SE9400036

STOCKHOLM UNIVERSITY

DEPARTMENT OF PHYSICS

DESCRIPTION OF A LASER VAPORIZATION SOURCE AND A SUPERSONIC CLUSTER BEAM APPARATUS

M. DOVERSTÅL, B. LINDGREN, U. SASSENBERG and H. YU

USIP Report 93-11

November 1993

Description of a Laser Vaporization Source and a Supersonic Cluster Beam Apparatus

M. Doverstål, B. Lindgren, U. Sassenberg, and H. Yu
Department of Physics, Stockholm University

Abstract

Laser vaporization of an appropriate target and recent developments in molecular beam technology have now made it possible to produce supersonic cluster beams of virtually any element in the periodic table. This paper describes the design and principles of a cluster source combined with a time of flight mass spectrometer built for reaction experiments and spectroscopic investigations at Stockholm University.

1. Introduction

Investigation of clusters is a fairly new field of research especially considering metal- and semiconductor clusters. A cluster can be characterized as an aggregation of physically united atoms or molecules. The size of a cluster varies from two to hundreds of thousands of atoms or molecules. The study of metal clusters is expected to be important for the understanding of surface phenomena such as hydrogen storage and heterogeneous catalysis. Other fields where especially carbon clusters are believed to be of importance is astrophysics and super conductivity physics.

One of the reasons for these expectations is that a cluster in a beam has a large (compared to its volume) and totally clean surface. Since the area per volume ratio for a cluster is proportional to $n^{-1/3}$, where n is the number of atoms or molecules in the cluster, one finds that the surface of a typical cluster indeed is very large.

The technique of laser vaporization was developed by Smalley *et al.* [1] in the early 80's. It was mainly designed to make refractory metals in the gas phase, which so far had been difficult. The new method made it possible to produce vapour of almost any element. Free expansion of a mixture of the vapour and a carrier gas which made it possible to cool the clusters to extremely low temperatures. The carrier should be inert towards the vapour and therefore an noble gas, usually argon, is used for this purpose.

With financial support from the Knut and Alice Wallenberg Foundation and the National Swedish Board for Technical Development a laser vaporization cluster source and TOF mass spectrometer has been constructed at this department.

2. Principle of the apparatus

The method and apparatus design follow the principles developed by Smalley and CO-workers at Rice University [1]. Their method can be described briefly in the following way: A pulsed laser, the second harmonic from a Q-switched Nd:YAG-laser, is focused onto a target (The fundamental wavelength usually works well but since it is invisible to the naked eye the use of it makes alignment more difficult.) The laser vaporizes a certain amount of target material directly from solid to gas phase. No part of the construction is heated.

The vaporizing laser beam is focused on a surface of either a rod or a disk with a spot size of 1 - 2 mm in diameter. To prevent the laser from drilling a hole in e.g. the rod, it is slowly rotated and translated by a screw in such a way that every laser pulse vaporizes a fresh part of the rod surface. The rod is placed close to the nozzle of a pulsed helium beam. The laser pulse is synchronized with the peak density of a helium gas pulse controlled by a specially-designed fast valve. This valve is operated by solenoids which can be pulsed by currents up to 100 A. As can be seen from Fig. 1 the carrier gas sweeps the vapour into a narrow channel through the nozzle.

A high temperature vapour is formed in the ablation process when the latter part of the 4 - 8 ns long laser pulse heats it to ionizing temperatures. The high density of carrier gas surrounding this plasma causes a re-combination process, i.e. cools it so that condensation of the vapour to clusters becomes possible. By varying the length and shape of the channel the distribution of cluster sizes can be varied within broad limits.

When this hydrodynamic flow of clusters and helium carrier gas is allowed to expand freely into vacuum (10^{-5} torr) it causes a substantial cooling lowering the temperature to a few Kelvin. A skimmer is used to isolate the central part of the formed cluster beam for further studies. The beam is well collimated and almost collision-free. It can now be investigated in a high vacuum environment in several ways.

Mass spectroscopy using a time-of-flight (TOF) mass spectrometer is a common way to monitor the cluster production. If the carrier gas is mixed with, or if another gas is added downstream the beam, the reactivity of the clusters can be studied. Fig.2, taken from Smalley *et. al.* [1], shows the intensity distribution of some metal clusters as studied by time-of-flight mass spectrometry.

