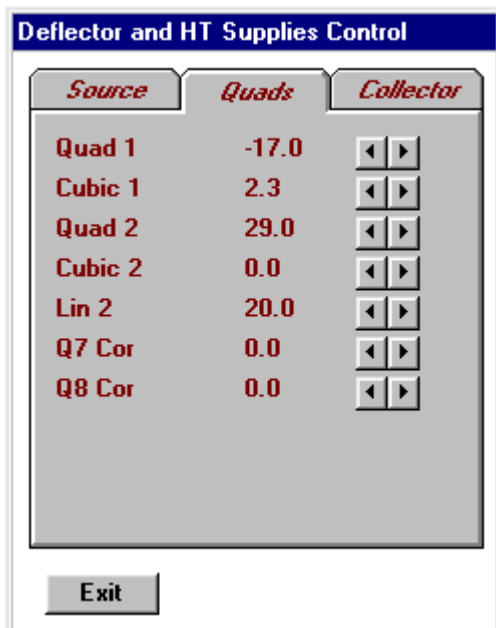
## The “Supplies” window

This allows the user to alter the main instrument source, quad and Faraday collector voltages. As may be seen, the window splits these tree groups into three “tab”, any one of which may be selected by clicking on the relevant identifier at the top of the forms. We show the source tab, in which two of the parameters are “protected” to stop accidental alteration. They may be enabled from the “Protection” menu, as mentioned previously.

The value of any of the parameters may be altered, either by selecting it by swiping the present value with the left hand mouse button depressed, and then typing in the new value. The data is then accepted by use of the “ENTER” key on the keyboard. To avoid the problem of a new value being typed in, but not sent out, the colour changes to green until it has been accepted. Alternatively the value may be changed by use of the spin buttons to the right of the displayed entry. These cause the value to alter in equal steps, whose step size is set in the file “Deflector.dat” as discussed previously. The program will not accept entries which exceed the minimum or maximum values for each parameter, again as defined in the file.

Source	Quads	Collector
Source HT	4000.0	◀ ▶
Half Plate V	3610.0	◀ ▶
Trap	400.0	◀ ▶
Trap Voltage	25.0	◀ ▶
Repeller	-6.2	◀ ▶
Filament V	-73.2	◀ ▶
Delta HP	11.0	◀ ▶
Z Lens	320.0	◀ ▶
Delta Z	-4.5	◀ ▶
Max Power	17.0	◀ ▶

Exit

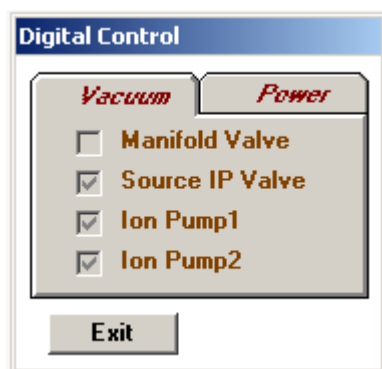


The “Quads” tab has a number of possible entries, and as such warrants further discussion. The Quad 1 and Quad 2 entries are the “normal” quadrupole voltages, applied to the lens arrays. However since they are arrays, rather than simple rod devices, more complicated fields may be applied, to do further interesting things.

The “Cubic” term applies a voltage which scales as  $v^3$ , as opposed to the  $v^2$  of the quad entries. This has two beneficial effects: it rotates the final image plane at the collector and applies an aberration correction to the observed peak-shape. It is not expected that the “Cubic 2” will be used in future.

The “Lin” term acts as a simple deflector. It may be used to image the ion beam onto different parts of the ion multipliers. Although the beams must obviously still pass through the slits, this affects the angle of incidence at the slit. This is useful if you “accidentally” burn one of the multiplier front dynodes and wish to use a different part of the surface. The last two terms allow one to alter the absolute value of the voltages on the outer plates of the array – for further fine tuning of the peak-shapes at the extremes of the instrument’s dispersion range. They may be removed in future.

### The “Digital” window



This is again split into different regions, the “Vacuum” tabbed area and the “Power” area, as shown here. The “Vacuum” entries will be seen to be greyed out. This is because they are returning the observed state of the two vacuum valves (i.e. the state as recorded by the sensor on the pneumatic actuator of both valves), not the state which the system micro requested. The last two entries illustrated here, “Ion Pump 1” and “Ion Pump 2” are checked if the relevant ion pump supply is switched on and communicating

with the system micro. For instrument safety reasons, it has been decided not to let the PC directly switch any of the pumps.

The second “Power” tab is used to control the digital outputs shown, directly:

The “Remote Control” check box, if enabled, will force the filament supply unit to only responds to changes from the PC, i.e. local (front-panel) is inhibited

The “Bakeout” will switch the bake-out power on and off (but it must also be switched on at the mains distribution unit). It is suggested that the automatic “Bakeout” utility discussed below is used to control this feature in practice.

“IC Enabled” switches on and off the high voltage supplies to the ion multipliers (and retardation filter if fitted)

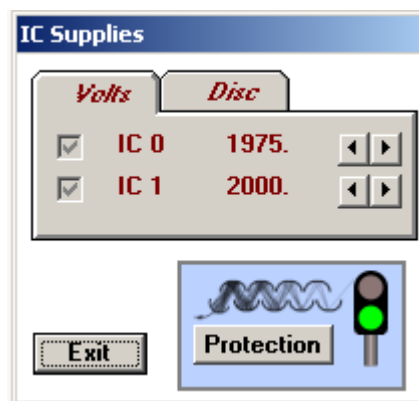
“HT” controls the high voltage and extraction potentials of the source.

If one is switching on the high voltage to the source, the program will temporarily set the HT and extraction voltage to zero, then switch on the supplies, and then slowly ramp up the two voltages to their former values. This is done to minimise the possibilities for flashover.

### The “IC Supplies” window

This option is only enabled if the “IC Enabled” option is selected in the “Setup” window.

It permits the user to control the voltages on the ion multipliers, the discriminator settings for each IC detector channel and the voltages on the retardation filter (if fitted). Since many of these parameters should not be altered without due thought, they are protected via the “Instrument Setting” protection routine described previously.



Also present in the window is a button used to “protect” the multipliers from large signals. It has two states, the “pass” state as shown above, and the protected state shown on the left, where the traffic light is on red and the electron cloud is absent from the multiplier dynode chain. Although you will probably not protect the multipliers yourself, during normal operation (since it will also kill the beams to the faraday), if the instrument is forced to protect them automatically (due to a trip state being reported), you will find that the protect state will be shown. To continue to use the instrument, it will be necessary to manually reset the button, since this is not an operation which can safely be left to automatic control.

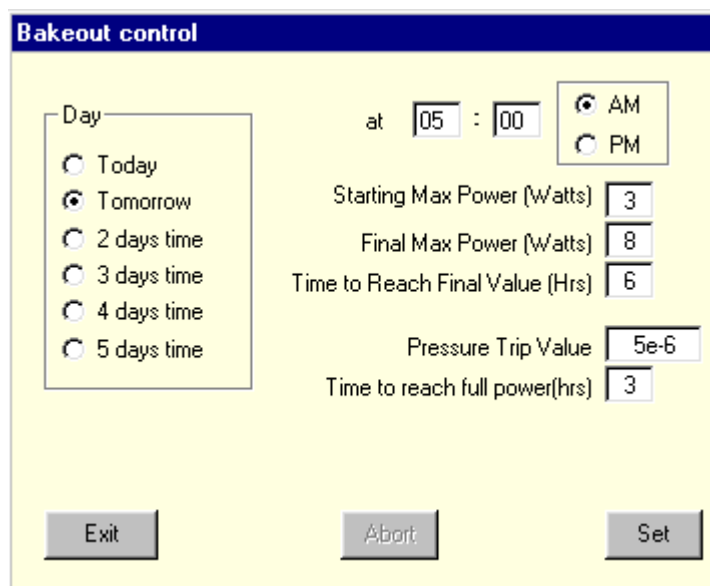


### The “Bakeout” window

This facility enables the bakeout operation to be terminated at a defined time, so as to allow the instrument to cool down before the morning shift arrives. The actual temperature control of the heaters is by the regulators fitted to the mains distribution unit, but there is no power output unless the “Bakeout” digital output control is set. This utility controls this output.

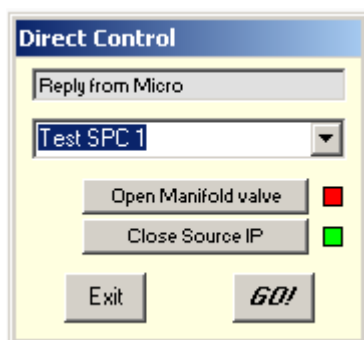
It is also possible to control the power being applied to the filament during the bakeout process from this window. As may be seen, it is possible to ramp the filament from a lower, to a maximum power value over a set number of hours. If this is active, the “Max Power” control on the “Source Supplies” tab (see above” will flash “Bakeout” to warn the user that it may not be set to the normal running condition. This option is only activated if the maximum power is equal or less than the “starting” value at the commencement of the bakeout process. Otherwise the program has to assume that the user has set the desired value automatically, and it will not change the settings. Thus in order to use this option it will be necessary to manually bring the

maximum power down to (say) 3 watts before the bakeout cycle commences. In practice this is not a silly restriction, since it is not sensible to switch the power to the filament to its full value until most of any impurities introduced during the venting have been removed.



Since the program can monitor the observed ion pump pressures, we also provide the facility to suspend the heating either of their values exceeds the input set point. Due to the time delay from ceasing the power to the heaters, and the pressure stopping to rise, if a problem is experienced, the value shown is considered a reasonable maximum. Also, to avoid any problems from a rapid change in temperature, the rate of rise can be limited.

### The “Switches” window



This may not be often used by the user, but permits direct control to a number of digital outputs and intelligent peripherals, fitted to the instrument. The actual captions, and hence commands executed by this window depends on the contents of the file “Switches.dat”, discussed above. In the example shown, we have set the file so that the two standard pneumatic valves are controllable by pressing the corresponding button, and the awake user will notice that the observed states shown agree with the “Vacuum” window shown above. Also a series of “complex” commands are supported, which may be displayed by clicking the downward arrow to the right of the combo box. The command illustrated “Test SPC1” sends a serial string to the ion pump controller “SPC1, and the reply from that unit would be displayed in the top text box. To send these commands, one uses the “GO!” button.

Possibly the most use of this option would be to check that remote “intelligent” peripherals are communicating before a batch run is started. Experience has shown that many of the interfaces of such systems are not as robust as they should be, and the use of a command such as “Send Version Number” or the equivalent, which

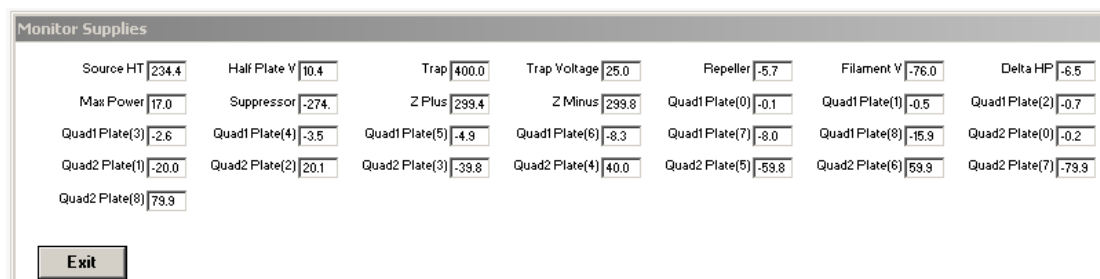
should cause a talking unit to respond with their current software version, is recommended to check communications are OK. We have found a reboot of the controlling computer (not ours!) often reinstates control.

## The Monitor Menus:

The various windows accessible under this menu bar are shown on the left. The may be split into two main areas, those which just report the measured value of an instrument parameter (not the set one), and the last two options which perform a series of tests to monitor the instrument performance. The top option “Window” merely brings up the monitor window, which resides in the bottom right hand part of the main Nu Noble window. It will probably only be necessary to use this option if you are running the program off-line, since the on-line mode will call up the window automatically. The layout of this window was discussed previously.

## The Supplies, IC Supplies, Vacuum and Magnet windows

We will deal with this group together since they are very similar. We show below the form of the supplies window, where it will be seen that most of the outputs controlling



the source and quad lenses are to be found. Many of these parameters were set up from the “Monitor.dat” file, and we show here part of that file to illustrate how these windows may be customised:

```
"VACUUM"; "Turbo1 Speed", "%", "LIN", 2, 0, 10
"SUPPLIES"; "Read Source HT", "kV", "LIN", 2, 0, 1
"SUPPLIES"; "Read Half Plate V", "kV", "LIN", 2, 0, 1
```

The first line in this example (which isn't the first line of the file) shows how the entry monitoring the speed of the turbo will be reported on the “Vacuum” window. The description, "Turbo1 Speed" must match the corresponding name in the MS.DEF file in order that the program knows where to pick up the required information. The final parts of the string shows that the fit of the measured output (nominally a 0 to 10 volt signal) is related to the required parameter (turbo speed in percent). The next two entries are to appear on the “Supplies” window, and it will be seen that (in this case) the leading part of the string (the “Read “ part) has been striped off by the program.

This window also shows that the individual parts of the quad plate outputs are also being monitored. We could have added this to the “Monitor.dat” file, but in practice this part has been hard coded into the Nu Noble software for simplicity.



## The Peak Flat Window

This window will run a peak flat analysis, simultaneously over a number of collectors if so required (one of the marvels of multiple collection!). The form is shown below.

The collectors, which need to be characterised, are selected using the tick boxes (the two ion counters in the example shown) and the masses identified. The program is “intelligent” in that if one mass is entered, the others are automatically calculated (using the mass dispersion and input collector positions from the “Collector.dat” file). The peak to centre on is identified by double clicking the required mass, when its colour changes to purple, as shown for the mass 36 in IC1 collector in the example. One must then also specify the mass for the zero measurement to be undertaken and the time for the zero measurement, corresponding to the mass of collector selected for the “centring”. The measurement process is split into a series of cycles starting at the low-mass extreme of the peak, taking seven measurements through the centre to the high mass extreme, and then back down to the low mass end again. This is done to try to compensate for any beam intensity drift during the measurement process. The user can specify the number of cycles per bloc, as well as the number of blocks in the whole process. The program will limit the total number of cycles to 100.

Collector	Mass 1	Mass 2	Mass 3	Mass 4	Mass 5	Mass 6	Mass 7
IC0	38						
Far							
IC1	36						

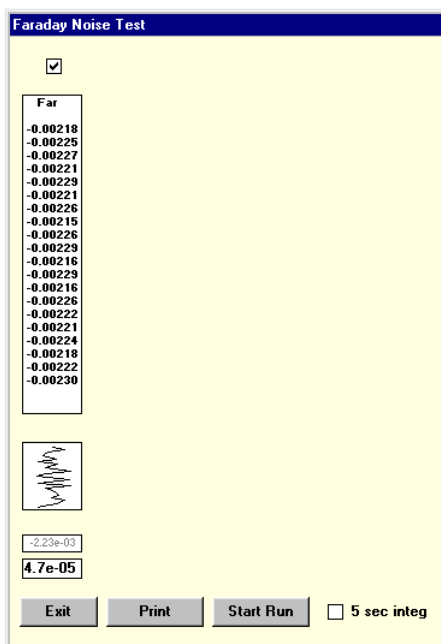
Zero Value: 36.5    Zero Time: 20    Cycles per block: 20    Num Blocks: 2

Scan range +/-: 99

If one wishes to change the total range of the analysis program (in the example it is set to  $\pm 100$ ppm) the new range may be entered in the text box in the lower right hand corner. To accept the value use the “ENTER” key on the keyboard, as normal, when the captions at the top of the window will also change.

To start the run, use the “Start” button. Each block is preceded with a peak-center run, and after each block is finished the intensities for the peaks, relative to the centre intensity, is reported in the relevant slot. This feature may be the main reason to split the measurement into a number of blocks, since the total program may take quite some time to run, if a large number of cycles is selected. The program also reports the standard deviation of each measurement. If the measurement is noisy, and the error is large, it will default to printing a set of stars, so as not to overfill the text box.

## The Faraday Noise Test



This utility records 20 individual measurements on all the faraday collectors present on the system, and then calculates the noise on that measurement. By default the integration time of one second per measurement is used for each measurement, but this may be changed to 5 seconds by checking the box in the bottom right of the window.

To ensure that you are only measuring the noise of the detector, and the associated electronics, it is obviously necessary to ensure that there is no ion beam (even a small one!) incident on the detectors.

The output may be printed, after the run, if a hard copy is required.

## The “Calibrate” Menus

Two utilities are provided under this heading, as shown here.



### The Magnet calibration routine:

This utility is used to provide a least squares fit of the observed mass against the required digital to analogue converter output in the magnet controller, the form being shown below.

As discussed in the section on the magnet scans, it is possible to obtain the DAC settings at any time from that window. Thus using a series of impurity peaks, or peaks from an air slug, a set of data spanning the required range of use of the instrument may be obtained. These can then be input into the text boxes at the top of the present window, the desired order of fit selected (usually 2 or 3) and the fit calculated via the use of the “Calculate” button. The data set may be reanalysed as often as required, changing the fit order or removing or editing entries.

To select an entry from the editable section of the table, simply double click the required entry, which will then result in the data being transferred into the data entry text boxes. Once a satisfactory fit is achieved, the data may be stored to file using the “Save” button.

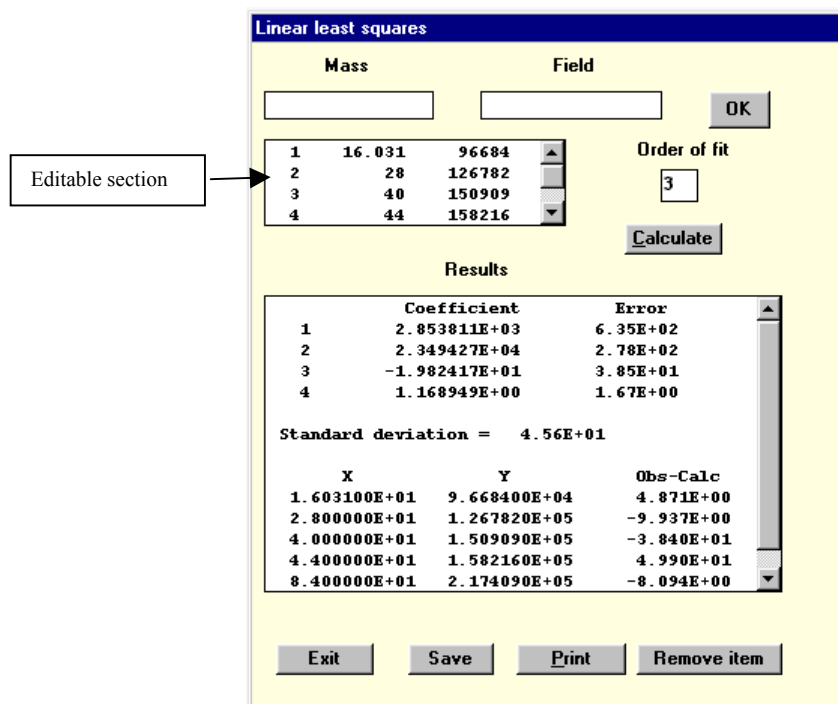
Since the formula relating the magnet radius, mass of the ion and the magnetic field is of the form:

$$\text{Radius} = K \times \sqrt{(\text{mass})} / \text{field}$$

where  $K$  is a constant at fixed acceleration energy, we see that to first order that the magnetic field is proportional to the square root of the mass. We have therefore used a routine to fit the DAC setting (i.e. field) to the square root of mass, in the form:

$$\text{DAC setting} = A + B * (\text{mass})^{1/2} + C * (\text{mass}) + +$$

This approach will be found to be much more satisfactory than the more normal approach of fitting DAC steps to a straightforward power relationship in mass, when higher orders in the fit are required.