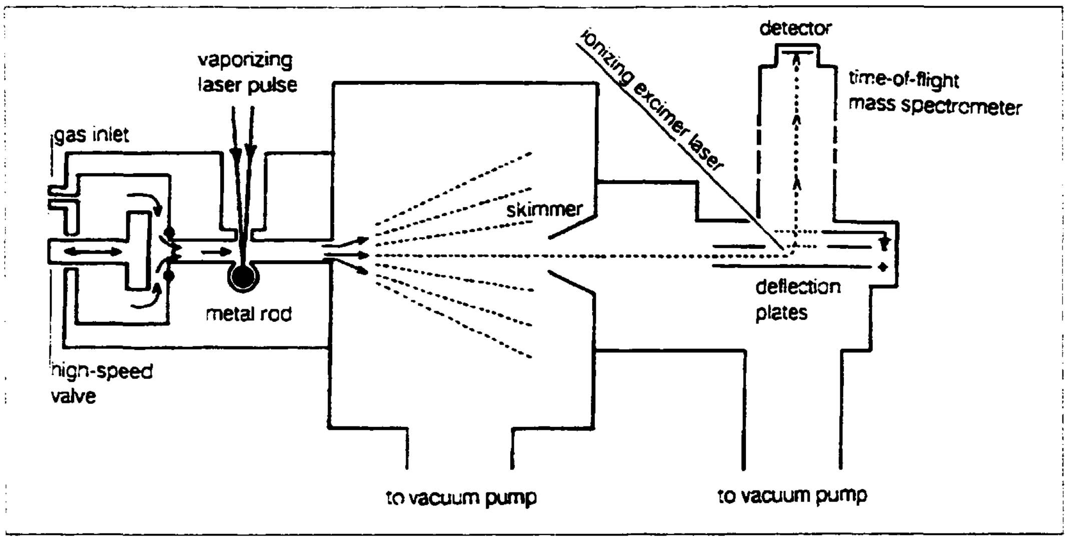

Fig. 1. Principles of the apparatus: Clusters are generated by laser vaporization of the target material in a channel flowed through by a carrier gas. Inter molecular collisions cool the vapor and thus allowing it to condense and to form clusters of varying sizes. The cluster/carrier mixture expands rapidly into vacuum hence cooling the clusters to almost zero temperature. The clusters are ionized by a short wavelength laser and accelerated in an electric field. Different masses are separated according to the time spent in the time of flight tube. Simultaneous spectral information can be achieved in a process involving resonance enhanced multiphotonionization when exciting an intermediate state with a tunable laser.

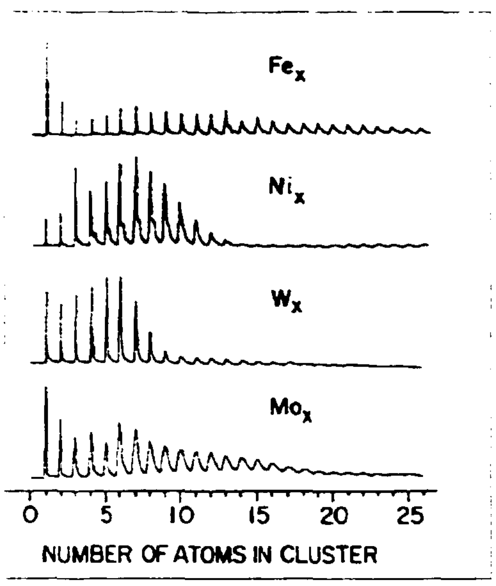

Fig. 2. Time-of-flight mass spectra of some metallic elements. The clusters are produced by laser vaporization.

A large fraction of the produced clusters are electrically neutral. In order to mass analyze the clusters they must be accelerated in some manner. The simplest way is to ionize the neutrals in a static electric field at the beginning of the spectrometer. Very often short-wavelength pulsed lasers (excimers) are utilized to photo ionize the neutrals thus providing a convenient starting pulse for the TOF spectrometer [2,3].

Resonant two-photon ionization spectroscopy (R2PI) is a method to investigate the electromagnetic spectra of small clusters. Two photons are absorbed by the neutral cluster. The first photon (from a pulsed dye laser) excites the molecule to an intermediate state and the second photon (from a YAG or excimer laser) ionizes the molecule. One have to make sure that the energy of the second photon is enough to ionize the molecule from the excited state but less than the I.P. The ion production is then measured as a function of the frequency of the first photon. The method simultaneously produces a masspectrum and gives information about resonance transitions for clusters of selected masses [4,5].

Information about the lifetime of the intermediate state is provided by varying the delay time of the ionizing laser pulse relative to that of the tunable laser [6]. It should be noted that not only different clusters can be selected but also the different isotopic variants (isotopomers) of a certain cluster [4].

3. Laser ablation source.

This type of laser ablation source, briefly described above, has now become very common and is usually called the "Smalley-type source". In our case we have chosen to use the fundamental frequency of a Q-switched Nd:YAG laser (Lumonics Mini-Q, originally intended for ophthalmology) for vaporization. The maximum pulse energy is 18 mJ in TEM_{00} mode and 120 mJ if the internal aperture is removed to allow all modes.

The laser light is focused onto the metal rod by a f = 25 cm lens. It is desirable to make the entrance channel for the laser light as narrow as possible since plasma is lost into this channel during vaporization. The diameter of the light channel was originally 0.5 mm but it soon turned out that such small a diameter made it very difficult to align the laser beam. When the hole was enlarged to 1 mm the alignment became fairly easy but the back streaming of plasma caused problems with dirty windows and clogging of the channel (when using a carbon target). This problem was solved by feeding the vaporizing laser light through a glass micropipette.

A small DC motor rotates and translates the rod at a speed of 1 rpm. As the pitch of the screw is 0.5 mm a 35 mm long rod can expose a fresh surface for at least 60 min. at a repetition rate of 10 Hz. The rod translator and the laser entrance hole are placed on the same stainless steel block, which can be moved at right angels to the optical axis of the apparatus (Fig. 3). To improve cluster production stability the rod must be prevented from wobbling. This is done by a spring load pushing the rod towards one side. The volume of the vaporizing chamber can be adjusted and tuned for maximum production of specific clustersizes. This volume is sometimes referred to as "waiting room" [7].