The user should be aware that when a new data set is stored, any data stored by the peak centring routine will be obsolete. As such the files associated with this routine (see below) are purged, and the program will start with a new slate.

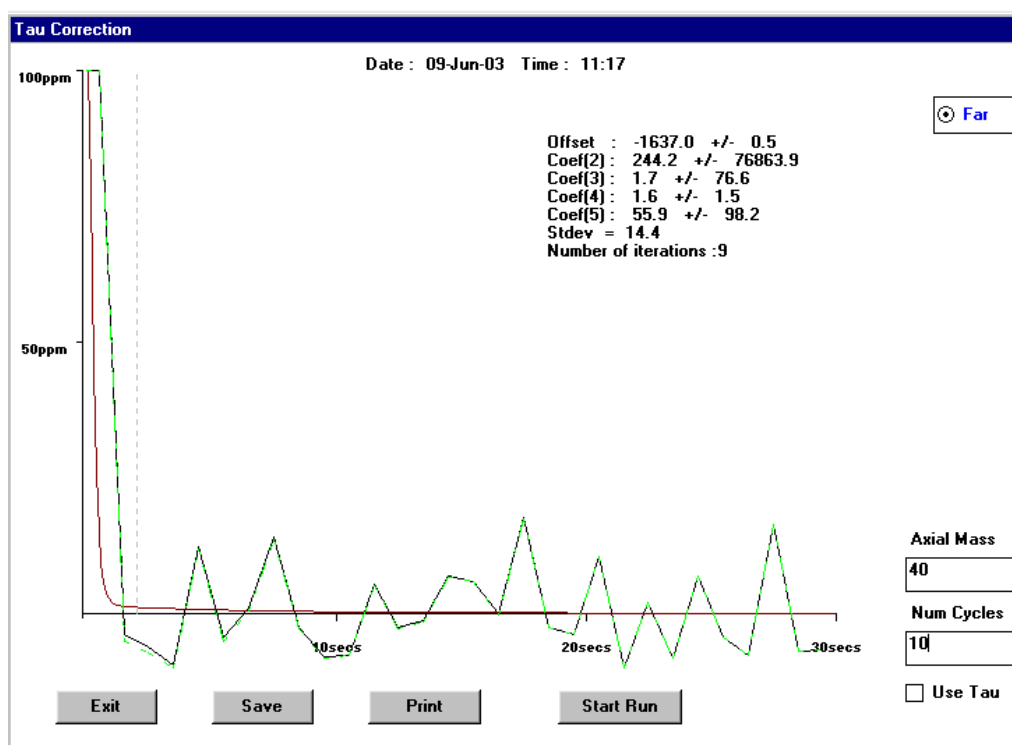
### The Tau Correction routine

The “Tau correction” refers to the decay experienced by the faraday detectors when the incident intensity suddenly changes. Since the feedback resistor in the preamplifier is so large (normally  $10^{11}$  ohm), any small parasitic capacitance can result in unexpectedly large decays as the intensity changes. Thus a 1 pico Farad capacitance will give a decay constant of 0.1 secs, which will mean that the signal will decay to only 45ppm of the original intensity after one second. This is the limiting factor for undertaking fast measurements with a faraday, since in practice the laminated magnet can normally move faster than this.

To compensate for this effect, we have provided a utility to measure this decay, and the analysis program then uses this data to “correct” for the observed decay to emulate the response of an “ideal” system with minimal stray capacitance.

The program performs the following series of actions:

Firstly the beam is switched off the collector and the zero is obtained. This value is not critical, since it is merely an offset, but a reasonably accurate value ensures that the displays are drawn sensibly during the run.



A series of measurements follow, in which the beam is placed onto the detector for 10 seconds, and the switched off (by going to the half mass position). The intensity of the observed signal is then recorded for the following 30 seconds. The data is integrated for the required number of cycles (10 in the given example), and at the end of the measurement the data fitted to a double exponential decay. The program then shows the “ideal” fit, the observed data set, and the observed data set corrected with the ideal fit.

If the run is satisfactory the data should be saved.

### The “Utilities” menu

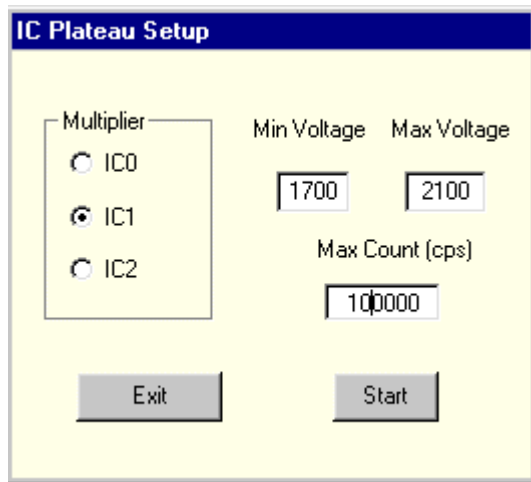
Three menu options will be discussed here, the other two (“Reanalysis” and “Create Auto Sequence”) being more conveniently covered in later chapters.

Two of these other utilities are provided to produce low level control of the instrument, and may well not be ever used on site. The “Hysterisis” routine will enable the magnet to be de-gaussed, if ever required, by sending it through ever decreasing field cycles (starting with full maximum to full minimum, reversed, field extremes). The window also “hides” the routine to download the parameters to the magnet slave unit (See chapter 2), which may be accessed by dragging the right hand edge of the window sideways.

The “Interrogate Micro” utility will allow the user to directly access the individual memory locations of the system micro. The utility accepts both integer and Hex word input, and can both read and write to the designated memory location. It also accepts

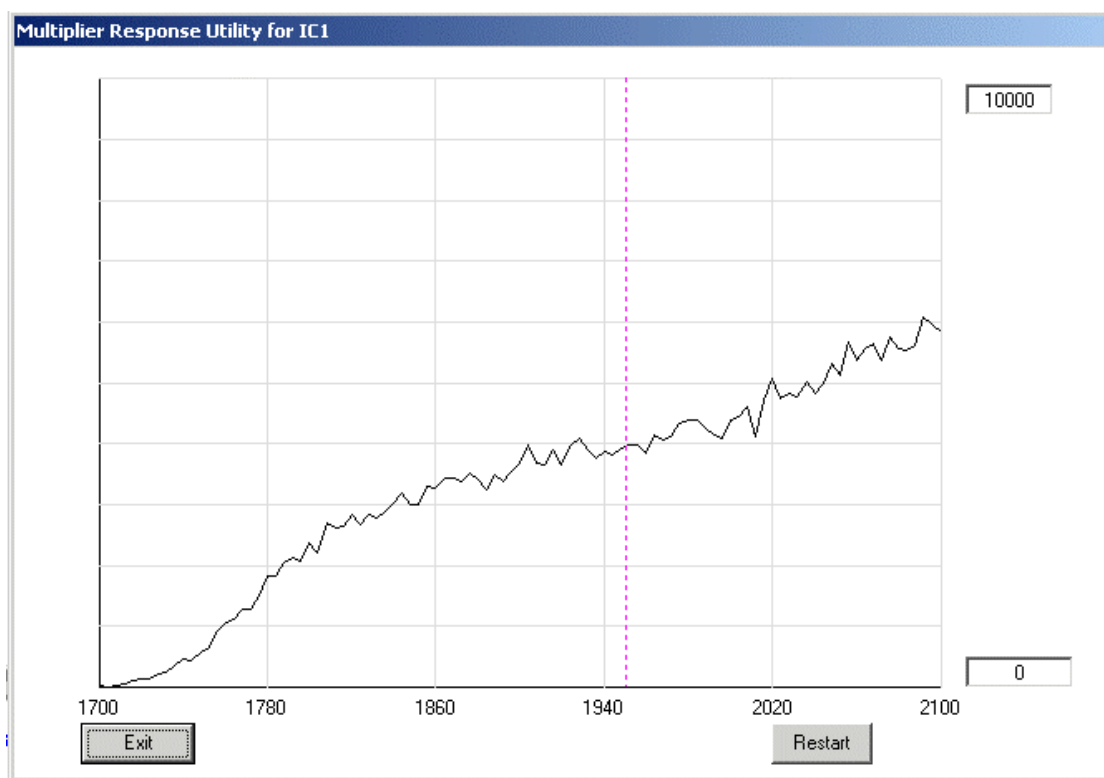
both the normal (16-bit) word and (32-bit) long word formats, selected using the check box on the form.

The final utility discussed here, enables the multiplier running voltage to be found.



Basically the utility will alter the tube running voltage and plot the observed response as it does so. To use this utility, firstly set the magnet so that a peak of a few thousand cps is incident on the multiply you wish to use. We assume this is IC1 in what follows. The select the "Find IC Plateau" menu option, when the window shown here is displayed. Select the required detector from the list, and change the values shown for the voltage range and maximum beam permitted, if the default values are not suitable. Hitting the "Start" button will cause the following

window to pop up, and the scan will automatically start. When completed, the entry multiplier voltage will be shown as the dotted line. If you wish to change the set voltage, do so via the "IC Supplies" window, and (probably) repeat the plot using the "Rescan" button.



# Chapter 7: Data Acquisition

## Overview

This chapter describes the data acquisition process with Noblesse.

We start by describing how to create a basic mass-data-file (mdf file); this file specifies which masses and collectors to use and measurement times.

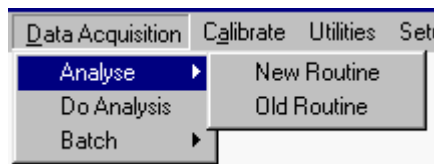
Having created the mass data file; we must then create a corresponding calculation-run-file (or NICE file). NICE is *Nu Instruments Calculation Editor*; we use it to obtain useful results by doing calculations involving the parameters measured by the mdf file (for example, subtracting a detector baseline, or calculating the ratio of two isotopes).

Next we describe how to start a single data acquisition run.

Finally, we describe batch and sequence files, to show how Noblesse may be used to obtain data in batch mode.

Data *analysis* is described in chapter 8.

## Starting from Scratch – the mass data files



We will firstly discuss the setting up of the analysis files which will define the sequence of events to be undertaken during the analysis. Assuming that a completely new analysis is to be undertaken, select the “New Routine” from the

pull down menu, when the window shown here will appear:

We will now describe each entry section in detail.

*Collector identification:* The program produces a series of check boxes, representing the approximate position of the collectors fitted to the instrument, as shown above for a quad (3×IC plus high mass Faraday) configuration. This information is obtained from the “Collector.dat” file. Ensure that you check the detectors that you wish the program to monitor during the analysis you are designing.

*Number of zeros and steps:* This defines the number of zero measurements (obviously), up to a maximum of two, to be carried out prior to each sequence of beam measurements. We call the number of succeeding different beam measurements the “number of steps” in this table. Thus suppose an experiment was being designed using only the axial Faraday (in this example configuration) and you wished to perform a series of measurements of argon, recording the following sequence for a number of times:

Measure **zero 1** at mass **39.5**  
Measure **zero 2** at mass **40.5**  
Measure mass **40**  
Measure mass **36**  
Measure mass **37**  
Measure mass **38**  
Measure mass **39**

This set of measurements would be identified as two zero measurements followed by five steps of measurement. The actual values may be selected using the up-down spin buttons at the right hand side of the relevant entry box.

*Use Standard Mass Dispersion:* When the next window is displayed, the program will try to be “intelligent” and fill in all the masses for each cycle, when any one mass is entered. This can obviously save a lot of typing, but it does need to make a number of assumptions to get this correct. The definition of “standard” is the same as discussed previously under the Quad Values window discussion in Chapter 5, and will not be repeated here. However if any one of the analysis cycles does not use the standard dispersion value for the instrument of 480mm, uncheck the tick box and you will need to give the (approximate) values in the next window. In practice this is only likely to occur for Xenon analyses.

*Peak Centre Mode:* There are two possible analysis modes to use with the instrument. The first, most common one, assumes one has a clean, single peak, and the program can undertake a “standard” peak centre and sit in the mid point (as defined as the mid way between the half height at the start of the peak to the corresponding half height position at the end of the peak). The other mode of operation analyses a peak which has a shoulder before (or after) part of the peak containing two components. The peak centring routine will then find the half height position at the start (or end) of the observed peak profile, and sit at a defined offset position from this point. The section describing the magnet scans described how to set up this offset. It is possible to mix these two types of requirement in an analysis, but if any of the peaks are of the shoulder type, it is necessary to select this option at this stage, and you can define which step does what later.

*Quad Control:* The Nu Instruments’ Noblesse mass spectrometer is unique in that it can allow multiple collection of beams, and change the dispersion during the analysis. This enables coincident detection to be achieved at differing axial masses. To achieve this, the voltages on the quadrupole lens arrays are altered during the analysis sequence, and the option boxes in this section define the three ways this may be done. Obviously no change in quad settings will not maintain exact coincidence

during the analysis, but the option has been added for completeness. However, it is also the option used (obviously) if only a single collector run is being contemplated. The “Best Calculated Value” option will use a least squares fit of the data saved by the “Quad values” form, again emphasising why this data should be accurate.

*Preferred analysis Method:* Since different methods are provided to analyse the time decaying data, it has been noted that different methods may be optimal for different gases. By defining the preferred method with the analysis file permits multi gas analyses (eg Helium followed by neon) to be more conveniently undertaken. The choice may be altered during reanalyses by altering using the “Setup” + “Preferences” menu, discussed in Chapter 5.

Once this form has been completed, you can proceed to the next one by depressing the “Continue” Button. We will firstly look at a simple, single collector analysis and then continue with a more complicated example.

### A simple example:

	Zero 1	Zero 2	Step 1	Step 2	Step 3	Step 4	Step 5
IC0	---	---	---	---	---	---	---
Far							
IC1	---	---	---	---	---	---	---
Int Time							

Max number of cycles:  Magnet delay time:

Zero each cycle

The window corresponding to the argon analysis described above is shown below. No masses have yet been entered, but we were in the process of typing in the mass value for the first zero cycle, when the screen capture was made. Moving the cursor to the relevant box and pressing the left-hand mouse button will result in the vertical flashing cursor to appear, and the required data can then be entered. Once it is input, the data is accepted using the “ENTER” key of the keyboard, as usual. Thus we would type 39.5 into the first text box illustrated, if we were setting up the analysis discussed above.

Once the masses have been input, the required integration times for each measurement should be decided upon and entered. The magnet delay time (the time between the end of one measurement and the start of the next, allowing the magnet to settle after the move, and the signal to decay to a reasonable level, as discussed above under Tau Correction) and the total number of cycles input should then be decided and input. If



you do not wish to undertake a zero measurement each cycle (for instance if you are getting this data from a previous analysis and passing the data to this run (see below) then uncheck the relevant box. (If you do not wish to record any zero measurements as part of the analysis, the set-up form will allow a zero input for the number of zero cycles.)

Finally you should decide which peaks should be used for peak centring. This is done by double clicking the required mass, whereupon the selected box will have a yellow background, to mark the selection. Double clicking again will undo the selection.

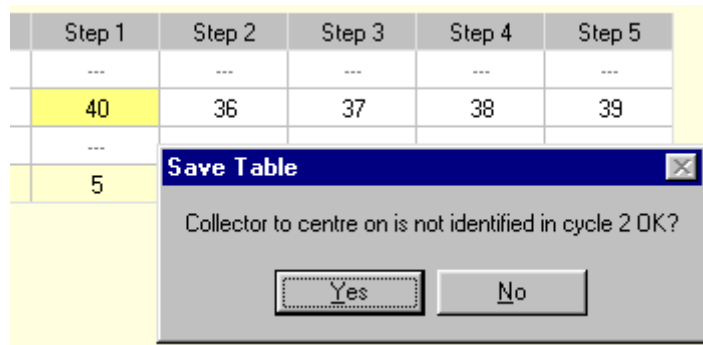
The 'rules' regarding the centring requirements are as follows:

It is permissible to have no selection, as for example would be done for "Blank" runs. In this case (obviously) no selection should be made. In this case the program will search the database kept by the peak centring program to find the last time that the mass was successfully centred, and it will then use the stored "correction factor" to extrapolate from the typed input mass to the actual, observed, central mass. If no data is present (the peak had not been previously centred) the program uses the values of the correction factor for the first peak it finds in the list of the required masses, starting by searching for entries for masses entered in the table prior to the one in question, and if none are found, the program will search for the succeeding mass entries.

Any number of beams may be selected for centring, but if any are, then the first measurement step must be one of them. The reason for this is due to the way the program allocates the "correction factor" to go from the input mass to the actual, observed, central mass. The program will always search for the data for previous mass entries in this case. This is done to allow for the case where one analysis is used to measure two elements. Thus a simple Argon followed by Neon analysis could use the mass 40amu beam to determine the correction offsets for all the following argon beams in the list, and the neon 20 beam could then be centred to give the correction factor for the subsequent neon peaks.

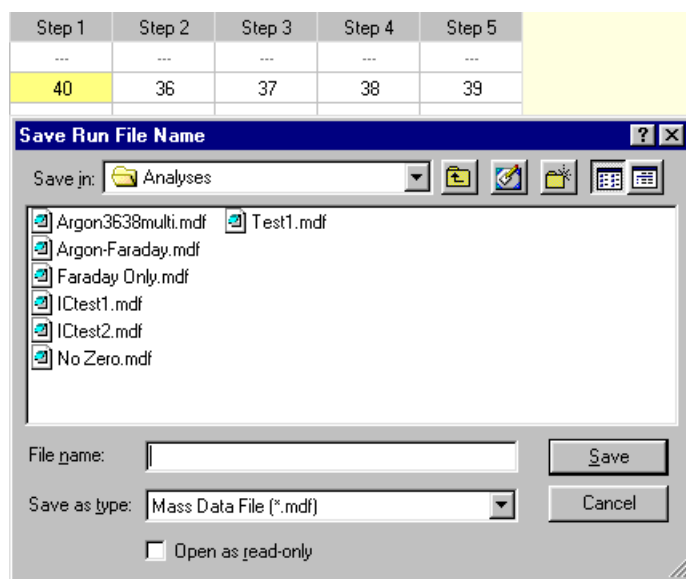
The analysis program will also use the correction factor measured for the first measurement step to calculate the actual mass to be used for the zero measurements. This is unlikely to be a problem in blank measurements, again making the requirements less severe for the case where no centring is requested.