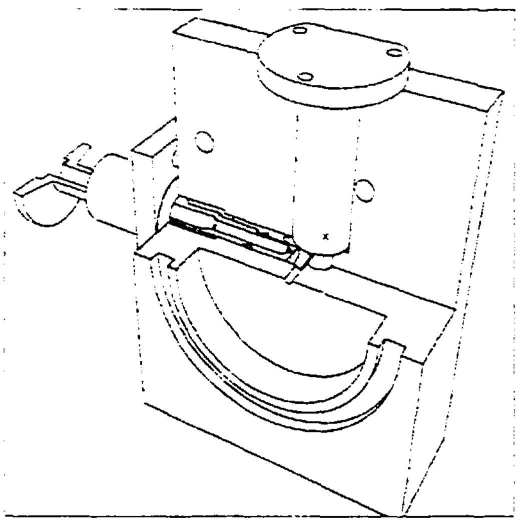

Fig. 3. Stainless steel block. Light enters from the left, through the flange and micro pipette. The gas pulse is injected perpendicular to the light channel. The cluster/carrier mixture expands into the vacuum chamber from the other side of the block.

The helium beam is controlled by a specially designed solenoid-driven valve (Fig 4). The tip of the piston turned out to affect the performance of the valve extensively. Several types of tips were tested and one made from a short piece of a 3.2 mm diameter o-ring turned out to be the most reliable construction. The major advantage with this construction is that the valve only has to open against the force caused by the small area of the crossection of the o-ring as opposed to when an ordinary o-ring forms the seat.

No spring load is necessary for closing since the piston bounces very swiftly against a heavy metal rod. The position of this metal rod is adjustable from the outside and the behavior of the valve is very sensitive to this adjustment. To prevent further bouncing of the piston a weak second solenoid, actually just a coil, keeps the valve in closed position. Minimum opening time is around 100 μs which has been measured with a condenser microphone inside the vacuum chamber.

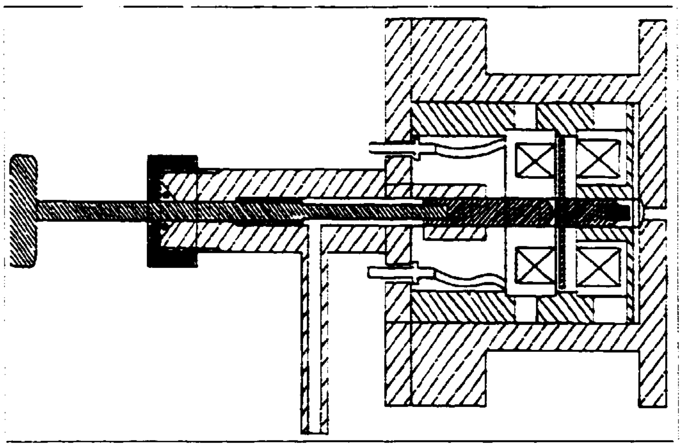

Fig. 4. Valve design. Two solenoids one of which with a ferrite core are placed on each side of a circular soft iron disc. Running a current pulse trough any of the solenoids causes the disc to be attracted to it. A stainless steel piston is attached to the disc. The tip of the piston seals against a polished slightly conical stainless steel seat.

The solenoid current supply is a simple construction consisting of a variable transformer and a rectifier bridge. The capacitor is quite large. The pulses are provided by SIP-MOS power transistors, two for the open pulse and one for the close pulse. The circuit drawing is shown in Fig. 5.

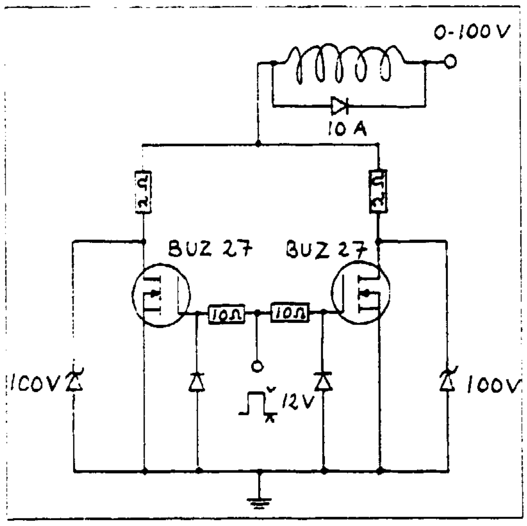

Fig. 5. Solenoid switch.

4. Vacuum details

When designing the apparatus the physical dimensions of the two diffusion pumps and the size of the laboratory gave limitations to the construction. Since it was regarded as an advantage if the whole apparatus was movable the two diffusion pumps were placed on a trolley with the vacuum chambers on top of the pumps (Fig. 6).

The large diffusion pump has high pumping capacity (6000 l/s) and can maintain a low pressure in the system even with a heavy gas load. The amount of gas in one pulse can be much larger than in a more conventional cluster apparatus.

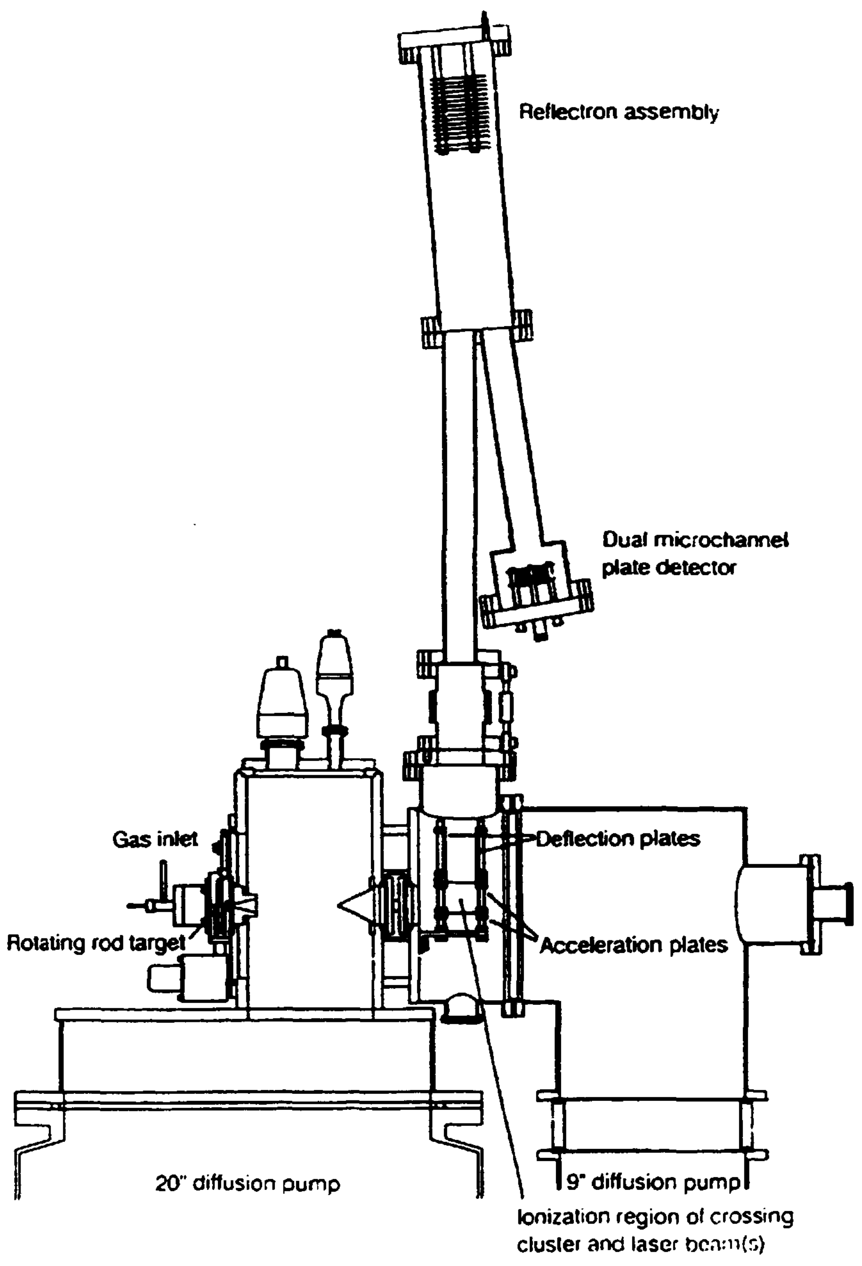

Fig. 6. Total assembly. From the cluster source the beam expands into a vacuum chamber which is mounted on the water cooled baffle of a 20" oil diffusion pump (Heraeus Di 12000). The central part of the created super sonic beam is isolated and transferred into the next chamber by a skimmer (Beam Dynamics, Inc. Model RU-83-1). The cluster source and the two chambers can be isolated from each other by flat valves (VAT Series 12). The second chamber serves as the experimental region where the time of flight mass spectrometer is attached. It is evacuated by a separate 9" oil diffusion pump (Edwards F903).

The backing system of the diffusion pumps consists of two parallel rotary pumps (Alcatel 2033) connected via metal bellows to the rough vacuum steel tube line. The pressures are controlled by a Leybold Combitron system with two Pirani and one Penning gauge. The second high vacuum chamber is also equipped with an ionization gauge (Leybold Ionivac IM 220) since the required vacuum has to be better than 10^{-6} torr.

5. *Free expansion*

The free expansion and subsequent cooling of the vapour depends mainly on the Mach number (M) of the beam and the c_p/c_v ratio (γ) of the carrier gas. Let the temperature before the expansion be T_0 and after the expansion T. If the expansion is isentropic one can relate the temperature decrease to the Mach number as: $T/T_0 = (1+M^2(\gamma-1)/2)^{-1}$ [8]. It is obvious that the temperature decrease can be substantial ($M = 5$ and $\gamma = 5/3$ gives $T = 0.11\ T_0$, that is $T = 32$ K if $T_0 = 293$ K).

The most probable velocity, v_{mp}, for a free gas phase molecule at any temperature T is $v_{mp} = \sqrt{2kT/m}$. The average speed of the molecules at maximum beam intensity is on the other hand given by $\sqrt{3kT/m}$ in an effusion source. The ratio between v_{mp} and the beam velocity v_b after complete expansion can then be written as $v_{mp}/v_b = \frac{\sqrt{2/\gamma}}{M}$ at this new temperature T. This ratio is also a measure of the beam divergency since it reflects the ratio $v_{x,y}/v_z$ with a beam proceeding in the z direction. If the beam initially is 2 mm wide it will be around 7 mm at the skimmer. Since the orifice of the skimmer is 5 mm one finds that half of the beam is lost.