Once the table has been set up, the data may be saved using the "Save" button. The program then performs a series of checks before proceeding, to ensure all data has been input and is valid. If the number of centring masses is less than the number of analysis steps, it obviously does not know whether this is correct, or that an error has occurred and the table has accidentally not been set up properly. In this case the following message box will appear:



In this example, centring is only required in the case of the  $^{40}\text{Ar}$  beam, and its correction factor will be used for the subsequent measurements. Since this is intended, answer “Yes” to the question (the default reply) and the program will continue with its checks.

Once these have been successfully completed, the standard “Save” Windows dialogue window will appear, permitting the user to save the file with whatever name is required. To enable the Nu Noble program to find the file during subsequent analyses, the extension is restricted to the MDF type shown. Also note that the user will have to provide a program to convert the observed beam intensities into required isotopic ratios, and this will be achieved using the Nu Instruments Calculation Editor program, which is discussed later. However the files created by that program must match the names of the files saved here, again to allow the Nu Noble software to find these programs.



### A more complicated example:

Here we have chosen just about every combination possible, to illustrate the various entries.

	Zero 1	Step 1	Step 2	Step 3
IC0				
Far	...	...	...	...
IC1				
Quad 1				
Quad 2				
Int Time				
Dispersion	480	480	480	480
Use Offset				

Firstly, we unchecked the “Use Standard Dispersion” box, since we will set up a step measuring  $^{128}\text{Xe}$  with  $^{134}\text{Xe}$  (which can be regarded as “standard”) as well as one simultaneously measuring  $^{129}\text{Xe}$  and  $^{134}\text{Xe}$ , which is not. Thus entering the mass “134” in the step1,

IC0 box will produce the entry “128” in the IC1 box, but to obtain the required “129” entry for the second step, one has to increase the dispersion to about 580. Note two things here. Firstly the form loaded with the standard dispersion value filled into all the entries of the relevant, dispersion, row. Only in the case of the steps that are non-standard is any extra data required. Also you can use the form itself to estimate the required dispersion value. Simple keep trying different values as you re-enter the first mass, until the other masses correspond to their required values.

Since the quad values will be completely different for these two measurements, we have selected to type in their values into the table. They can be found most easily using the “optimise” feature in the magnet scans window, as described previously.

Finally we have assumed that in one of the cases, we will study the peak in a doublet form. In this case we have to indicate this by double clicking the “Use Offset” box for the corresponding step, whereupon a “YES” will appear in that box.

The final, completed, form is shown here. We have assumed that the 124 peak has a small impurity under it, which can be resolved out to produce a doublet with a shoulder for the pure  $^{124}\text{Xe}$ . Notice also the use of two slightly different input masses to represent the  $^{134}\text{Xe}$  peak, which will be centred on

	Zero 1	Step 1	Step 2	Step 3
IC0	132.5	134	133.95	130
Far	---	---	---	---
IC1	126.5	128	128.95	124
Quad 1	6	6	-20	6
Quad 2	-49	-49	27	-41
Int Time	10	5	5	10
Dispersion	480	480	580	480
Use Offset				Yes

in two different steps. The reason for the use of these different masses is because the peak centre routine keeps a database of the successful centring sequences, and we wish to distinguish between the case of normal zoom and the larger dispersion case. This ensures that the program doesn't try to use the data from the first peak centre in the second case, where the mass correction factor is expected to be completely different, due to the differing zoom values.

### Modifying an old analysis routine

The routine to be edited may be selected via the “Old Routine” menu option, whereupon one can select the required routine for the “Open File” dialogue box. The program will then load the completed table for you to edit as required.

You must now provide the Calculation code to go with this analysis.

## The Nu Instruments Calculation Editor (NICE)

This program is provided to interface with the main control suite of the Nu Instruments Noblesse mass spectrometer. The control programs themselves do not convert observed beam signals into “useful” values, such as isotope ratios, since there are too many different possible combinations. It would not be sensible to try to hard code all these possibilities. Instead we have taken the approach of separating the

recording of the beam intensities from their conversion into “useful” data. What happens in practice is as follows.

The main controlling software will record the beam intensities on the detectors selected. These intensities are then sent across to a second, independent, routine where the individual values are combined to give the required results, they are then sent back to the main program to be displayed, and statistically treated, if required. Thus suppose we have a simple two-collector experiment, where beams are recorded on detectors number zero and one. The signals recorded by detector number zero may be 2.5 units, whilst that recorded on the detector number one is 7.5 units. This would be all that the controlling software is aware of, but the user may be interested only in the isotopic ratio given by the ratio of these two signals. (We will ignore such things as zero offsets in this example!) It is the job of the second routine to do this conversion. In this simple example the formula may be:

$$\text{Required Result} = \text{Signal}(1) / \text{Signal}(0)$$

which then gives the measured isotope ratio of 3.0.

This number can then be sent back to the main controlling software.

As will be obvious from this simple example, there is one more piece of information, which we need to send to the controlling software suite, namely the description for the result. It is of little use displaying a table of the form:

First Result	1.2345
Second Result	2.3456
Third Result	3.4567

Rather we might wish the display to be in the form:

132 / 130 observed ratio	1.2345
134 / 130 observed ratio	2.3456
136 / 130 observed ratio	3.4567

The second job of the editor is thus to define these titles.

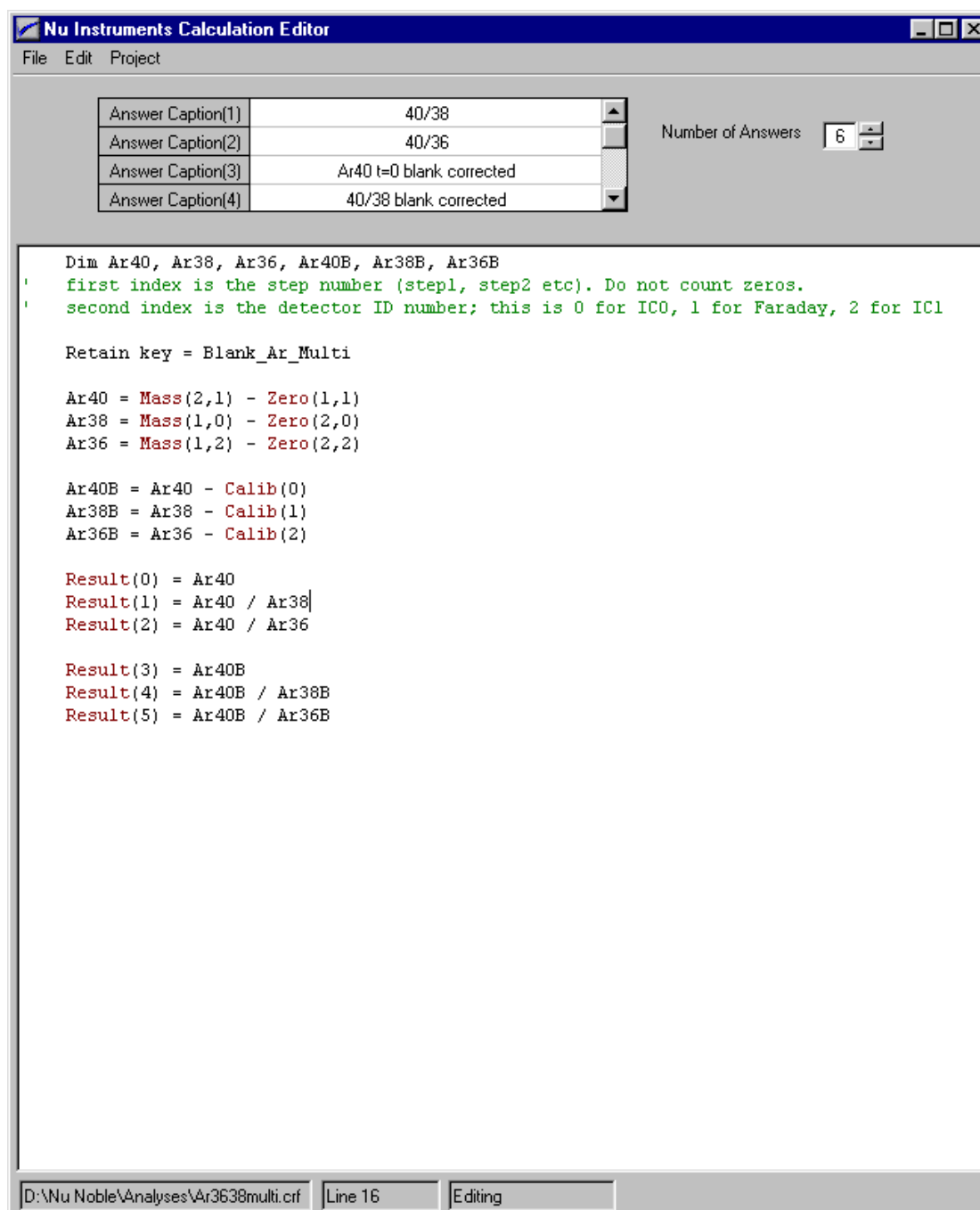
For most noble gas analyses, it is necessary to take into account the variation of the beams during the analysis time, and extrapolate the data to obtain the signals at gas inlet time. The Noblesse software has a comprehensive set of curve-fitting features, which are described fully in chapter 8. For the time being, we note that the crf file must be set up prior to acquiring the data, independently of the details of the curve fitting.

### **Running the Editor:**

The editor may be launched via the *NICE* icon on the desktop. The basic form of the editor can be seen from the example shown on the next page, which has obviously had some code typed into it.

The menu bar is of the conventional format. Files may be read in, saved or printed by pressing the “File” menu and choosing the required option from the drop-down list. Editing of the code may be undertaken via the “Edit” menu options, although the standard shortcut keys also work:

Ctrl + “z”	Cut
Ctrl + “c”	Copy
Ctrl + “v”	Paste
Ctrl + “z”	Undo.



Since the compiler is line-number aware (you will already have noticed that the status bar at the bottom of the screen shows that the cursor is on line 16 – in fact on the

extreme RHS), we have also added a “Go to” line option under this heading. The code may be “compiled” into the required format, by using the “Project” menu.

New analyses may be saved via the “Save As” menu option, which will prompt the user to type in the new file name, whilst the “Save” option will overwrite the original file with the new one. The original file name is shown in the leftmost panel of the status bar on the bottom of the editor screen. **The name used should be the same as that of the corresponding mass data file** (“Ar3638multi” in the example shown), and is saved with the “CRF” (Calculation Run File) extension.

### **The Editor language:**

A simple glance at the above example will show most (but not all) of the language syntax.

The top of the code area should hold the declarations of all the variables and constants to be used in the program. The compiler is very “strict” in this manner – everything must be declared, but then any “typos” or other simple errors will be found early on. All variables are treated as 32bit (double precision) numbers and there is no requirement to distinguish between integers and non-integers. All constants declared here (see below) are similarly held to this precision. The names are totally case insensitive. Thus the following are all equivalent:

NAME1                    name1                    Name1                    NaMe1,

and any other combinations you wish. Obviously some are more sensible than others.

The words “Dim” and “Const” are two examples of what is called “reserved” words. These are words which have a special meaning to the compiler, and may not be used as general variable names. The full list of these reserved names is:

- Dim
- Const
- Retain
- Retain Key
- Mass( , )
- Zero( , )
- Calib()
- Result().

and their meanings will become apparent as we progress.

Comments can appear anywhere in the code. All characters after and including the single quote (‘) are ignored by the compiler. To aid readability, these comments are shown in green by the editor, as soon as the line is changed. We recommend the full use of comments, since you will probably not be able to follow the logic of the program when you revisit it after a long absence if they were absent.

The observed intensities measured by the main instrument controlling software are held in the arrays “Zero(X,Y)” and “Mass (A,B)”. The first of the two parameters

inside the brackets define the step of the measurement sequence. Thus for the Nu Noble software suite, two zero steps are possible, and twenty measurements steps are permitted, giving the possibility of X having the value 1 or 2, whilst A can range from 1 to 20. Unfortunately it is not possible for the compiler to check that valid parameters are entered here, although the main program will check the compiled code, so please try to ensure their validity as the code is written. The second parameter (Y and B respectively), define the collector identification, starting with collector zero as the one in the highest mass position. The numbering goes from zero to the total number of collectors minus one. *Be careful with this notation:* in the examples shown, the collectors were IC0 (high mass side), Faraday (axial), IC1 (low mass side). It does not necessarily follow that collector IC $n$  is always collector  $n$ ! Normally we are not interested in the values of these variables, but their corresponding differences, which is the signal due to the beam (removing any electronic offsets), and the first part of the illustrated code shows these values being obtained:

```
Ar40 = Mass(2, 1) - Zero(1, 1)
```

Thus the Argon40 beam is the difference between the intensities measured on collector one in the second measurement step and the zero beam measurement, etc.

The answers to be returned to the main instrument controlling software are given by the array “Result(Z)”. Up to thirty results may be determined, corresponding to the Z parameter having a value between 0 and 29 inclusive. The actual number being calculated may be set by using the up-down “Spin” selector at the upper right of the window. In the given example, six results are calculated, but not all of the code is visible in the window. The scroll bar at the right of the code window would have permitted the rest of the code to become visible in the real case. The answer caption corresponding to each of these returned results may be entered in the table above the code region, which in this example is displaying the first four captions. Ensure that the “ENTER” key is pressed after changing or adding a new caption, to force the program to accept it.

As implied above, it is also possible to define constants in the body of the text. Thus for example, we could define True\_R as a variable, and add the line:

```
True_R = 0.7219
```

or use the definition :

```
Const True_R = 0.7219
```

at the top of the form after the “Dim” statements. This approach is not recommended however, since it will only store the constant to 16 bit (single precision) accuracy, and the first method of defining the constants at the top of the program is strongly recommended.

The arithmetical functions supported are:

+ (add), - (subtract), \* (multiply), / (divide), ^ (raise to the power), Sqrt (the square root of), Exp (exponential), Log (natural logarithm) and = (equals).

To aid readability of the source code, the editor will check that any constants or variables match the case as typed. Thus if the variable “Ar40” is defined at the top of the program, the editor will match this typing all through the code, with the first letter being upper case. Similarly, if you decide to change this at a later stage for some reason to “ar40”, the editor will automatically update all references throughout the text. Further, as a second aid to readability, reserved words such a “Mass” and “Zero” appear in a different colour to the main text. Finally, the number of spaces is automatically sorted to sensible values.

### Passing data from one analysis to another:

Sometimes it is useful to use data from one experiment in a subsequent one. One example of this would be if one wished to measure the “blank” rather than use beam zeros in the calculation process.

To facilitate this process, we have provided the “Retain” feature. To illustrate this, we will consider the case where the blanks are measured in the first experiment, and their values used in a second (as shown in the example above).

#### 1) *The calibration process (saving parameters for use in a later analysis)*

The required code to do this is shown below:

```
Dim Ar40, Ar38, Ar36
' first index is the step number (step1, step2 etc). Do not count zeros.
' second index is the detector ID number; this is 0 for IC0, 1 for Faraday, 2 for IC1

Retain Key = Blank_Ar_Multi2
Retain Result(0) as Calib(0), Result(1) as Calib(1), Result(2) as Calib(2)

Ar40 = Mass(2,1) - Zero(1,1)
Ar38 = Mass(1,0) - Zero(2,0)
Ar36 = Mass(1,2) - Zero(2,2)

Result(0) = Ar40
Result(1) = Ar38
Result(2) = Ar36
```

The “Retain” statements tell the compiler which of the results are to be stored, whilst the key ensures that the correct calibrations are picked up by the following analyses, and not some determined in an entirely different program.

#### 2) *The use of these values in another routine (reading in parameters from an earlier analysis)*

At the top of the code in the program used with the samples, add the same Retain Key:

```
Retain Key = AnyNameYouWishAsOneWord
```

You can then use the Calib variables in the code that follows, and the main program will inset the final, averaged, values determined by the previous standard run. The printout of these analyses will show these values amongst the heading information, to



enable you to be sure of what has happened. Note how the example given allows for any changes in electronic zeros between the two measurements (although this should be minimal).

### The Compilation Process:

To initiate the compilation, use the “Project” + “Compile” menu tree, as mentioned above. This option is only active if the source has been saved.

The compiler will try to find any errors before finalising the compilation process. The process is stopped at the first error found, the form of the error indicated in a message and the editor then highlights the line in which the problem is located. If no errors are found, it will produce a file with the “NCC” (Nu Compiled Code) extension, which is used by the main program to obtain the required displayed answers. The analysis file, calculation file and compiled file should all be in the same directory, since the main instrument program uses the selected path of the analysis file to look for the compilation code.

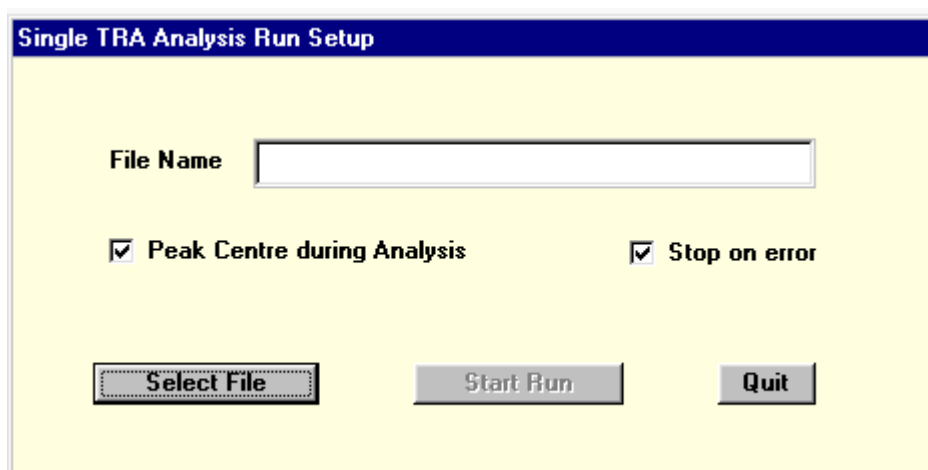
Successful compilation is indicated in the rightmost panel of the status bar at the bottom of the editor window. Although the compiled code will be automatically stored, the source code is not. Thus if you have to make corrections to get the compilation process to go to completion, please ensure that you also save the final source. Obviously any subsequent changes will also require the new code to be saved and the new version has also to be recompiled.

Some errors may only become apparent once the analysis is run. The most common is when one of the variables has a zero value due to a mistake in the code and it is used on the bottom line of a division. With previous editors, the user is given little assistance in finding where the problem lies. However we have designed this editor to be as user friendly as possible, and if such a problem does arise, the line where the error occurred will now be indicted by the main instrument program.

## Single Data Acquisition Runs

### Preliminaries

Clicking the “Do Analysis” menu option under the “Data Acquisition” header will result in the following window being displayed:



This will allow the user to browse the directories for the required (.mdf) analysis routine, via the standard “Open” dialogue window. The two options on the window allow the user disable the automatic peak centring when the run is started, and to continue if an error is detected, rather than the default of stopping in such a situation.

Once the file has been selected, the “Start Run” button is enabled and on pressing it the main analysis window is loaded.