The geometrical dimensions of the apparatus gives a Mach number at the skimmer of $M \simeq 60$ when a mono atomic ($\gamma=5/3$) carrier is used and $M \simeq 20$ when a diatomic carrier ($\gamma=7/5$) is used. The consequent translational beam temperature T is therefore 0.25 K and 3.6 K respectively. It should be noted that the temperature depends only on γ and the Mach number. The rotational temperature, however, does depend on the mass of the carrier in the sense that a heavier carrier gives a lower T_{rot}. There is no simple way to estimate the T_{rot} but quite generally $T_{trans} < T_{rot} < T_{vib} < T_0$, because cooling might take place in the reverse order. Here T_{trans} is the temperature associated with the translational movement. That the rotational temperatures indeed depend on the mass of the carrier can be seen from the facts that argon in our application on the copper dimer gave a rotational temperature around 7 K. The corresponding temperature for helium was 15 K.

When dimensioning the apparatus it is important to avoid that a shock wave, the so called Mach disc, shatters the molecular beam. The position of the Mach disc depends on the reduced distance L/x, where L is the distance from the nozzle and x its aperture, and the pressure ratio P/P_0, where P is the pressure in the chamber. With $P \approx 10^{-5}$ torr, $P_0 = 1$ atm and x = 2 mm the disc is expected to be located at L well over 1 m. T'e distance between the nozzle and the skimmer should therefore be less than this distance.

The speed of the cluster beam can easily be estimated from knowledge about the Mach number. Table 1 summarizes the beam velocities and temperatures for different carriers.

Table 1 Beam velocities and temperatures for different carriers.

Carrier	v_b (m/s)	T_{trans} (K)	M	γ
H_2	2955	3.62	20	1.41
He	1715	0.24	60	1.66
N_2	770	3.62	20	1.40
O_2	730	3.62	20	1.40
Ne	775	0.24	60	1.64
Ar	570	0.24	60	1.66
Kr	390	0.24	60	1.68
Xe	315	0.24	60	1.66

It is of interest to estimate the mean free path of a molecule in the second chamber at $P = 10^{-6}$ torr. The number density at T = 293 K is 3.5×10^{10} molecules/cm^3, hence the mean free path is approximately 70 m. This distance is almost two orders of magnitude greater than the dimension of the appa-atus. One can thus conclude that inter-molecular collisions do not need to be considered. It is usually difficult to estimate the specific cluster density in the beam. Without proper ways of measuring even the primary gas pulse after the skimmer any effort to get a hint about a specific cluster density is doomed to fail. From experiments on carbon we know that we often record of the order of 20 ions per pulse and cluster size for heavy carbon clusters.

The recording efficiency is high but the photo ionization cross section is unknown. In a specific example with focused light and a cluster beam with a ionized volume of the order of 0.03 cm^3 we apparently get > 500 ions per cm^3, specific cluster

size and pulse. This means that there are probably $> 10^3$ clusters/cm^3 of heavy carbon clusters in the beam.
Based upon the pump capacity every skimmed gas pulse contains of the order of 10^{14} molecules/cm^3. The constituents are carrier gas atoms and a large fraction of atomic carbon besides a broad fraction of soot particles and some particular clusters when referring to carbon. On the other hand the density is much higher just outside the nozzle where fluorescence technique can be applied, (the spectrum of TiO has been recorded in this way). This indicates a number density $> 10^7$ molecules/cm^3, which is usually taken as a lower limit in laser excitation experiments for ordinary molecules.

6. *Mass spectrometer*

The mass spectrometer is of time-of-flight (TOF) type and it can be used in two different configurations, with or without a reflectron. It consists of centrally mounted accelerator plates followed by deflector plates and an optional einzel lens. In the straight TOF spectrometer this arrangement forms the starting point for a perpendicularly mounted 1.2 m long flight tube. The detector is a dual micro channel plate package (R.M. Jordan Co.). All necessary voltages are provided by a stabilized high voltage power supply (Wentzel Elektronik N1130-4). A Varian high voltage lead-through mounted on a UHV flange is used for current supply. With this configuration one will obtain a practical mass resolution of about 200 ($R=m/\Delta m$).

When the neutral clusters are ionized in the static electrical field their final velocity spread, i.e. the mass resolution, depends on where in the field they are ionized. It is therefore important to keep the ionized volume as narrow as possible. This is done either by a slit in front of the ionizing laser or by a focusing lens. This is of course a compromise since the ion production depends on the size of the ionized volume.

Single field acceleration. The final velocity of a molecule ionized in a static electric field depends on three parameters, initial position, thermal velocity of the ion and field strength. Consider a molecule ionized at distance s from the ground electrode in an electric field of strength E_{acc} (Fig. 7). The thermal velocity of the ion in the direction of the field is v_E. The final velocity can be written as:

$$v_{single} = \sqrt{\frac{2qsE_{acc}}{m} + v_E^2} \tag{1}$$

where q is the elementary charge and m the mass of the molecule. Since v_E denotes the projection of the thermal velocity in the direction of the field it can be in the range $-v_{th} \le v_E \le v_{th}$, where v_{th} is the average thermal velocity of a molecule at temperature T. One notes that v_{acc} does not depend on the sign of v_E.

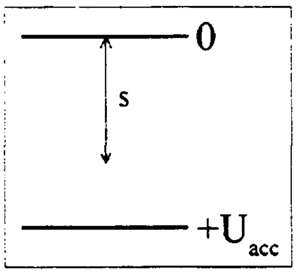

Fig. 7. Geometry of the single field.

The time spent in the accelerating field depends on the sign of v_E in the following way:

$$t_{acc} = \frac{m(v_{single} \mp v_E)}{qE_{acc}} \tag{2}$$

taking v_E positive in the direction of the field. The total flight time in a straight flight tube of length ℓ is then given by:

$$t = \sqrt{\frac{m}{2qsE_{acc}+mv_E^2}} \cdot \left\{ \ell + 2s + \frac{mv_E^2}{qE_{acc}} \right\} \pm \frac{mv_E^2}{qE_{acc}} \tag{3}$$

One can thus conclude that the thermal velocity produces a maximum uncertainty in the flight time as $\Delta t_{th} = \frac{2mv_{th}^2}{qE_{acc}}$. If Δt_{th} is evaluated for Zr_2 (m=180), E_{acc}= 75 kV/m and T=0.25 K one obtains $\Delta t_{th} \simeq 1.3$ ns. It turns out that the uncertainty produced by the thermal velocity is negligible since $\Delta t_{th} \ll t$. It is thus possible to set $v_E = 0$. Equation (3) then reduces to

$$t = \sqrt{\frac{m}{2qsE_{acc}}} \cdot (\ell + 2s). \tag{4}$$

The fact that the width of the ionizing laser beam can not be infinitesimal gives a spread in s which one has to take into account. In the early stages the third harmonic of a Nd:YAG laser (355 nm) was used to ionize the molecules. The laser beam was focused onto the molecular beam with a cylindrical lens. The vertical size of the YAG pulse was around 1 mm and hence $\Delta s = 0.5$ mm. This results in a variation in v_{acc} and consequently in t. Denoting the difference in flight time as $\Delta t_s = \frac{\partial t}{\partial s} \Delta s$ one obtains

$$\Delta t_s = \sqrt{\frac{m}{8qs^3E_{acc}}} \cdot (\ell - 2s)\Delta s \tag{5}$$

Evaluating Δt_s for Zr_2, taking $s = 30$ mm, $\Delta s = 0.5$ mm and $E_{acc} = 75$ kV/m, gives $\Delta t_s \simeq 190$ ns. It is obvious that the uncertainty in s is a serious problem when doing mass spectroscopy.

The mass resolution, R, for this type of setup can be estimated in the following way. Consider the flight time for two molecules of mass m and $m+\Delta m$. The difference in their flight times can be written as $\Delta t_m = \frac{\partial t}{\partial m} \Delta m$. Equating this expression with the uncertainty depending on Δs one obtains the following expression for the resolution

$$R = \frac{m}{\Delta m} = m\frac{\partial t}{\partial m} \Big/ \frac{\partial t}{\partial s}\Delta s \tag{5}$$

In the case of the single field straight masspectrometer $\Delta t_m = \frac{(2s+\ell)\Delta m}{2\sqrt{2mE_{acc}qs}}$. The expected resolution of the instrument is

$$R_{single} = \frac{(2s+\ell)s}{(\ell - 2s)\Delta s} \tag{6}$$

which results in a resolution of 66 at $\ell = 1.2$ m. It would not be meaningful to do experiments with such a poor instrument. It is important to notice that the resolution is independent of E_{acc} and q. Keeping Δs fixed the resolution will depend only on s and ℓ, $R = R(s,\ell)$. It is obvious from equation (6) that $\ell-2s = 0$, or equivalently $s = \ell/2$, would maximize the resolution.

It is thus, in principle, possible to build a straight time of flight masspectrometer with arbitrarily high resolution. The time resolution (10 ns) of the data acquisition system and the length of the ionizing laser pulse (also around 10 ns) puts a limit on how compact a spectrometer it is possible to build. One has to consider how short a flight tube one can make and still be able to resolve two different masses. The difference in flight time, Δt_m, between a cluster of mass m and one with mass m+Δm must not be shorter than say 30 ns. The shortest possible flight length, ℓ, is then $\ell \geq -\varepsilon + \sqrt{\varepsilon^2 + \frac{2\Delta t_m^2 mqU}{(\Delta m)^2}}$. The field strength has been expressed as $E_{acc}=U/(\ell/2+\varepsilon)$. The shortest possible flight length is 600 mm, taking U= 3 kV, ε= 10 mm, m = 720 (C_{60}) and $\Delta m = 1$. The total length of the masspectrometer would be 900 mm of which the field should be unrealizable 300 mm.

So far nothing has been said about the relative orientation of the flight tube to the molecular beam in such a case. Normally but not necessarily there is an angle often around 90°, to quickly accelerate the constituents out of the beam, between the axes. In our device this angle is close to 90°. This also means that the initial drift velocity must be compensated for either by a lens or a deflection field or by simply allowing for a certain drift depending on the combined velocity from acceleration in the electric field and the initial speed (essentially common to all sizes of clusters) of the beam. All three ways have been tested. It is important to notice that to first order this drift velocity effects neither the flight time nor the mass resolution but is at the same time the most serious limit as to why the ideal one-field TOF spectrometer cannot be realized.

There are several ways to improve the situation when using a straight flight tube. Either to reduce Δs or increase ℓ or s. It is unattractive to increase s, since it is difficult to obtain a homogeneous electric field over a large distance, in particular since the ion beam is not restricted to the center of the tube as will be the case with a perpendicular tube arrangement. Ideally one would like to find a way of focusing the ion beam within the acceleration region to compensate for the initial drift velocity. Unfortunately this is difficult in the perpendicular arrangement and puts a practical lower limit to the field strength. The physical dimensions of the laboratory also restricts the length of the flight tube, so the most straight forward way to improve R is to reduce Δs. It has turned out that Δs = 0.5 mm is a good compromise for ordinary beam cluster densities.