### **The Analysis window**

When originally loaded, it is fairly bland, with only the predominant “Start” button catching the eye. However the user will note that the program has picked up the active collectors and displayed them in the centre top of the window. Also the sample name text box can be filled in at this time (and it is probably sensible to do so before it is forgotten).

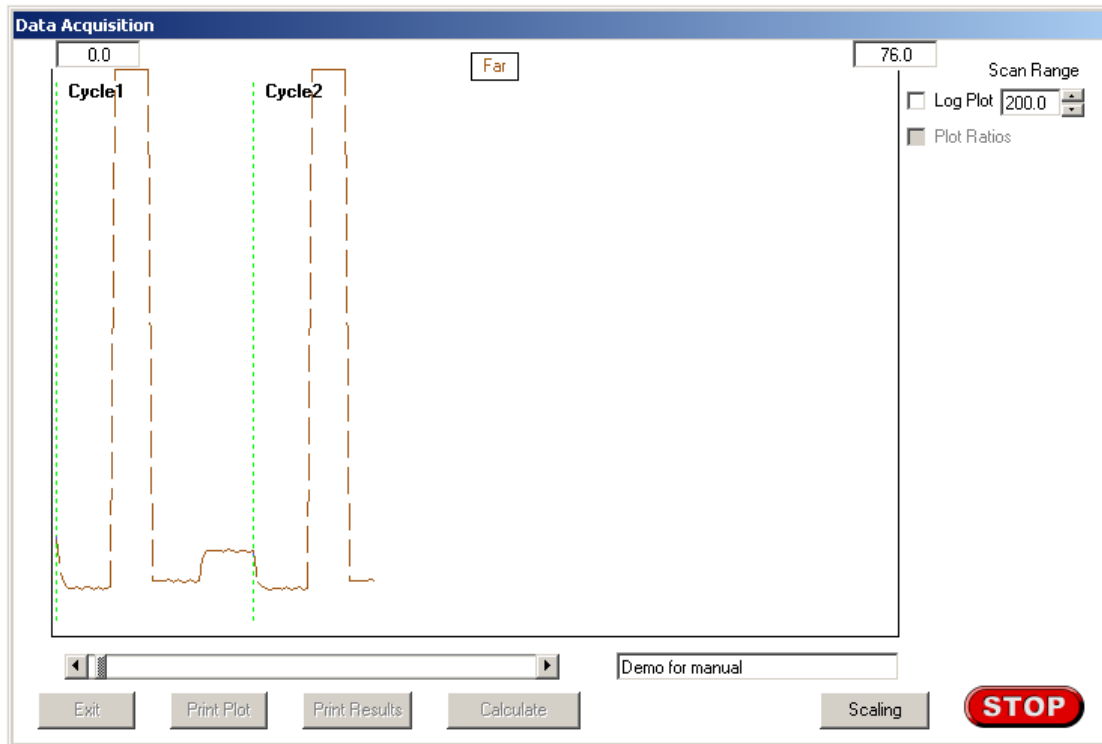
Once started the program will reference all timings to the time the start button was pressed, so if you are letting in a sample from a manifold it is best to start the analysis as soon as the manifold valve is opened. Ways to allow for this accurately during automatic runs are discussed in Chapter 9. Obviously this is easier if the run is automated, but that comes later!

After the start is pressed, the program will automatically peak centre the required masses (as long as this option wasn’t disabled in the window shown above), and then start the data collection process. In a simple analysis this will involve:

1. Move the magnet to the position for the first zero step
2. Wait the “magnet settle time”
3. Measure the first Zero intensities
4. Move the magnet to the position for the second zero step (assuming two zeros)
5. Wait the “magnet settle time”
6. Measure the second Zero intensities
7. Move the magnet to the position for the first measurement cycle
8. Wait the “magnet settle time”
9. Measure the first set of beam intensities

Etc etc.

The program uses one second integration periods for all its data collection, and the current time during the run will be shown in the right most text box at the window top, 76 seconds into the run in the example shown here. Please note that this time is the time since actual data acquisition started (i.e. excludes the time for peak centring etc) and not the time from the commencement of the run the pressing of the start button. The difference is stored in the memory and used in any analysis.



The program plots all the beams being recorded, and hence the trace can be a bit confusing if more than a couple of detectors are enabled. By default the program picks up the beam scaling (maximum and minimum values) from the magnet scans window.

In the trace shown, the recorded signal is showing the following measurements:

1. The zero measurement
2. The  $^{40}\text{Ar}$  signal (off scale here)
3. The  $^{38}\text{Ar}$  signal
4. The  $^{36}\text{Ar}$  signal

The process is then repeated. The green vertical line indicates the separation between each cycle of measurements, and each cycle is identified, as shown.

For the form, it is also possible to:

- Plot the data in a log, rather than linear, format
- Plot the isotopic ratios, rather than absolute intensities (if more than one beam is measured!). All plots are to the same denominator signal.
- Alter the scaling of any, or all, of the recorded signals.
- Look back at previously recorded data using the slider bar at the bottom of the trace.
- Show a different time scan by altering the “Scan Range” value box

However it is not recommended that you try to do too many of these things while the instrument is collecting data, rather please wait until the acquisition has finished.

## The “Scaling” window

	Min Value	Max Value
IC0	0	.0005
Far	0	.5
IC1	0	.001

This allows the minimum and maximums of the recorded beams to be altered. The collector is selected by double clicking the entry on the list, whence the entries are transferred to the two text boxes. In the example shown the Faraday entries have been selected for editing. The values may then be changed and the data stored via the “ENTER” key, as normal. The program stores the

plotting range, with other information (see below) and so once altered, the ranges will be permanently related to this data run.

The plots will be seen to alter as soon as the new values are entered.

If the ratio plot is enabled, the scaling window will permit the denominator beam to be altered, as well as change the plot intensity range of the scans. To change the denominator, click the “Select Denominator” button, and a set of option buttons will permit the reference beam to be changed.

	Min Value	Max Value
IC0/IC1	0	6
Far/IC1	280	320

## Batch Runs

### Preliminaries

The Nu Noble software suite can handle automatic sequences (including analysis), to enable automatic preparation system to be controlled. Two independent sequences can be run simultaneously, allowing, for example, a sample to be prepared in the manifold whilst its predecessor is being analysed in the mass spectrometer.

The program accepts a series of types of commands, which are concatenated to form a script-like series of instructions. The sequences are stored as readable text, and the sequences could therefore be generated using any standard text editor. We also provide a utility to generate these files, and the user is recommended to use this approach to minimise any errors, and to make use of the internal checking facilities the utility undertakes before the data is saved to file.

## Supported Commands

We will separate the current commands into two sets, the first consisting of simple direct commands, whilst the second group permit control of “intelligent” peripherals using a serial interface. The list is contained in the file “Supported Commands.DAT”, and you may wish to separate out those which you do not think you will use, and thus restrict the choice in the pull down list in the window used to create the run sequences (see below).

The current full contents of the file is shown here, where the commands and their structure can be seen:

```
"Switch On",1,"Name"
"Switch Off",1,"Name"
"Switch On Identified",2,"First Part", "Identifier"
"Switch Off Identified",2,"First Part", "Identifier"
"If Beam(I)<=X",3,"Axial Mass","Collector","Test Value(X)"
"If Beam(I)>=X",3,"Axial Mass","Collector","Test Value(X)"
"If B>=X",2,"Sensor(B)", "Test Value(X)"
"If B<=X",2,"Sensor(B)", "Test Value(X)"
"If B is True",1,"Input(B)"
"If B is False",1,"Input(B)"
"Delay",1,"Secs"
"Acquire",0
"Set Start Time",0
"Wait for OK to Start",0
"OK to Start",0
"Wait for OK to Proceed",1,"Analysis File"
"OK to Proceed",0
"Go To",1,"Step"
"Abort Batch",0
"Load Tuning",1,"File Name"
"Send String(1)",2,"Port", "String"
"Send String(2)",3,"Port", "String(a)", "String(b)"
"Send String(3)",4,"Port", "String(a)", "String(b)", "String(c)"
"If Reply is NOT(1)",3,"Port", "String", "Reply"
"If Reply is NOT(2)",4,"Port", "String(a)", "String(b)", "Reply"
"If Reply is NOT(3)",5,"Port", "String(a)", "String(b)", "String(c)", "Reply"
"If Reply is NOT(with Timeout)(1)",4,"Port", "String", "Reply", "Secs"
"If Reply is NOT(with Timeout)(2)",5,"Port", "String(a)", "String(b)", "Reply", "Secs"
"If Reply is (with Timeout)(3)",6,"Port", "String(a)", "String(b)", "String(c)", "Reply", "Secs"
"EOF"
```

As will be seen the format is the command name followed by the number of input parameters, which are required to be associated with the command. The next parameters in the string will be the title of the prompt, to identify the parameter in the input utility. For example, with the “Delay” command, one will need to specify the number of seconds for the delay, so the identifier has been called “*Secs*”.

We will now discuss the individual commands in detail.

## **The “Direct” commands:**

### **OK to Start, Wait for OK to start, OK to Proceed and Wait for OK to Proceed**

In order to have two sequences running concurrently, the program needs some way of knowing:

- a) when it is safe for the next sequence to start
- b) when it is safe to let the gas into the mass spectrometer.

Although there are many ways of achieving this, the way we have chosen relies on the use of the tokens, “OK to Start” and “OK to Proceed”. These are both set as the program starts, and if the sequences all start with the “Wait for OK to Start” command, the first will start, which then automatically sets the flag “OK to Start” to false, so preventing the second sequence from commencing. The “OK to Proceed” flag is used to control the entry of gas into the mass spectrometer. As the first sequence prepares its sample of gas, when it is ready to do the sequence of events to admit the sample for analysis, you place the “Wait for OK to Proceed” command. This first sequence will find it set to “true”, so continuing to admit the gas. However this will automatically set the flag to “false”, preventing the second sequence from interrupting the analysis. This first sequence can then commence the pump-out of any remaining gas in the preparation system, set the “OK to Start” back to “true” by use of the command, and start the data acquisition.

Similarly when the first sequence has finished its data analysis and pumped out the mass spectrometer, it can reset the “OK to Proceed” flag to tell the subsequent sequence that it can now let gas in.

### **Switch ON and Switch OFF**

This will control any digital output, and is hence normally used to open and close pneumatic valves. The name must correspond to a valid name used in the MS.DEF file, but is not case sensitive, and the underscore ( \_ ) character required in the MS.DEF file can be replaced with a space.

### **Switch ON Identified and Switch OFF Identified.**

If on has a very large manifold, one can simply refer to the ports with such names as Port 1, Port 2, Port 3 etc. To help with the input of these names in the batch files, we have provided the “*identified*” command, where the name of the digital output is split in to two parts. In use with the given example names, one would assign the first part parameter to “Port ” (don’t forget the space), and set up the batch file entry so that the user has merely to specify the port number (1,2, 3 etc) associated with each sample. This sort of entry will be discussed further below.

## **Delay**

This provides a simple delay, for an given number of seconds. This will obviously be used mainly after a valve is opened in order to let the gas equilibrate or a pump-out to be effective.

## **If Beam(I)<=X and If Beam(I)<=X**

These two commands allow the sequences to automatically check for over beam samples, to permit the gas slug to be split (using further sequence commands). This may be achieved by, for example, closing the manifold valve, pumping the mass spectrometer portion away, and then readmitting the manifold portion into the mass spectrometer. It could also be used to check that a sample is actually present, before continuing. The parameters to be input with these commands are the **AXIAL** mass when the required collector is monitoring the required peak, the identification of this detector, and the test value for checking.

Before jumping the magnet to the required mass, the program checks to see if that mass has been previously centred using that detector, and if so uses the correction factor found during the last centring. Otherwise the input value is used.

## **If B>=X and If B<=X**

This provides a simple decision process to be performed. For instance if you wish to ensure that the pump-out process has proceeded to its conclusion, you could envisage the following sequence of events:

- a) Open the ion pump valve
- b) Wait 30 seconds
- c) See if the ion pump pressure is greater (or equal to) a given value
- d) If it is then go back to (b)  
otherwise continue

This would involve the “If” statement. It will also be necessary for the program to know where the sequence of commands to be executed if the “If” statement is true actually ends. This is done by the use of the “End If” command. In practice, if the utility is used to create the sequence, the “End If” statement will automatically be created whenever the “If” is selected. (Similarly if the statement is deleted the program will automatically also remove the corresponding “End If”!)

In the statement, the A parameter is some analogue input address, which must be present in the MS.DEF file, whilst the B parameter is the required, numerical test value.

## **If B is True and If B is False**

Similar to the above, but checks the state of a named digital input. Useful if two systems communicate by toggling output lines.

## **Abort Batch**

The program will force the sequences to end. Useful, for example, if a peripheral device ceases to communicate (see below).

## **Set Start Time**

In the analysis of the recorded run data, all the beam intensities are referred to “time zero” (see Chapter 8). Normally this should be the time that the manifold valve was opened, and the sample let into the mass spectrometer. This command can be used to define exactly where in the sequence you wish this “zero time” to be placed. If the command is not invoked, the program will use the time the actual analysis routine is started (i.e. before such actions as peak centring etc), as is the case if manual analyses are undertaken.

## **Go To**

Often despised by “Pure” programmers, the “GO TO” command is very useful, and provided for use with the “If” conditional statements. The line number referred to is the target command line number.

## **Acquire**

This will start an analysis run, automatically loading the data acquisition form discussed in Chapter 8. The actual run file to be used will be defined in a manner to be discussed later. It is possible to have more than one “acquire” command in any set of sequence events, and each may reference a different file. This may be required, for example, if one of the beams is found to be larger than expected, and a more optimum analysis routine could then automatically be employed.

## **Load Tuning**

This is provided for the case when more than one element is expected to be analysed in a batch situation. If the source tuning parameters are to be changed before an analysis (for example if a Neon analysis follows one for Helium on a sample using a programmable cryo trap), one should load the new tuning file as early as possible in the relevant sequence. Again, only one tuning file may be associated with one sequence.

## **The Serial Commands:**

### **Preamble**

The program can talk to other serial devices using one of the ports on the system micro. A driver is provided to control the protocol of the conversation, and the general form of the driver code in the MS.DEF file is as follows:



```

port SIO5 is GenSerial
{
  opts
    Baud_Rate := 9600
    Parity := 4
    nStops := 7           //encodes one stop bit
    nBits := 8
    Rx_Term := 13         // Receive terminator is Chr$(13) CR
    Rx_IgnoreLF := 1     // will ignore any line feeds returned

  vars
    Tx_nBytes is Transmit_Byte_Nos2
    Rx_nBytes is Receive_Byte_Nos2
    TxBase is Transmit_Base2
    RxBase is Receive_Base2
    Status is Gen_Status2
}

```

Although you do not to know too much about the code here, it will be useful to understand some thing about the processes involved if you are to obtain the maximum benefit from this facility.

Here we have defined the port 5 of one of the serial cards (the program segment does not give us information as to which slot is being used here) for the general serial interface driver. From the section after the “vars” (for variables) statement, you will notice that all the names end with “2”, for example Transmit\_Byte\_Nos2. This defines that this will be identified as serial port number two in all calls by the sequence code. Up to eight serial ports can be set up in this way, numbered 0 to 8. One trusts that the meaning of options at the top of the example is obvious.

### **Send String(N)**

This is the simplest serial command. There are two parts to the command, the first being the port number (0 to 8 as discussed above), whilst the second part describes the string to be sent. This can be a concatenation of up to three parts (the maximum value of N thus being three), allowing the user to make up string such as “Go to position XX”. Here the first part (“Go to position ”) is the first of two parameters, and would probably be a common part for many samples, whilst the second part (“XX”) represents a spot number for a laser blast, and would probably be set up as data entry term in the program used to set up a batch run. NOTE: Whether the string is case sensitive will depend on the remote device. Please check.

### **If Reply is NOT(N)**

Similar to the above, except a reply is expected. The program will wait until the reply is received and continue to execute the next section if the actual reply exactly matches the input value, otherwise it will jump to the code after the corresponding “End If” statement.. This is normally used if the remote device returns a message such as “OK” when it has completed its operation, so that our program knows it is safe to continue. In practice this is the same as the following command, with a default timeout value of 5 seconds being imposed, to ensure that the program doesn’t “hang” if no reply is received. The command incorporates the negative, so that the action to follow if the reply is incorrect (or a timeout occurs) follows this statement immediately, up to the

“End If” line. Subsequent commands in the list then define the “normal” sequence to follow when the correct reply is received. The received reply has been made **non** case sensitive.

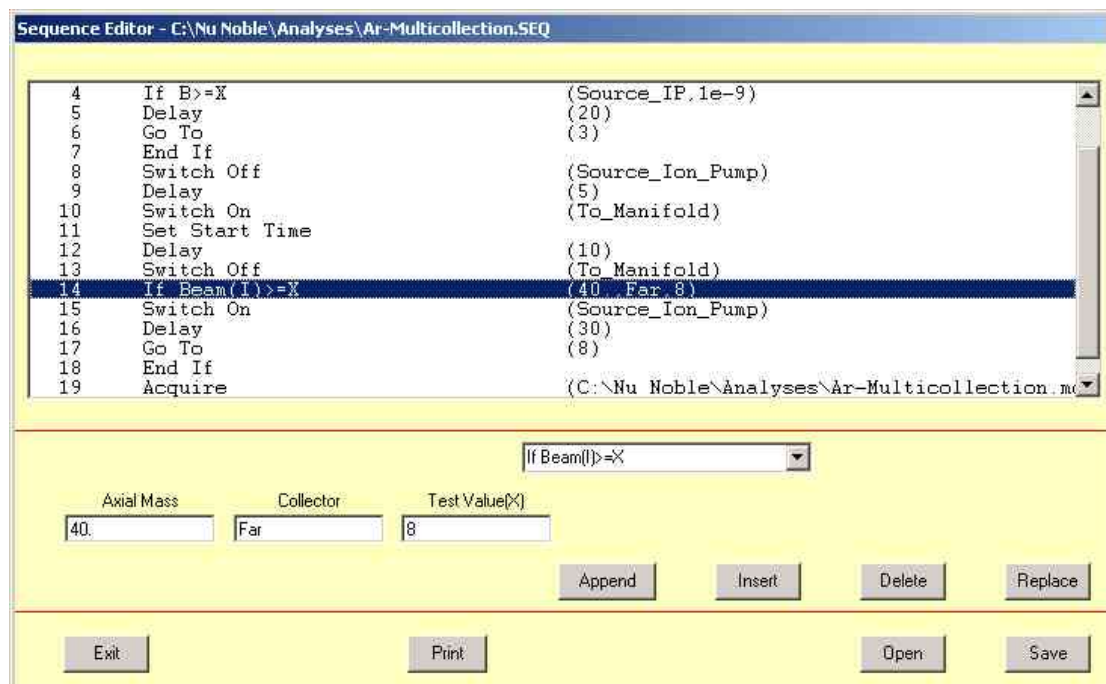
### If Reply is NOT(with Timeout)(N)

This performs the same function as the above, but permits other timeout values to be input. Although it may be used if you feel that a longer time is required, in practice it would be better to write the code in a loop, re-sending the question after a short delay and use a short (say 0.5 secs) timeout value. The reason for this is that the program will wait for the required time before continuing, and if a second sequence is running concurrently with the one containing this command, this sequence will also be suspended for this period.