Dual field acceleration. The idea to divide the accelerating field in two parts, one weak and one strong, comes from the fact that the effect of Δs is reduced in this configuration. The strong field is E_s ($=U_s/d$) and the weak field is E_w. Typically the field strength of the strong field is E_s= 150 kV/m. The time the ionized clusters spend in the two fields can be calculated from (2).

The final velocity of the ions after the dual field acceleration is

$$v_{dual}=\sqrt{\frac{2q\left\{sE_w + U_s\right\}}{m}} \tag{7}$$

The flight time of the ions can be expressed as

$$t = \sqrt{\frac{2m}{q}}\cdot\left\{\sqrt{\frac{s}{E_w}} + \frac{\sqrt{U_s+sE_w}-\sqrt{sE_w}}{E_s} + \frac{\ell}{2\sqrt{U_s+sE_w}}\right\} \tag{8}$$

Equation (5) gives a complicated expression for $R_{dual} = R_{dual}(E_w,E_s,s,d,\ell)$. In practice the parameters s, d and ℓ are fixed and can not be changed easily. As a numerical example E_{strong}= 75 kV/m gives E_{weak}= 7.46 kV/m and E_{strong}= 150 kV/m gives E_{weak}= 14.94 kV/m.

A similar treatment as in the description of a single field gives a spread in t as a function of s. The width, Δt, is now 160 ns, considerably smaller than in the single field case. In principle there is no strong field influence on Δt so the accuracy depends solely on the weak field. The concept has a better resolution for practical values of s than the single field device. Here it is possible to achieve R = 200 with the present apparatus.

The single field reflectron. The resolution can be further improved if a reflectron is included in the TOF-MS. This device is an electrostatic mirror and the idea is that clusters with higher velocities will plunge deeper into the electric field and thus spend more time in the reflectron. All clusters will eventually reach the detector at the same time. The electric field of the mirror must however be homogeneous to perform in this way. One way to define the field is to use thin mesh grids in several steps. The transmission of the mesh is at the best 80%. The ion intensity is therefore seriously reduced by the reflectron in such a case.

How long a time the ions will spend in the electrostatic mirror depends on the angle of the reflectron. The impact velocity into the reflecting field is $v_{refl} = \sqrt{\frac{2qsE_{acc}}{m}}$ and the time spent in the reflectron is expressed as

$$t_{refl} = \frac{2mv_{refl}\cos\alpha}{qE_{refl}} \tag{9}$$

The total flight time of the ions can be expressed with (1) and (3), taking $v_E = 0$, as

$$t = \sqrt{\frac{2m}{q}} \cdot \left\{ \sqrt{\frac{s}{E_{acc}}} + \frac{(\ell + \ell_r)}{2\sqrt{sE_{acc}}} + \frac{2\cos\alpha\sqrt{sE_{acc}}}{E_{refl}} \right\} \tag{10}$$

where E_{refl} is the field strength in the reflectron, α is half of the angle of the reflectron and ℓ_r the length of the second path of the reflectron. There is one restriction on E_{refl}: The field must be deep enough to allow the ions to be reflected. After acceleration the ions have accumulated a kinetic energy of qsE_{acc}. The depth of the reflectron field, i, must be such that $sE_{acc}\cos^2\alpha \leq iE_{refl}$.

Equation (5) leads to the following possible solution for maximum resolution $E_{refl} = \frac{4s\cos\alpha}{\ell + \ell_r - 2s} \cdot E_{acc}$. Typical values of the parameters result in $E_{refl} = 5.9$ kV/m ($s = 30$ mm, $\alpha = 4°$, $E_{acc} = 75$ kV/m, $\ell = 944$ mm, $\ell_r = 646$ mm). The minimum depth of the field can be calculated as $i \geq \frac{(\ell + \ell_r - 2s)\cos\alpha}{4}$ which gives $i \geq 38$ cm. In contrast to what has often been assumed there is no first order dependence of the resolution upon the angle of the reflectron.

One problem in connection with reflectrons as pointed out above is how to make the electrical field uniform. One possibility is to use large diameter intermediate rings electrically connected via resistors. The central field will then hopefully be essentially unaffected by the outside grounded tube and then of course we must restrict the ion beam to the center of the reflectron. Furthermore in contrast to the situation with the acceleration field above the reflectron field is passed twice - in and out. So that if the trace is symmetric the device will be self compensating. The performance can be further improved by introducing a strong retarding field prior to the reflectron. The idea is to let relatively slow ions into the electrostatic mirror which will make the design much more compact.

The dual field reflectron. The idea of the dual field reflectron is to reduce the kinetic energies of the ions in a retarding voltage. The impact velocity into the reflecting field is $v_{refl} = \sqrt{\frac{2q(sE_{acc} - U_{ret})}{m}}$, where U_{ret} is the retarding voltage. The total flight time is

$$t = \sqrt{\frac{2m}{q}} \cdot \left\{ \sqrt{\frac{s}{E_{acc}}} + \frac{(\ell + \ell_r)}{2\sqrt{sE_{acc}}} + \frac{2\cos\alpha\sqrt{sE_{acc} - U_{ret}}}{E_{refl}} \right\} \qquad (11)$$

It is obvious that (11) is valid only if $sE_{acc} > U_{ret}$. Equation (5) implies that $E_{refl} = \frac{4(sE_{acc})^{3/2}\cos\alpha}{(\ell + \ell_r - 2s)\sqrt{sE_{acc} - U_{ret}}}$ maximizes the resolution. Assuming the same parameters as before and $U_{ret} = 1.5$ kV one obtains $E_{refl} = 7.4$ kV/m. The condition for the depth of the field is now $i \geq \frac{sE_{acc} - U_{ret}}{E_{refl}}$ which leads to $i \geq 20$ cm. The introduction of the retarding field leads to a more compact reflectron as desired.