### The Sequence Editor

This may be accessed from under the “Utilities” + “Create Auto Sequence” menu tree. Once loaded, either a previously created sequence can be opened, using the “Open” button in the bottom segment of the window (as was done with the example shown), or a new sequence created.

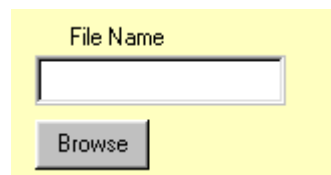
The various supported commands may be found in the pull down combo list in the central section of the window. In the example we have selected the “If Beam(I)>=X” Command, and the program has found that three input parameter are associated with this entry, and the associated titles are shown. This can be confirmed by reference to the print out of the “Supported Commands.DAT” file, given above.



To add a new command to the sequence, select the required command from the list. The selection process will automatically open the required number of text boxes with their titles. Previous entries in the boxes may be left, but since it is necessary to fill in

the required terms in all the boxes, this should cause no problems. Once all the boxes have been filled in, the entry may be added to the list via the “Append” button. As will be seen the entry in the list consists of the entry number (required for the “Go To” statement), the command itself, and the list of the input parameters, in parentheses. Entries may be edited by double clicking the required entry, or selecting the line and using the “Edit” button. After the edit has been done the “Replace” button (the edit button being renamed during the editing operation) will overwrite the old entry. Facility is also provided to insert entries into the list and to delete entries.

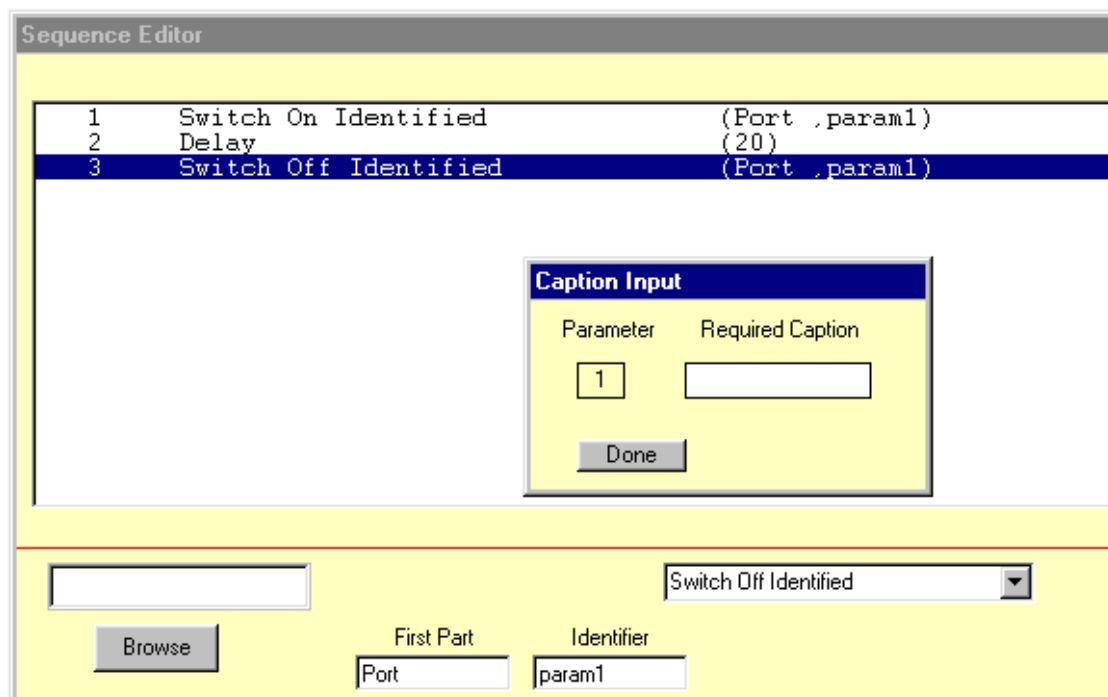
If a data acquisition is to be done during the sequence run, it is necessary to identify the associated analysis routine. This may most conveniently be found using the “Browse” button, which appears when this command is selected and which will allow the standard directory search facilities to be opened.



If the command “Load Tuning” is chosen, again the standard text boxes with titles is replaced with a new one, with its dedicated “Browse” button. This will allow the tuning file to be found using the normal dialogue windows.

The sequence can be saved to file, using the “Save” button. A number of checks are undertaken before the “Save To” dialogue box appears, to ensure that all the commands have valid data present. If an error is found, a message will be presented, identifying the problem, which must be fixed before the save process can occur. The files are saved with a .SEQ extension, although, as usual they are in CSV format and may be read (and edited, but this is not recommended) with any standard text editor. Once the new sequence is saved, a hard copy may be printed.

### The use of unassigned Parameters



If one is combining a series of similar analyses in a batch run, as for example if one is analysing a series of laser spots or samples from a series of different ports, it would be useful not to define the laser spot position or port address at this stage of creating the sequence, but rather add it later when the batch is compiled. This feature may be achieved by using the reserved word “ParamN” (where N is an integer between 1 and 6) as the relevant input in the text box. Thus, if we use the example referred to earlier with a number of ports whose valves are labelled Port N, we can use the “Switch On Identified” command and enter “Port ” for the first parameter, and (say) “Param1” for the second. When we come to the command to close the valve we then use the command “Switch off Identified”, with the same two parameters entries.

If two (or more) parameters are to be input in a single command, the program will check to see if they are adjacent, and if they are, automatically add a coma between them. This facility will allow a set of X and Y coodinates to be set up, as for example, in a command such as “SetPosition X,Y”, when each X,Y co-ordinate will correspond to a different sample in the batch.

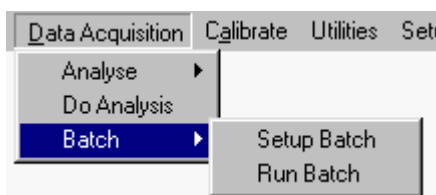
When the sequence is run in a batch the program will know that there is one parameter which must be input (assuming only one ParamN value is specified) whenever the sequence is loaded, and will prompt the user to type in its value during the entry process.

When you come to save the sequence file, the program will prompt you for the identifier for each parameter (one in the example shown) which you wish to identify the text box on batch data entry, as shown above. A possible caption for this example could be “Port number” or “Port ID”.

### Testing the Sequence

It is recommended that the sequence be run in isolation, before being combined into a batch of samples, to enable any errors in the logic to be determined. This is possible using the “Run Batch” menu, described in more detail below, when access to the .SEQ files may be achieved by use of the down arrow in the file type selection box of the “Run File Type” dialogue box. Only files which do not contain any ParamN variables may be run or tested in isolation, but obviously the sequence may be temporally set up with the parameter set to one of the possible values for this test to be undertaken.

### Setting up a Batch Run

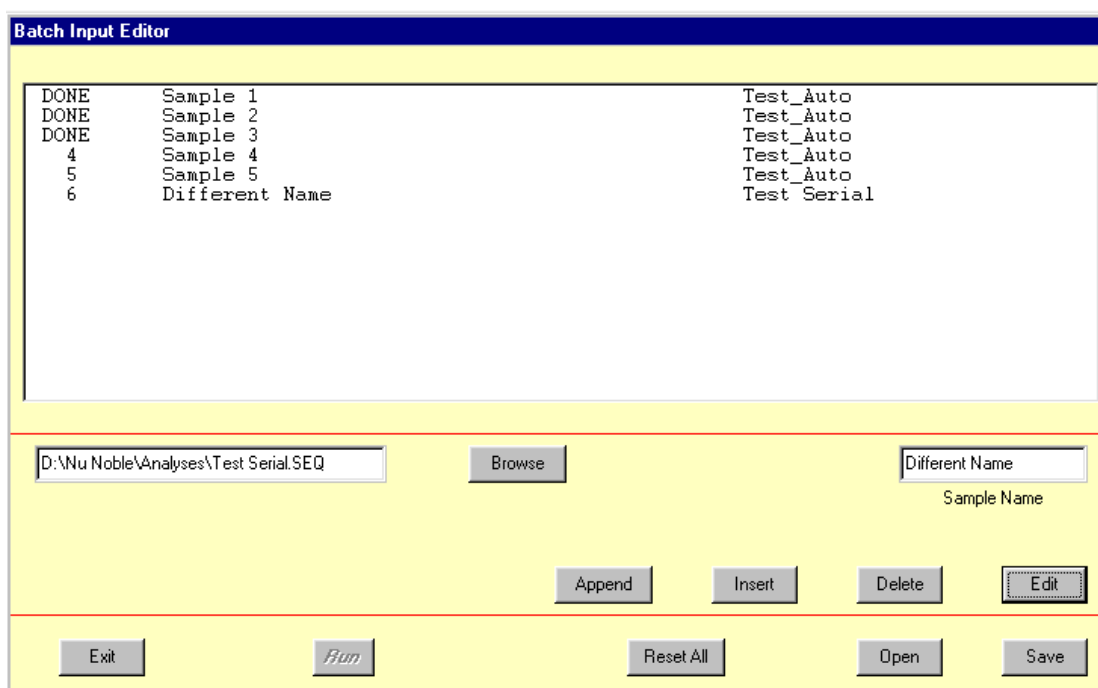


The batch run may be set up using the “Setup Batch” menu, as shown here. This will load a window, similar to the Sequence Editor format, in which a list of sequences may be defined, the combination of which will result in the definition of

the batch run. As above, previously created batches may be opened for editing on the form.

We show below on such file, which has been loaded for editing, since this illustrates how the program runs. After each new sequence is loaded for running in the batch, the sequence is labelled by marking it “DONE”, as for the first three sequences in the example shown. Thus, in this case, if you want to continue the batch after an interruption, the program will start from sequence four, as required. However the normal way the batches appear to be used is for a set to be created, and the same file used continually. To rerun the batch, it would be necessary to remove all the “DONE” statements from every sequence in the batch list. To enable this to be done with the minimum of effort, the “Reset All” button has been provided, which will automatically do this job.

In order to create the batch list, the list of existing sequences can be found by use of the “Browse” button. Each entry should be labelled with the required sample name at this stage, and this will be the name used during any data acquisition. As usual, new entries can be added using the “Append” button, whilst entries may be edited, deleted or inserted to change the run order.



If the sequence contains on of the ParamN input variables, the required number of text boxes will automatically be shown, together with their associated titles, to permit the values of these parameter to be input at this stage.

Once the batch list has been completed, it can be saved to file, and is saved with a .TXT extension. This is to enable other programs to easily alter the entries, if so required.

Once saved the batch can be run, either directly, using the (now enabled) “Run” button, or via the “Run Batch” menu option.

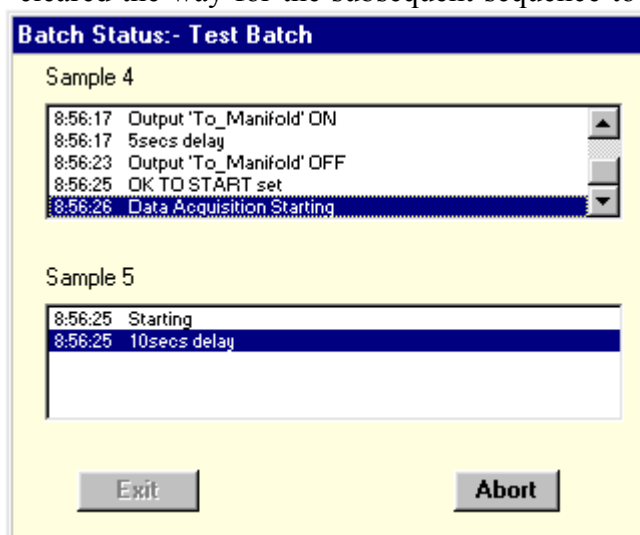
## The Run Batch Option

If selected, this will load a window from which the required batch file may be selected, and the run started via the “Start Run” Button. (See the top of chapter 8.) If the stop on error is unchecked, the program will ignore any detected errors. Since this could result in your samples being used, without a correct analysis being undertaken, it is obviously *not* recommended!

## The Batch Run

During the running of the batch, the window shown below is present, in which the present status of both current sequences is shown.

In the example shown here, the second sequence has been “held” up, by the first, and only permitted to commence once the “OK to Start” flag has been set, as discussed above. This can be confirmed by reference to the times given on the left of each line. The first sample is just about to start its data acquisition, as indicated, which has cleared the way for the subsequent sequence to let gas into the manifold. Obviously,



the first sequence can also stop the subsequent ones from using the mass spectrometer via the “OK to Proceed” Flag.

The batch may be aborted either by use of the “Abort” button, as shown here, or by the “Stop” button on the Analysis form, which will also stop any further sequences from running, not merely the present one. If the abort is invoked, it may take a few seconds for the “Exit” button to be enabled, especially if the sequence

is executing a delay when the button was depressed. This is because the abort will only stop at the end of the present operation, to ensure that the program is exited cleanly, and hence the outstanding delay must be completed before control is returned for manual operation.

## Interacting with other computers

Some users have preparation systems controlled separately, with another computer, and wish to use this as the “master” during batch runs. Although the interfacing is the responsibility of the user, the following comments may help.

There are two possible ways for the two computers to communicate, via a set of digital input and output lines or using a dedicated serial (RS232) line to the system micro.

The latter case is more complicated, and will not be discussed here, since the Nu Noble software suite cannot respond to unsolicited serial communication. The serial

communication protocol provided is discussed above, and the user will need to define a set of commands to be used in a set of sequences, to enable the communication to occur. The basic theory of the method, however, is similar to that discussed next.

The simplest method of ensuring the Noblesse software waits for the preparation system is to use a set of digital inputs and outputs as a signalling system. Thus one computer can set one of its outputs to tell the other it is ready for the next step to be carried out, and (preferably) the receiving computer can then acknowledge receipt of this signal by setting one of its own outputs, in reply. The sequence can have a simple polling routine written, whereby the Nu Noble software waits in a loop until the digital input state is set, before proceeding to the next step. The code used would look like:

```
If B is false           (where B is assigned to a digital input address)  
Delay for N secs  
Go to top of loop  
End if  
Continue here when B is set
```

It is also possible for the preparation computer to define which analysis sequence is run, if the batch file being used is overwritten before the digital state is changed. Since the batch editor will run the next sequence which does NOT have the "DONE" string set at its start, merely writing the file in the correct format, with new entries will enable the second computer to control the choice of sequences. The required format can easily be inferred by printing out some of the batch files (which are saved with a ABC.txt name for easy manipulation). Such high level communication is done using standard network file writing methods. However, if this level of control is used, please be aware that the batch control program is intelligent enough to know that if only one sequence (or two at the start of the batch) is left in the text file, once that one is read in, the batch will end. In this case it would not be necessary to read the file again. To "fool" the batch program to keep reading the text file, it may therefore be necessary to have dummy sequences following the one you are adding, rather than just append the required sequence to the end of the existing batch.

Since, by using such approaches, the degree of flexibility for sample analysis is almost endless, it may also be useful for the secondary computer to keep track of the data filenames for each analysis. To enable the secondary computer to define the identifier, rather than use the normal incrementing run number, the analysis routine will read the file "Batch\_Run\_Info.DAT" in the RESULT directory being used for the analysis (as defined in the "SETUP" window). If this exists and has been modified less than 60 minutes prior to the analysis being started, the string in this file will be used as the identifier, otherwise the normal DATA\_nnn.RUN name is employed.

## Chapter 8: Data Analysis

### Overview

This chapter describes data analysis, *i.e.* how we derive ‘useful’ quantities from the ‘raw’ data obtained during data acquisition. As far as the Noblesse software is concerned, the processes involved fall into two, distinct, categories:

- Subtraction of detector baselines, or calculation of isotope ratios. These processes are defined by the calculation run file (*NICE* file), which was created by the *NICE* software described in the previous chapter.
- Fitting a curve (or straight line, or a simple mean) to a quantity versus time, in order to estimate the value of that quantity at gas inlet time. This quantity can be a beam measurement, a baseline (zero), or even a ratio of two isotopes.

In the following, it will be helpful to remember that

**Beams** refers to directly measured quantities, either the ion beams themselves or the zeros (baselines), whilst

**Answers** refers to quantities calculated following the procedures defined in the *NICE* files.

The program can combine the above processes in two different ways. One method involves curve-fitting the beam data to get the beams at time zero, then applying the *NICE* procedures to the time-zero beams to calculate the answers. The second method involves computing the answers *for each cycle*, then fitting a straight line to these answers to derive the answer at time zero. The second method allows us to plot an isotope ratio versus time.

This chapter also describes the format of the raw data files.

### Configuring the data analysis

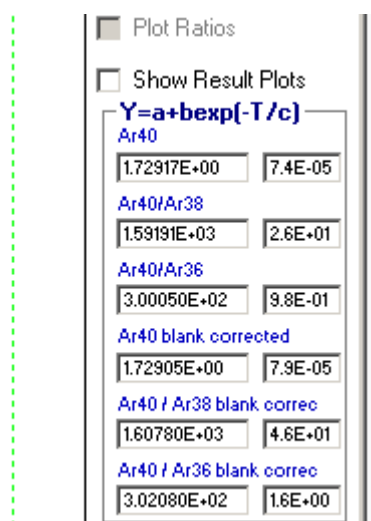
Part of the data analysis procedure has already been specified using *NICE* (described in the previous chapter). The remainder of the procedure is set via the **setup – preferences** menu.

### How to do the data analysis

Once the analysis ends, either by the required number of cycles (as defined by the analysis set-up window) being completed, or the “Stop” button being pressed, the “Calculate” button becomes enabled, permitting data reduction to be undertaken.



At the start of the recorded data, a vertical dotted red line will be seen. (Move the plot area using the slider bar if the lower time slice of zero is not visible.) Further, the end of the data set is marked with a purple dotted line. These two markers define the range of data, which will be used for the analysis routine. These markers may be moved by dragging with the right hand mouse button depressed, if so desired. These terminators will tend to “jump” to the nearest completed cycle marker when released, so as to ensure that the maximum data set is analysed. It is not expected that you will use this facility unless, for example, something happens half way during a run, when the latter part of the data set may need to be discarded.

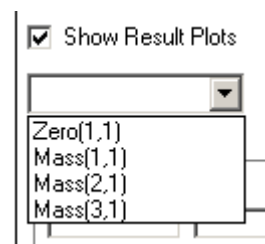


Having pressed the calculate button, the final results (answers) are shown in a frame on the right of the plot area. If a large number of results are required, there may not be room to show all of the required answers, but if the “Print Results” button is clicked, the complete analysis result set is sent to the printer, together with the fits of the individual beams.

The form of the fit is also shown, as seen here. If you wish to change the analysis method (e.g. from exponential to linear fit) just select the required method from the Set-up window and reanalyse the data using the “Calculate” button.

If the data set is calculated, the program assumes that the measurement was valid, and saves all beam and run time data to file. The data is saved in CSV format in the given “Result” directory (as defined on the Set-up window form) under the name Date\_NNN.run, where NNN is an automatically incremented integer. The program will not resave this data set if a re-calculation is invoked. The format of the saved data file is discussed fully below. The full identification of the filename is given at the top of the hardcopy of the data analysis printout.

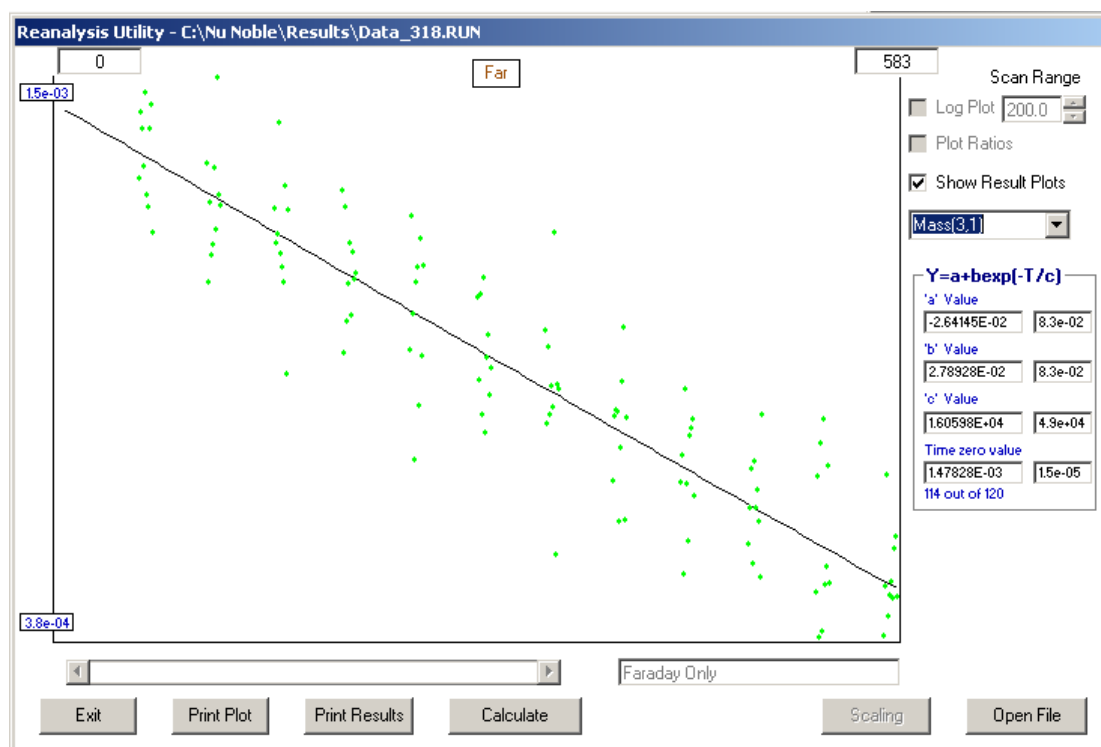
It is possible to view the fits to the beam intensities by checking the “Show Result Plots” box. As will be seen, a drop down list can be accessed, which refers to all the beams (and zeros) referenced by the analysis program. This limitation is done to minimise the number of traces to those with required information. If you wish to monitor a beam during the analysis, but the intensity of this signal is not used in the analysis, simply refer to it using a dummy assignment in the program to ensure that data is shown.



Once a beam is selected the program will show the data, best fit and numerical values for the constants. The result frame also gives the time zero calculated intensity, together with its standard error. The number of data points used in the fit (out of the total available) is also recorded.

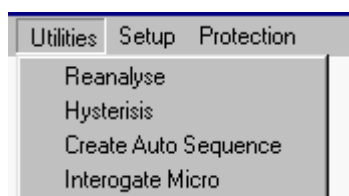
As mentioned above, this data is also printed out when the “Print Results” button is pressed. However, sometimes a hard copy of the plot itself is also required. This may be obtained using the other “Print Plot” button.

As may be seen in the example shown here, the data “groups” according to the data acquisition slots. Also the data does not start at time zero, indicating the time spent in peak-centring this particular run.



If the signal is very small, as in ion multiplier zero measurements, the plots may appear odd, in that the data will group together in discrete values. An extreme example of this would be a data set with either zero counts or 1 count only. The program traps this example for the exponential decay to stop errors occurring.

## Data Reanalysis



Previously stored data sets may be re-analysed using the above program, via the “Utilities” + “Reanalyse” menu tree, as shown here. The usual “Open” dialogue box will appear, and once the required file has been chosen, the program will plot the data as previously. The reader will have noticed that the above plot was obtained by this

manner, since the window header identifies the file referenced. It is possible to have any number of reanalysis forms opened at one time (e.g. for comparing peak traces etc), and to reanalyse one form whilst data is being collected. However, since the program is running under a Microsoft operating system, this is not recommended!

## How the data analysis is done

As mentioned above, two methods of calculating the required answers from the observed beams are provided; in both cases the calculated beams are sent to the Nu Instruments Calculation Editor (NICE) package to evaluate the required answers. In the first method, the values of the beams (including any zero measurements) are extrapolated to time zero, and these “Time Zero” beams are used to obtain the required results. The second method calculated the values of all the beams (and zeros) at the mid point time of each measurement cycle, and uses these data, with their associated errors, to calculate a set of answers for each cycle. These answers are then combined in either a linear time decay to obtain the “Time Zero” results, or simply averaged. The method used is defined in the “Setup” + “Preferences” menu option.

### Method 1: Extrapolate beams to time zero; then compute answers

*This method is used if the ‘Ratios Extrapolation’ box is **not** ticked.*

Firstly, the program fits curves to all the beam data sets (see **Curve Fitting**, below), so as to identify any measurements which should be rejected due to noise spike etc. Often you will have the program automatically select whether to use an exponential decay, straight line or simple mean, but you can force it to choose a particular type of fit if you wish. Each beam is fitted, the standard deviation of the fit calculated, and any data points lying outside the required n SD acceptance window, are marked so as not to be used in further calculations. (The value of n can be set in the “Setup” + “Preferences” menu option.) Both peaks (actual ion beams) and zeros (baselines) are fitted.

Next, the time zero values of both beams and zeros are computed.

Finally, the answers are computed by sending these results to the *NICE* package.

When you print the results, you will get both the parameters of the fitted curve, and the answers calculated by *NICE*.

### Method 2: Plot answers versus time; extrapolate answer to time zero

*This method is used if the ‘Ratios Extrapolation’ box is **is** ticked.*

Firstly, all the beam data sets are analysed using the “*Automatic Analysis*” procedure (see **Curve Fitting**, below), so as to identify any measurements which should be rejected due to noise spike etc. Each beam is fitted to the decay curve identified as appropriate, the standard deviation of the fit calculated, and any data points lying outside the required n SD acceptance window, are marked so as not to be used in further calculations. (The value of n can be set in the “Setup” + “Preferences” menu option.) Both peaks (actual ion beams) and zeros (baselines) are fitted.

## Obtaining answers at the midpoint of each cycle

In general, plotting isotope ratios versus time is not straightforward. This is because if any peak switching is used, the two isotopes of interest may not be measured simultaneously. Also, the baselines (zeros) cannot have been measured simultaneously with the beams and if there is any time variation with the baseline, this too must be considered. The program has been written to handle this general case.

Suppose that we want to plot  $(A-Z_1)/(B-Z_2)$  against time. We must evaluate  $A$ ,  $B$ ,  $Z_1$  and  $Z_2$  at the same times – the program does this at the mid point of each cycle.

To evaluate, say,  $A$  at the midpoint of, say, cycle 5, consider a plot, against time, of the deviations of the individual data points from the global fit. (The global fit will typically be an exponential decay/increase – what we are doing here is removing any trend and making the data array horizontal). The value of  $A$  at the mid point of cycle 5 is taken as [global fit value at mid point of cycle 5] + [average of deviations for cycles 4, 5, and 6]. For the first and last cycles, we take the average of deviations from just two cycles, rather than three.

This procedure is repeated for  $B$ ,  $Z_1$  and  $Z_2$ . Next, these mid-cycle-time *beam* values are fed to the NICE procedure to evaluate the mid-cycle-time *answers*. NICE also computes an error in the answer; in this instance this would be derived from the errors in  $A$ ,  $B$ ,  $Z_1$  and  $Z_2$ . Note that if there are  $n$  measurements per cycle, then  $A$  at the cycle mid point is derived by taking the mean of  $3n$  measurements, but the error in  $A$  not the error in the mean of these  $3n$  measurements, but rather  $\sqrt{3}$  times the error in the mean, to allow for the fact that each point is used three times. (For the first and last cycles, read  $2n$  and  $\sqrt{2}$  instead). The errors in  $B$ ,  $Z_1$  and  $Z_2$  are calculated in the same way.

## Plotting answers (e.g. ratios) versus time

These cycle-mid-time answers and their errors can now be plotted. The program will either fit a straight line to, or take the mean of, these answers. Exponential fitting is not available when extrapolating answers to time zero. The program computes the error in the intercept by two methods — from the error bars on the points, and from the weighted scatter of the points about the fit — and takes the larger of the two. (In other words, if we were taking the mean and the error bars were all the same, we would compare [error bar]/ $\sqrt{n}$  and [standard deviation of points]/ $\sqrt{n}$ , and take the larger of the two).

In this plot, the error bars should give a good indication of the uncertainty in an individual mid-cycle-time answer if we consider the points from that one cycle; and these are used to obtain the error in the time-zero intercept. But the actual plotted points were obtained using a running average (apart from the first and last scans, scan  $n$  is also averaged with scans  $n-1$  and  $n+1$ ). This is the reason why the rms scatter of the points is less than the size of the error bar, or – put another way – why almost all the error bars cross the fitted line.

The example thus far considered is a very simple one, where we just require the ratio of  $A/B$ , after baseline subtraction. In some cases, you may even wish to omit the baseline subtraction. A few other cases deserve special comment:

## **Blank subtractions**

As you will be aware, having read the part of this manual concerning the NICE software package (!) it is simple to pass results from one analysis to a subsequent one. This would then enable the blank to be measured, and then employed in the next sample measurement. However, it will be appreciated that what has been saved is the “Time Zero” blank value (with the error in its determination); the actual decay form is not stored. Thus to use these data with the method of determining answers every cycle is not ideal and is not recommended.

## **Errors in previously determined parameters**

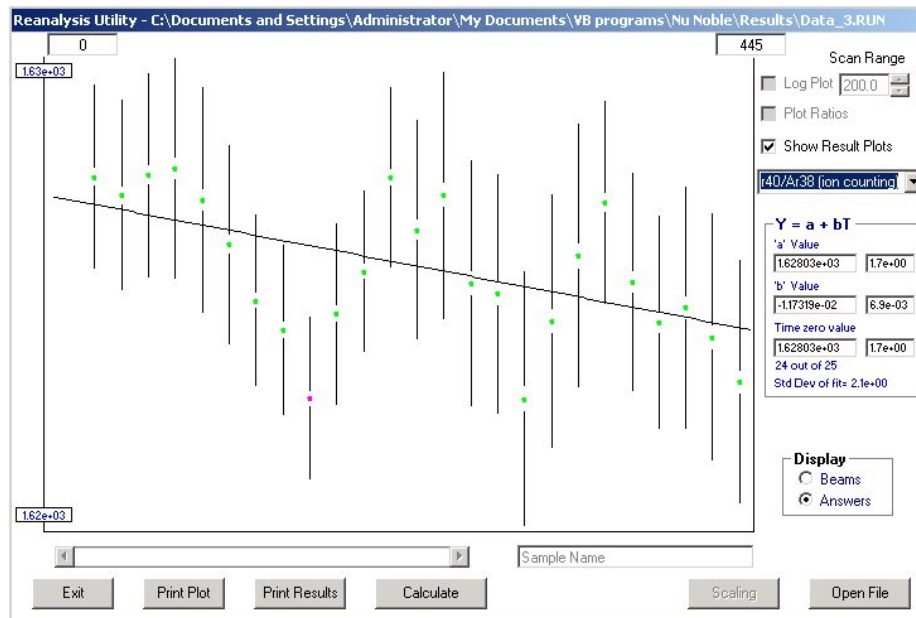
If data are imported from a previous analysis, the error in that data set is fully allowed for in the final computation. If the input data set is poorly defined (*e.g.* you measure a multiplier gain, but don’t do it very well), the final result set will have similarly large (and probably unexpected) errors. In such a case, the original “constants” can be redetermined, and the sample data set reanalysed as discussed below, when the new constants will be picked up automatically. It should not be necessary to rerun the sample.

## **Don’t plot ‘answers versus time’ to get an amount at time zero**

Finally, beware of using this method 2 to get an amount at time zero. It may be better to use method 1 instead. The reason is simply, that although you can define an answer as  $(A-Z_1)$ , in using method 2, the program will calculate the answer at cycle mid-time (which is an unnecessary complication) and then usually extrapolate to time zero using a *linear* fit. Since decay curves are frequently exponential, this last step may be completely inappropriate. Nevertheless, the beam data are available on the printout, and these *may* use an exponential fit.

## **General Comment on Curve Fitting**

Fitting data using different models (*e.g.* linear or averaging) can yield similar results, but one method may give a lower error. It does not necessarily mean the model is better, since, for example, averaging has a lower number of degrees of freedom than a linear fit. You must check the data, and not just rely on the numerical results. We show below what you may be missing (obtained from an instrument early on in test when still not clean)!



## Curve Fitting

Depending on what was set in the **setup-preferences** menu, the software will fit an exponential, a straight line, or a mean to the data. It can choose between these fitting methods automatically if you wish.

### Exponential decay

This is probably the standard decay method to be used. The set of beam intensities is fitted to the equation:

$$\text{Observed Intensity} = A + B \exp(-\text{Time} / C)$$

where the time used is the time from the start of the data run, not from the time of the first measured beam. The fit uses a non-linear algorithm, and determines the values of the constants A,B and C, as well as their errors. The data is then tested to see if any points lie outside the two-sigma band of the measurement set. Those outside are rejected and the data set reanalysed to give the final values for the constants.

The fit is based on the Levenberg-Marquardt method, with the program code based on that given in the book *Numerical Recipes*, Cambridge University Press 1986, ISBN 0-521-30811-9. From the calculated fit of the data, the standard deviation of the analysis is calculated and this data then used to obtain the standard errors of each parameter.

The program then determines the value at time zero for the beam, and the error of its measurement. These are the values used to determine the required answers. The program calculates these data for all beams referred to in the calculation program, and then uses these calculated time zero intensities to give the required results, together with their errors by passing the values to the compiled *NICE* calculation program. The reported error is the one standard deviation value.

### **Linear decay**

The data is fitted to the equation:

$$\text{Observed Intensity} = A + (B \times \text{Time})$$

where the same comments as to the time zero point apply as above. The data undergoes two-sigma rejection testing and is then the two constants of the fit re-determined. Again the time zero values for all the beams referred to in the analysis program is calculated, together with their errors, and these data are then used by the calculation program to obtain the required results.

### **No time decay**

Here the data from the analysis is averaged, to obtain a single value for each beam and zero measurement. The data is sent through a noise filter to remove any measurements that lie outside the two-sigma band for each particular data set. These data are then analysed by the *NICE* calculation program to give a single set of required results.

Since the beams obviously decay during the run, it is unlikely that this non-decay approach will be used, apart from with ion counting zero measurements.

### **Automatic Analysis**

This approach uses a statistical test to determine which of the three decay possibilities, discussed above, should be used. Each beam set is treated independently, so it is quite possible that zero data will be analysed using a linear fit (or even no time decay) whilst the beams themselves are fitted to the exponential decay. To decide which fit to use, the program initially uses a linear decay, and calculates the range of the fitted line between the start and end of the analysis time region. This is then compared to the standard deviation (SD) of the fit. If the range is less than one SD, the non-time decay is used. If it is greater, then it refits the data to a quadratic. If the SD of the new fit is lower than 0.9 of the previous (linear) value, we assume that the decay is curved and an exponential fit will be invoked, otherwise a linear fit is assumed. However since there may be a weak isotope present in the analysis which would be analysed using a linear fit, whilst the stronger members would be fitted with an exponential, we have added a extra check, such that if one of the beams is definitely exponential, any linear candidates are also force to be analysed exponentially. Since the calculated precision of the time zero extrapolation is directly related to the number of degrees of freedom (e.g. exponential has three), this approach minimises the reported errors. Obviously there is no sense in using a fit with more degrees of freedom than is statistically significant, and if this approach is not selected, the user may have to manually analyse the data using the different decay approaches, and combine the time zero data off-line.

## **Format of the saved data file**

The data is stored in ASCII to the data file, and so can be read by any text editor, and easily imported into spreadsheets for off-line analysis, if so required.

The form of the file is as follows:

1. The version number
2. Data defining the run. Output with a string to identify each parameter.
3. The start time of the cycles, in a list.
4. A list giving the active detectors, the plot ranges for the plots (both normal and ratio plots) and the identifier of the denominator beam for the ratio plot.
5. Information defining how many of the beams were centred at the start of the analysis. If centring occurred, the success, or otherwise is noted in each case. (It is obviously important to know of an error in the run!)
6. The setting of the instrument at the run end. If differing quad values were employed during the analysis, due to changing the zoom settings, this information will have to be recovered from the .MDF analysis set up file.
7. The full name of the analysis file
8. The name given to the sample
9. A spare line

The recorded beam data then follows. We show a (small) sample to illustrate how to decode the information:

```
1.453786E-04,6223997,1.933494E-03,15,0
5.287237E-07,4.543092E-02,2.105193E-03,16,0
5.607676E-07,-4.156246E-03,2.09804E-03,17,102
```

This data set comes from an analysis undertaken using three detectors. Only beams from the detectors enabled in the +original run set-up file are measured and recorded. The beam intensities are recorded even if the calculation program does not use them. The first three numbers of each line therefore correspond to the observed intensities on these three active detectors. The penultimate number gives the time of the measurement (starting from zero for the first measurement whilst the offset time is given in the data set at the top of the file), the first line being recorded fifteen seconds into the run. The final number is a code to identify what is being measured. The code is:

0	A wait state
1	The first zero measurement step
2	The second zero measurement step
101	The first analysis measurement step
102	The second analysis measurement step
103	The third analysis measurement step

etc, etc.



## Chapter 9: General Instrument Service

### Changing a filament

Hopefully this should be the only time that the user will have to break the instrument vacuum and do general maintenance inside the vacuum envelope.

The replacement filaments are available from Nu Instruments, under the stock number 1030015, and may be purchased as required. One spare is supplied with the instrument. Since it is likely that the filament has been sitting around for some quite some period before use, it is recommended that it be thoroughly cleaned before use. We recommend a good ultrasonic clean in a mild solution of pH neutral detergent such as “Decon Neutracon” for at least 15 minutes, followed by at least 4 ultrasonic washes in distilled water. Please do not use standard “Decon” to clean these items, since, although this detergent is ideal for cleaning stainless steel, it is too basic to use where differing metals are present.

It is also recommended at this time that a set of tools is thoroughly cleaned for working on the source region. You will require at least a good small electrical screwdriver, capable of working on small bolts (M1.6 and M2), and a good pair of tweezers for picking up both the filament and small bolts, which may be removed from the source assembly. The filament may be changed with the source in place, as discussed below, or the source may be removed from the flange, and the filament changed with the source on a (clean!) bench. In the latter case an Allen key is required to undo the (vented) M4 socket bolts which attach the source baseplate to the flange (3mm across flats). You should also prepare a new gasket for the source housing seal. We recommend a silver plated gasket for this purpose, and they can be obtained from Nu Instruments (part number 1004517) or from most vacuum catalogue suppliers. (The flange has an outside diameter of 114mm.) We recommend that the gasket be cleaned in the same manner as the filament (i.e. with a neutral detergent followed by distilled water, rather than the more conventional “wipe” with acetone. Obviously it is essential that you use gloves at all times when dealing with all internal parts of the mass spectrometer. Finally you will need a 13mm AF spanner to undo the flange bolts. All tools were supplied as part of the spares kit with the mass spectrometer.

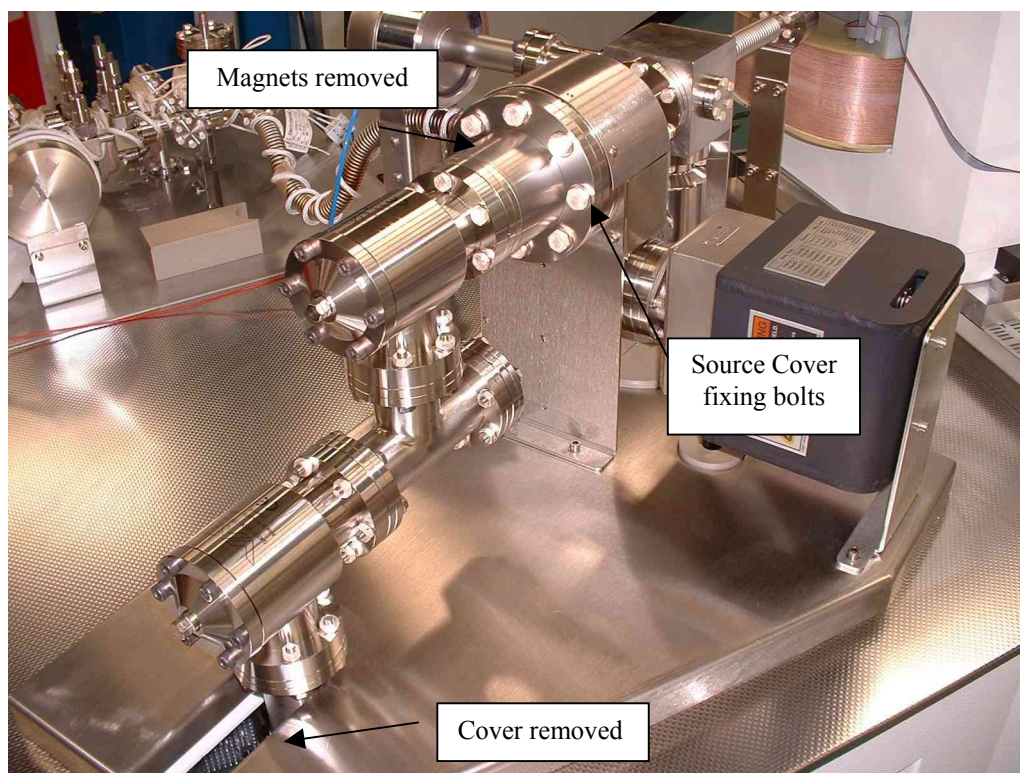
The instrument should now be prepared for venting. Firstly switch off the source supply, the detectors, high voltage units and “1kV” units at the Mains Distribution Unit at the bottom of the central rack of the instrument bench. This may be done using the middle of the three rows of mains switches on the left hand side of the front of the unit. The magnet may be left on during this whole process, but if you prefer it may be switched off using the relevant switch on the mains distribution unit.

Leave the instrument for at least an hour in this state since, not only will the source be hot, but the multipliers also run quite warm during normal operation, and all these items take a surprisingly long time to cool under high vacuum conditions.

Although the instrument can be vented and evacuated via the getter assembly and the two valves surrounding it, this approach will fill the whole getter area with air for a long period, requiring it to be reactivated after the filament change, and subsequent bake. If at all possible therefore, we recommend the venting and initial instrument

evacuation to be done via the user's manifold. However for the purpose of this discussion we will assume that this is not possible, and that the venting is to be done via the getter region. We will discuss this approach, the user will hopefully be able to modify this approach if the manifold is used.

Once the system has been switched off for at least an hour, it is time to prepare to vent the mass spectrometer. Firstly the two ion pumps should be isolated. The collector one is done by closing the manual valve, whilst the source valve is shut via the icon on the vacuum schematic on the computer screen. It is safer to switch off both ion pumps at this stage, this being achieved by momentarily depressing the "HV on/off" button on the lower left hand side of each ion pump control unit. The display will then change to either "OFF", if the pump pressure is being displayed, or "0" if other displays (current or voltage) is selected. The vacuum schematic on the computer screen will display "10e-10mbar", whilst the "Digital Control" window will show the two ion pump status boxes unticked. You should now remove the source magnets and yoke, by undoing the two thumbscrews and place the yoke on a suitable surface away from the mass spectrometer bench. As you remove the yoke from the housing, a slight resistance will be felt, due to the attraction of the magnets to the field concentrators in the source. This is quite normal, and you should not worry about it. Also remove the high temperature "plastic" cover from the slot at the end of the source bench region, so as to permit the whole assembly to be moved back when required. The area around the source should now look like the photo below.



The instrument is now ready for venting. Assuming this is to be done via the getter, isolate the instrument rotary using the icon on the vacuum schematic on the computer screen. If the turbo pump is also being used to evacuate the user's preparation system, ensure that any valves to the pump are now firmly closed. Switch off the turbo pump using the switch labelled "Turbo" on the mains distribution unit front panel (top row), and wait for 10 minutes for the turbo to slow to a reasonable safe speed. Open both

the manual valves around the getter. The instrument may now be vented, and this may be achieved by slowly opening the vent port situated half way up the turbo pump body.

The 8 M8 silver-plated bolts fixing the source housing may now be removed. This will then permit the whole source housing and getter pump assembly to be moved backwards on the (hidden) slide assembly to allow access to the source. This is shown below:



The observant reader will note that we have placed the M8 bolts upwards on the prep system sub-bench, ready for reuse, but well out of the way to allow easy access to the source. Remove the old gasket at this time.



The filament will be seen to be at the top of the source block and can be easily removed. It is connected to the feedthroughs using solid silver wire and the connections made between the wire and the filament legs via small copper alloy connecting blocks. Firstly undo the bolts in these connecting blocks closest to the filament and remove the connector from the filament leg. The filament may then be removed by undoing the small M1.6 bolt through the centre of the filament ceramic body. The replacement filament may now be inserted in the source. You will find that it locates “automatically” via the small step on the ceramic block. Reconnect the two wire connecting blocks.

If you wish to remove the complete source assembly from the flange, firstly release all the connecting blocks from the source flange feedthroughs. If they are tight, remove the wire from the connector rather than trying to force the connector off the feedthrough central conductor. The source assembly may then be removed from the flange by releasing the three M4 socket head bolts, and taking the whole source backwards. The filament may then be removed, and replaced, as outlined above, but you may also wish to check the alignment before reassembling the assembly on the flange. To do this first remove the two field concentrator slugs, by removing their respective fixing bolts. Note that these are quite long, and fit into recesses in the slugs, to firmly fix and align them. Then remove one of the two bolts fixing the trap to the block (on the other side of the block from the filament), and loosen the second. The trap may then be swung out of the way to permit a direct line of sight view through the block to the filament coil. You may find use of a torch helps illuminate the coil. When you are satisfied with the alignment, reassemble and replace the source onto the flange. Reconnect all the wires.

Place the new gasket over the source and carefully push back the housing to cover it. You may find it simpler to place in two of the lower bolts through the cover into the source flange, and rest the gasket on them, to hold it in place as the cover is brought up to enclose the assembly. At this stage we would recommend that you check that there are no shorts from the internal wires, or block, to other wires or the outer housing. This may be done by removing the connectors from the back of the filament unit and “bussing” out the central pins to ground and other connectors. Alternatively the top part of the source flange wire cover can be removed to permit access to the outside of the source feedthroughs. Even if this approach is used, it is still recommended that the connectors at the rear of source unit be removed, since otherwise false reading will occur due to the circuitry in that supply. Since there is only the minimal amount of spare room in the housing, accidental shorts to the outer housing can be quite common, and it is best to confirm there are none at this stage. The only “short” which should be recorded is between the two connectors feeding the filament wire itself! Once this has been done, plug back in the filament unit connectors, before you forget.

Once you are convinced no shorts exist, tighten up the source housing using all of the eight bolts. Ensure that the flanges are face to face, or leaks may develop after the baking. Close the vent port on the turbo, and open valve above the instrument rotary using the icon on the vacuum schematic on the computer screen. Switch on the turbo pump supply, via the mains switch on the mains distribution front panel. Monitor the turbo speed on the vacuum schematic, and it should rise to 100% smoothly over about 5 minutes. Leave the instrument pumping on the turbo for at least one hour, after

which the ion pumps may be opened. As long as the filament change has proceeded reasonably smoothly, and the instrument hasn't been vented for too long a period, it should be possible to start the ion pumps at this time. If they do not start smoothly and go virtually immediately into the  $10^{-6}$  bar range (within about 5 minutes), switch off the ion pump controller, and wait a further hour before trying to start then again. The instrument is now ready for baking.

### **Changing the scrolls in the backing pump**

If the pump is kept on at all times, this should be done at least once a year. Assuming the standard pump has been supplied with the instrument (Edwards' XDS5 pump) the spare seal kit can be purchased either from Nu Instruments (part number 1010113) or directly from Edwards (their part number is A726-01-805).

The kit contains full instructions on how to perform the seal replacement itself, we will only therefore discuss how to prepare the pump for this simple operation.

Firstly ensure that all valves to the turbo pump line are firmly closed. This includes the source getter isolation valve (closed during normal instrument operation), but also any valves to the user's preparation system. In practice we shall not be venting the turbo, but closing all valves is good practice in case of any problems. Close the isolation valve above the scroll pump via the icon on the vacuum schematic on the computer screen. Now switch off the turbo and scroll pumps using the mains switch on the front panel of the Mains Distribution Unit situated at the bottom of the central bay of the instrument. The switches are on the top of the three rows to the left of the unit and labelled "Turbo" and "Rotary". Unplug the IEC mains lead from the pump. The pump may now be removed from the instrument bench by disconnecting the isolation valve immediately above the scroll pump. This then ensures that the backing line and turbo region remains evacuated during the service. Since the pump is quite heavy, it is recommended that two people lift it out from under the bench. Follow all recommended procedures during this process.

Replace the scroll tips as per the instructions provided in the packet.

Once done, replace the pump in the instrument bench, re-connect the mains lead and refit the isolation valve. Switch on the rotary via the Mains Distribution Unit switch, and leave it pumping for about 10 minutes. After this time the isolation valve may be opened (using the icon on the computer screen) and the turbo switched on. Monitor the speed until it achieves 100%, but leave to stabilise for about 30 minutes before using the turbo to pump out the instrument (shouldn't be necessary in normal operation) or prep system.

### **Re-activating the Source Getter**

Although we will only discuss the source getter, the discussion is also relevant for any getter pumps fitted to the user's preparation system.

Firstly, please be aware that during the re-activation process, the getter itself can be heated up to 800°C, and the housing and flanges will therefore rise to over 100°C.

The connector and plug are both designed to work at this elevated temperature, but your fingers are not!

To activate the getter the following protocol is recommended:

Firstly isolate the getter volume from the mass spectrometer, and open the valve to the turbo pump. This will ensure that any dirt or vapours produced during the heating do not pollute the instrument, but will be pumped away via the turbo. If not already connected, connect the BNC connector to the socket in the Getter flange. Ensure that the large control knob on front left of the Getter supply is turned fully anti-clockwise (i.e. off) before switching the unit on, which may be done using the large green rocker switch on the unit front panel.

Once on, slowly raise the current using the control knob, until the required 1.15amp value is reached. The current being drawn is shown on the red LED display to the right of the unit's front panel. This process should take at least 30 seconds. If you raise the power too quickly, the getter heater may destroy itself, this being especially a problem on the first activation. Leave the unit at this full power setting for the required amount of time, normally about 30 minutes, and then slowly wind down the control knob and switch off the unit.

Once completed, leave the getter pumping for at least a further 30 minutes. The reason for this is twofold. Firstly since the element is suspended in vacuum, there is no means for the heat to be lost, apart from (slow) radiation cooling. Thus the element is still in a position to pollute the mass spectrometer for a greater period than may be at first apparent. The second reason is that during the heating process, the seat of the valve to the turbo would have experienced quite a large amount of direct line of sight radiative heating. This must also be allowed to full cool before the valve is closed.

Once enough time has elapsed for the system to cool, the valve to the mass spectrometer can be slowly opened, and an eye kept on the ion pump pressures to ensure that they do not rise during this process (thus ensuring that no gas is still being emitted by the hot surfaces). Once this valve is fully open, the turbo isolation valve may be closed.

### **Baking the instrument**

The instrument has been carefully designed to ensure that the bakeout process is as simple as possible. As such, the source leads do not need to be disconnected, and all but one temperature sensor is already in place.

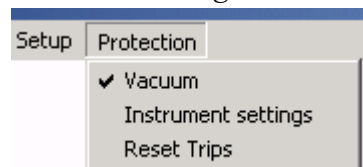
To carry out a bakeout the following procedure is recommended.

Firstly switch off the detectors, high voltage units and "1kV" units at the mains distribution unit at the bottom of the central rack of the instrument bench. This may be done using the middle of the three rows of mains switches to the left on the front of the unit. The magnet may be left on during this whole process, but if you prefer it may also be switched off using the relevant switch on the Mains Distribution Unit. Leave the source supply on, since we shall keep source region warm using the filament during the bake. Remove the quadruple lens array leads by undoing the connector at

the quad feedthrough flange, and lay them out of the way over the bench side, away from the instrument. Similarly unplug all the wires to the collector preamplifier, and similarly lay all the wires out of the way over the bench sides. Similarly, if the source getter lead is connected to the getter, unplug it and leave it lying out of the way.

Ensure that the turbo and rotary pump are both on!

From the menu bar of the Noblesse software, un-tick the "Instrument Settings" menu option. This will allow you to change the "Max Power" filament current setting on the "Deflectors and HT supplies control" window. Before you change it, ensure that you make a note of the original setting. Set the value to 3 watts, and check that the filament control unit display now shows this value (it may not be totally stable at 3 watts in this mode of operation, but this should not be a problem).



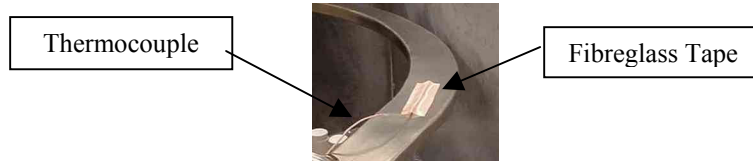
Remove the collector preamplifier, by carefully undoing the four mounting bolts. Try to remove this unit squarely, by supporting it as the bolts are released, so as not to place any undue strain on either the spring-pin connectors or the ceramic feedthroughs. Having checked that the maximum power to the filament is now limited to the safe value of 3 watts, you should now remove the source magnets and yoke, by undoing the two thumbscrews and place the yoke on a suitable surface away from the mass spectrometer bench. As you remove the yoke from the housing, a slight resistance will be felt, due to the attraction of the magnets to the field concentrators in the source. This is quite normal, and you should not worry about it. If you have just replaced the filament with a new one, as described at the start of this chapter, replace the high temperature plastic slot cover that was removed to allow the source housing and getter assembly to be slid back.

Ensure that all valves on the mass spectrometer are fully open. This includes the automatic valve to the source ion pump, which should be opened via the icon on the vacuum schematic on the computer screen, and any valves to the user's preparation system. We have found it best (and safest) to bake with all the valves open, and have provided an automatic valve above the system rotary specifically for this purpose. If a power cut should occur during the bakeout process, this valve would close, protecting the system from any air ingress, and the bakeout process cease. It will not restart when the power is reinstated, and the valve will remain closed.

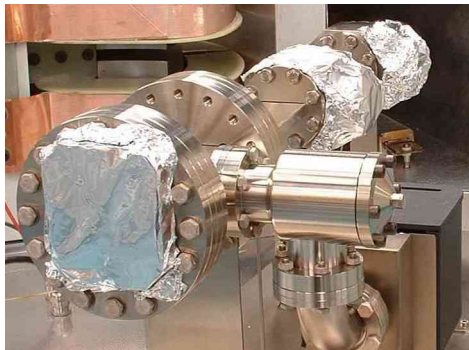
Since a lot of heat is unfortunately lost below the bench top during the bakeout process, you should now remove as many of the bench covers as possible. This will permit air to flow freely around the electronic units during the bakeout, and prevent any damage due to overheating. To put this into context, during the bakeout process itself, the ion pump pressure will increase, which in turn increases the power which the controllers have to supply. They must have free air circulation during this time.

The magnet should now be moved to the rear of its travel (to the front of the bench). To free it from its set position, undo the central knurled nut at the end plate below the magnet coils. Do not alter either of the outer two set bolts on this plate, since these define the magnet position when it is pushed back into place. If the "earthquake" bolt is fitted into the bottom of the magnet below the bench top of the instrument, it will

also be necessary to unbolt it. Slowly push the magnet to the rear of its travel (take care not to trap the wires to the coils or magnet control box), and preferably lock it into place using the “earthquake” bolt under the bench top surface. It is now necessary

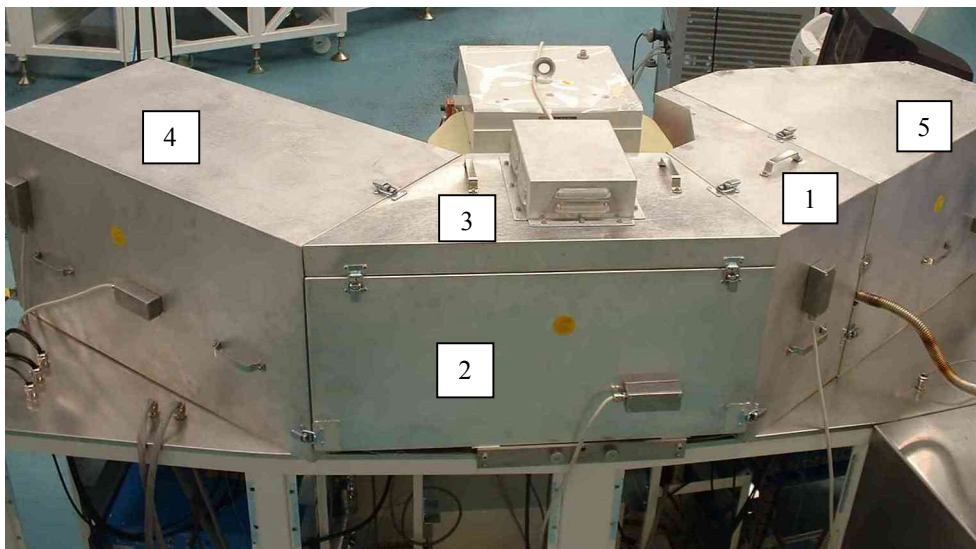


to fix the central thermocouple to the flight tube centre, using a piece of fibreglass tape. Plug in the thermocouple connector to the socket under the central controller (labelled “Flight Tube” in the bakeout control region of the Mains Distribution Unit).



You should now cover the exposed ceramic feedthroughs with some aluminium foil. Standard cooking foil is quite adequate. This will ensure that the fragile ceramic areas do not experience undue heating stress from any direct line of sight heating.

Now fit the ovens. This should be done in the order indicated in the following picture:



Please note that cover 4 should be lowered over the mass spectrometer from above, so as to ensure that the heaters do not hit any of the vacuum enclosure. Similarly item 5 should be lowered onto the bench from above, and then gently slid into contact with item 1. In this manner, the manifold connection tube, and automatic manifold valve (if fitted) will be easily accommodated in the slots cut out of the side of this item. Lock the separate oven sections together using the latches fixed at their edges.

Plug in the heater wires into the relevant sockets in the bakeout region of the Mains Distribution Unit. Note that item 1 and 5 are powered from the same supply, and so the lead from the heater of item 1 should be plugged into the spare connector of the 3



way splitter cable, before the plug of this cable is inserted into the “Source” output. Plug in the fan supply cable (seen laying over the magnet in the picture above) into the output labeled “Fan” on the bottom row of the bakeout sockets. Ensure that the bakeout switches are both switched on (a small green switch on the bottom right of the MDU unit, and the large circuit breaker to the left).

Set the three temperature controllers to the required bakeout temperature (we do not recommend going above 300° C since the ion pump magnets undergo a permanent magnet transition above this temperature. If a significantly higher bakeout temperature is required, it will be necessary to remove these magnets from the ion pumps.

It is now possible to start the bakeout process, using the bakeout window from the menu list of the main software suite (see chapter 6). It is recommended that you use the default settings loaded, since this ensures that the rate of temperature rise is not too fast, and the system will protect against excessive rise of pressure. Also, when stopping the bakeout at about 5am in the morning, should ensure that the rate of temperature fall is also safe.

At the end of the bakeout, the ovens should be unplugged and carefully removed. When the vacuum envelope has cooled sufficiently to be only warm to the touch (obviously test this with care!), the collector preamplifier may be fitted, and all the cables reconnected. Once the source magnets have been placed back over the source housing, the “Max Power” to the filament should be slowly raised to the original setting, and the “Instrument Setting” protection re-enabled.

Once the envelope has fully cooled, you may wish to re-activate the source getter, as described above.

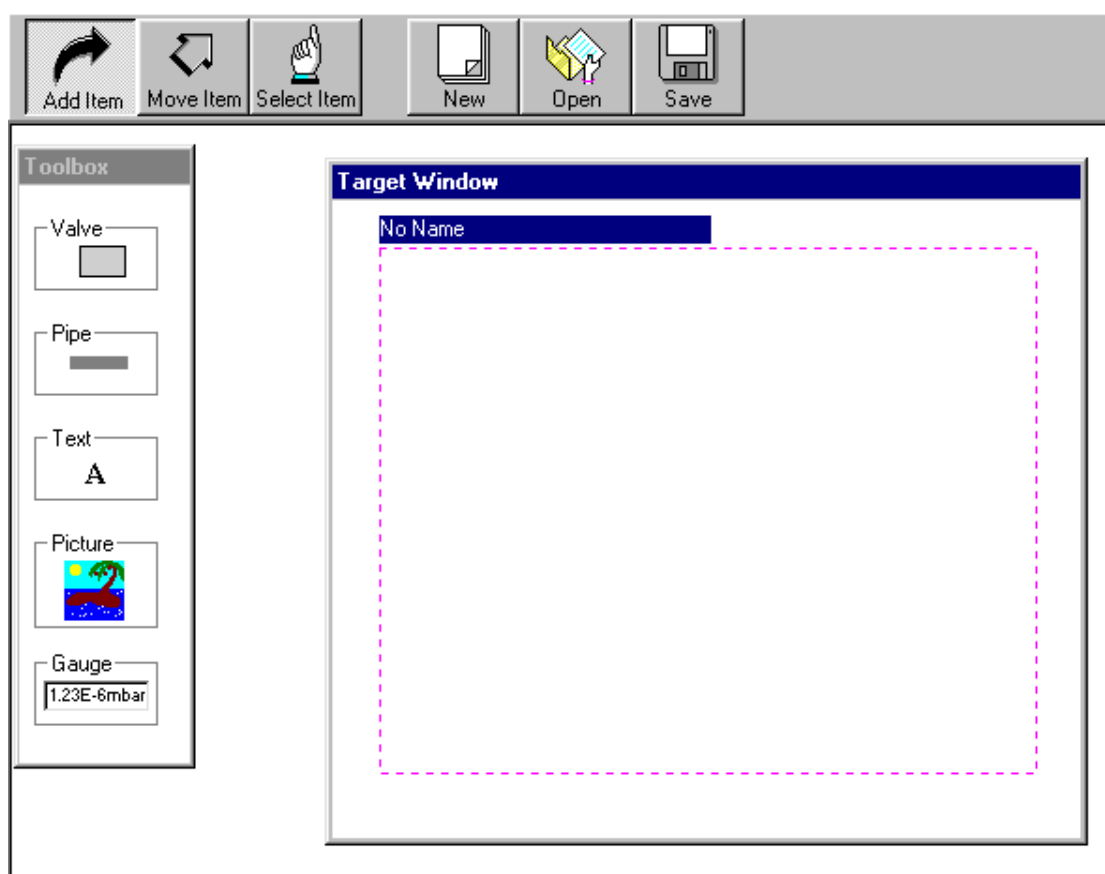
## Chapter 10: The *Nu Designer* Utility

This program allows the user to create diagrammatic representations of preparation systems and add-ons to the Nu Instruments Noblesse mass spectrometer. These diagrams can be read and displayed by the programs controlling the mass spectrometer, and can provide user interaction with valves and switches, as well as provide feedback of the observed outputs from a range of sensors. It thus provides the user with a means of extending the powerful control provided with the mass spectrometer, to custom designed accessories.

The utility is in the form of a basic drawing package, which allows a schematic representation of the accessory to be input, and overlaid with valves, pipework, gauges and text descriptions. The utility further provides the user with an interface to input information concerning the valves and gauges, such that the main, mass spectrometer, control program can talk to and read these active items. As such it removes the necessity to resort to separate control systems for preparation systems, whilst permitting them to be fully integrated into the main suite; thus enabling full automatic control of the complete preparation and analysis package.

### The Workbench Environment:

On starting the “Designer” program, the user will find the following window:

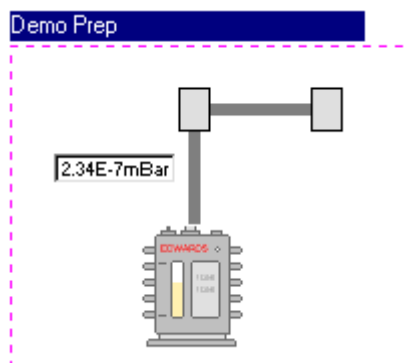


There are three main areas of interest:

Along the top is a set of menu buttons, split into two groups. The left-hand side set permits the target drawing to be manipulated, whilst the second group contains the familiar file operation buttons.

The right hand window (identified as the “Target Window”) is the region in which the desired schematic can be drawn and manipulated. Within this window is a dotted box (whose size may be altered), which represents the final size of the target window. The caption above this area (containing the words “No Name”) represents the title bar of the final window.

The toolbox to the left holds the five component types, which are used to create the final window.



We will illustrate the use of the package to create a simple preparation system consisting of two valves, interconnecting pipework, a pump and a simple pressure gauge. We show this simple example on the left. You will have seen this example earlier, in chapter 6, where it had been imported into the main program.

## Creating the Diagram:

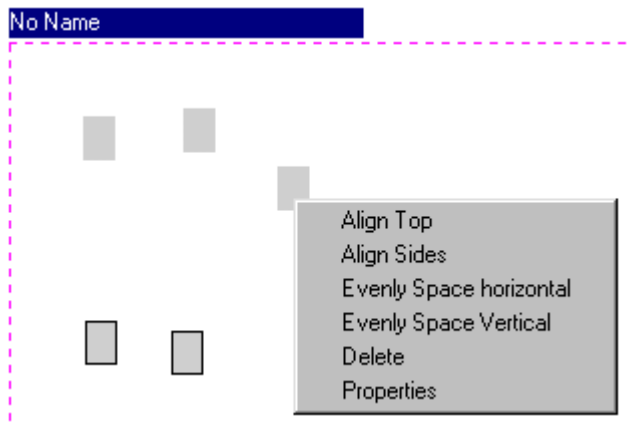
### Placing Valves on the canvas:

To add a valve, select the “Add Item” menu option on the top menu bar. When the cursor is moved over the toolbox items, as it passes over each one, it will be seen to change from the standard sloping arrow to a hand, as it passes over each diagrammatic representation. Move the arrow to the “valve” tool, and when the hand appears, depress the left mouse button. The text of the selected item changes from black to red, identifying the item, which can now be added to the canvas. With the valve tool selected, move the mouse over to the target window, where the cursor will change to a cross. Depressing the left-hand mouse button a second time will result in a valve icon being drawn at the selected position. Further valves may be added by repeating this procedure.

### Moving Valves over the canvas:

Most moving will be done by using the “Move Item” menu option on the top menu bar. With this depressed, the required item may be “picked up” as the cursor moves over it, by depressing the left-hand mouse button. By keeping this button down, and moving the mouse will drag the item with the cursor. It can be deposited at its final destination by releasing the mouse button. If it is dropped outside the target window area, the program assumes the item is no longer required and it is deleted from memory. (Other ways of deleting some of the items will be discussed below).

Sometimes, a whole series of valves may be present, such as in a simple manifold arrangement, where up to 10 or 20 valves could be present. Obviously in cases like these, it



is sensible to try to lay out the valve icons in a geometric arrangement, with equal spacing between each icon. Rather than use the “Move Item” button to achieve this affect, the program provides a simple method for automating this task. To illustrate this option, place a number of valves on the form, and then depress the “Select Item” menu option on the top menu bar. If you press the left-hand mouse button as the cursor passes over an

icon, which you wish to select for further manipulation, the item becomes “selected” and will flash. You may select as many valves as you wish to become simultaneously “selected”, and this option will allow the automatic moving to be achieved. With say three valves “selected”, depress the right hand mouse button as the cursor passes over one of these icons. A floating menu bar will then appear, as shown here. Choosing the “Align Top” option for the three selected valve icons (shown without the border in the example) will place the three icons level with the highest of the group selected, the “Align Sides” option doing the corresponding placement to the leftmost icon. Aligning the three to top, followed by aligning to the sides, will place all three icons on top of each other, which is as required, but a bit disconcerting if you don’t realise what has happened. They may be separated using the “Move Item” menu option. The “Evenly Space horizontal” and “Evenly Space vertical” menu options can be used to uniformly space the icons.

### Placing Pipes on the canvas:

As you would expect, to place a pipe on the canvas, it is first necessary to select the “Add Item” menu option from the top menu bar, and select the pipe tool from the toolbox. To add a pipe is slightly different form a simple valve, in that a pipe obviously has two ends. Thus when you move the cursor over to the target area, and turns to the cross, the first end of the pipe is placed where you first give a left-hand mouse click, and the pipe drawn to the position of the second click. However, other aids are provided to make the job of connecting items easier. Firstly the pipe tool draws orthogonally, from the first end placement. Of more interest is when ends of the pipe join valves. Here, the program automatically centres the pipe to the relevant valve edge, ensuring that the drawing looks neat. If the two valves are not aligned, the centring is to the second icon. Further the program will try to join pipes together intelligently if complicated pipework arrangements are required. If you prefer not to use this feature, or the program doesn’t produce the network you require, the pipes can of course be moved manually via the “Move Item” menu option button.

### Pictures:

When you attempt to place a picture on the target canvas, using the picture placement tool, after the position for the placement has been marked using the left hand mouse button, the program will display the “Open File” Windows Dialog Box. This is to enable the required picture to the identified. Three file extensions are supported:

GIF Images (with the .gif extension)

Bitmap Images (with a .bmp extension)  
Icons (with the .ico extension)

The rotary pump shown in the example above is supplied as the file “Rotary.gif”, whilst a turbo pump may be found in the file “turbo.gif”. Obviously you can create any other image you wish using commercial drawing packages. It is probably sensible to load the required image anywhere on the target area, and move it later to its required position via the “Move Item” menu option.

As in all composite designs, the various components are drawn on different “layers”. We have adopted the convention of placing pictures on the lower level, and the other items on a higher level. This enables one to have a large picture (for instance representing the complete instrument) and having valves, gauges and text shown on top of it. If the other way of ordering items was used, the picture would mask these items. This usually is not a problem, but if one moves a picture, some unexpected effects are seen. During the move, the picture becomes the item of focus, and masks these other items. Put another way, they disappear, which can be very disconcerting! To regain the normal ordering of things, select the picture with the “Select Item” button, and depress the right-hand mouse button. This will cause the picture to be placed onto the lowest level, once again, and those hidden items will magically reappear.

#### **Adding Text to the canvas:**

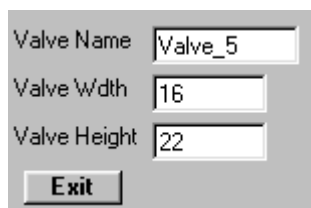
This is done in two stages. Firstly a text box is placed in (approximately) the required position using the “Add Item” menu button. The empty text box is shown as a small rectangle, with nothing inside. The actual text is then added by selecting the textbox using the left-hand mouse button after the “Select Item” menu button has been depressed. Clicking the Right-hand mouse button with the cursor over the text box will then bring up floating menu bar which will allow for text input via the “Edit” option. It is also possible to define the font, font size and colour using other menu options.

#### **Placing Gauges on the canvas:**

This is achieved, as expected, using the “Add Item” Menu button and dragging a gauge from the toolbox onto the target area.

#### **Customising Gauges and Valves:**

It is necessary to identify the valves and gauges added to the canvas so that the main instrument program can control and read them. This is achieved via the “Properties” menu option on the floating menu bar which may be accessed using the “Select Item” menu button, followed by a right-hand mouse button click over the relevant icon. The adjacent window shows the valve property version. The “Valve width” and “Valve height” entry boxes are obvious (note all entries are in pixels units), and may be altered as required. The Valve name entry MUST correspond to an entry in the “MS.DEF” file, which has been downloaded to the system micro. This restriction is actually sensible, since when a user clicks on the valve icon on the main controlling PC screen, something will be expected to happen with the relevant valve, namely toggle its state. Thus the system must be able to



The image shows a dialog box for configuring a valve. It has three input fields: "Valve Name" with the text "Valve\_5", "Valve Width" with the value "16", and "Valve Height" with the value "22". There is an "Exit" button at the bottom left of the dialog.

properly identify each valve shown, and this is only possible if the name used is also defined in the “MS.DEF” file. The match has to be exact, and although the names are case sensitive, the host PC will convert all to upper case, so as to make input easier. Also, spaces are not valid character in the “MS.DEF” file, but will be allowed here. The underscore is often used in their stead, as shown in the example given. Again the host PC will convert, but the user should be aware of what is occurring if similar names are used. Thus “Valve\_5” will be the same as “Valve 5”, but different from “Valve5” (with no space). However this makes the code difficult to check and maintain, and so using differing conventions is not recommended. When the main PC program tries to read the file generated by the “Designer” program, it will check that all entries are valid, and an error will be raised if the names do not match. The form of the “MS.DEF” file is discussed in more depth in chapter 3.

The Gauge customisation is a little more difficult, in that not only must the host PC be able to talk to the gauge (and hence know its name) but it must be able to convert it’s output (normally 0-10volts) to some meaningful, for example, pressure or temperature. The window to input these data is shown here, using a Pirani gauge as an example. Often modern gauges have built in intelligence, and the output is simply related to pressure (they are said to be “linearised”). Others require the user to do this conversion. As has been mentioned previously, The host PC program undertakes two forms of data conversion:

linear:            True reading =         $A + Bx + Cx^2 + Dx^3$

and exponential:

                    True reading =         $\exp(A + Bx + Cx^2 + Dx^3)$

As can be seen the example given uses the exponential fit with all four coefficients being necessary to achieve a necessary precision. The coefficients were found (in this case) by using the manufacturer’s data and fitting to the third order (4 coefficients) equation. Please note that since an exponential fit is used, if the gauge is “linearised”, such that  $10^{-9}$  mbar has a gauge output of 1 volt,  $10^{-8}$  mbar an output of 2 volts etc, the factor 2.3026 will have to appear in the coefficients in order to convert between natural logs, and logarithms to base 10.

The “Units” input field has been added to allow any general sensor to be monitored, rather than limiting the selection to pressure sensors. The input characters will be displayed in the gauge textbox, as shown in the example of the “Demo Prep” shown above.

### **Changing the Title of the Window:**

This is simply achieved by using the “Select Item” and right-hand mouse button when the cursor is above the dummy named window bar, and using the “Edit” menu. This is identical to changing text in the standard textboxes.

### **Changing the Size of the Window:**

With any of the left-hand menu buttons depressed, as the cursor passes over any of the dotted boundary lines, it changes to a double-sided arrow. Depressing the left-hand mouse button with the cursor in this state will cause the dotted line (and hence window boundary) to be altered.

### **File Control:**

Once you have produced your masterpiece, it may be saved using the “Save” menu button on the top menu bar. Depressing the button will result in the standard “Save” dialogue window being displayed, and the file is saved with the “.nsf” extension, as required by the host PC program. Existing files may be viewed using the “Open” menu button, but again only those files with the correct extension are listed in the dialogue window.

### **Modifying the host PC program to show the new window:**

The required information is normally held in the file “Startup Parameters.DAT” in the “Instruments Settings” folder of the main control software suite. The only “trick” to remember here is that this file is overwritten whenever the main program is stopped, so that, in order to have the change registered, it is important to exit from the main program. If this is not done, the changes you do will magically disappear the first time that they are tried out!

So with the main program fully stopped, look in the file “Startup Parameters.DAT” for some lines similar to the following:

```
"Number of manifold form files ",1  
"Form file number 1 ","C:\ Nu Noble\Instrument Settings\NG.nsf"
```

This example tells the main program that only one file is to be loaded (the default, which should only be changed if you know what you are doing!). To add the second file the number of form files is incremented and the location of the new file given after the first location. The input will then look similar to:

```
"Number of manifold form files ",2  
"Form file number 1 ","C:\ Nu Noble\Instrument Settings\NG.nsf"  
"Form file number 2 ","C:\ Instrument Settings\Demo Prep.nsf"
```

where we have now added the “Demo Prep” window. Please note that the name of the file need not be the same as the title shown for the window, although it may be sensible to use obvious names if you intend to create a large number of these windows. The host PC program is limited to five files in total.