The retardation field is provided by 16 plates, 10 mm apart. The bottom plate is solid and the upper ones have 30x15 mm holes. No grids are necessary because of the symmetry mentioned above. Usually wirenettings are used here, but these will not just limit the transmission but also spoil the homogeneity of the field around the wires. Often this effect is neglected since the overall effect will be rather nicer parallel field lines (necessary in the acceleration stage of the TOF spectrometer). The only disadvantage of the open structure if not properly adjusted is then a focusing effect. The mass resolution can be increased to 1000:1 with this device.

The ability to resolve masses is important in two cases. During reaction experiments on heavy clusters mass resolution is crucial to determine reaction products. When doing R2PI spectroscopy on clusters the different isotopomers provide important information and the use of a reflectron arrangement is more or less necessary.

As mentioned above it is necessary to compensate for the initial velocity component of the neutral beam. This is done by two deflector plates at the beginning of the flight tube. As an example: In order to estimate the required voltage over the deflector plates it is necessary to calculate the angle β between the flight path of the ions and the center of the flight tube. It can be calculated as $\tan\beta = v_{carr}/v_{ion}$.

Table 2 Ion velocities, required deflector voltages and other properties for argon carrier.

Species	V_{acc} (km/s)	β (°)	U_{defl} (V)	m_s	m_l
C	190	0.2	16	-	200
Na	137	0.2	22	-	240
Ti_2	67	0.5	46	-	420
Cu_2	58	0.6	53	-	483
Zr_2	49	0.7	63	8	586
C_{60}	25	1.3	125	261	1409

The time the ions spend in the deflecting field, E_{defl}, is $t_{defl} = b/v_{ion}$ where b is the width of the field (b = 50 mm) and v_{ion} the velocity of the ions. The offset, caused by the drift velocity relative the optical axis of the flight tube, at the detector should of course be zero for the specific mass, m_0, under consideration. The necessary field strength can be calculated as

$$E_{defl} = \frac{2\sqrt{2}\cdot v_{carr}(b+\ell_f+\ell_d+2s)}{b(b+2\ell_f)}\cdot\sqrt{\frac{m_0 s E_{acc}}{q}} \qquad (12)$$

where ℓ_f is the length of the flight path and ℓ_d is the distance from the accelerating field to the deflector plates. The required field strength depend on the velocity of the beam, v_{carr}, and the mass of the cluster, m_0, since the other parameters are fixed. Table 2 summarizes required deflector voltages for different species. The off axis position, r, on the detector plate for a particle of mass m (other than m_0) can then be expressed as

$$r = \frac{v_{carr}(b+\ell_f+\ell_d+2s)}{\sqrt{2qsE_{acc}}} \cdot \left\{\sqrt{m}-\sqrt{m_0}\right\} \tag{13}$$

Since the diameter of a channel plate is 18 mm it is obvious that $-9\text{ mm} < r < 9\text{ mm}$. The smallest detectable mass is $m_s = \left\{\sqrt{m_0} - \frac{r}{k}\right\}^2$ and the largest $m_l = \left\{\sqrt{m_0} + \frac{r}{k}\right\}^2$, where $k = \frac{v_{carr}(b+\ell_f+\ell_d+2s)}{\sqrt{2qsE_{acc}}}$. The effective mass window is therefore $m_l-m_s = 4r\sqrt{m_0}/k$. Some mass limits are calculated in table 2.

One has to be aware of that the mass distribution very much depends on the voltage over the plates. The reason is that heavy ions require a high voltage to enter the flight tube, a voltage that completely deflects lighter species. It should be noted that only parts of the mass spectrum can be observed with a fixed voltage. The deflector voltage has the advantage of preventing rest gas (pump oil) ions from reaching the detector since they have no drift velocity.

7 Timing

The timing control of the system is provided by a four channel pulse generator (Stanford Research Systems, Inc. Model DG 535). The channels can be individually programmed relative to each other or to the main frequency with an accuracy better than 1 ns. The maximum frequency at which the apparatus can be operated is 10 Hz which is the maximum frequency of the vaporizing laser. Another limit is the repetition rate of the valve, which is believed to be around 15 Hz. The frequency of the whole experiment is consequently 10 Hz. The timing cycle for R2PI experiments (argon carrier) is summarized in table 3.

Table 3. Timing cycle.

Time	Event
t_0	Valve electronics trigger
t_0+400 μs	Vaporization laser trigger[1]
t_0+820 μs	Pump laser flash lamps trigger[2]
t_0+1000 μs	Ionization laser trigger[3]
t_0+1000 μs	Pump laser Pockels cell trigger[4]

[1] The relatively long time delay between the trigger pulses to to valve and to the vaporization laser is due to mechanical inertia of the valve and the time it takes for the carrier gas pulse to reach the rod.

[2] Pump laser flash lamp trigger. The delay 180 μs between the flash lamp and the Pockels cell maximizes the output of the laser.

[3] Ionization laser trigger. The delay is a direct measure of the velocity of the carrier.

[4] Pump laser Pockels cell trigger. The offset is due to different response times of the YAG and excimer lasers and on different path lengths for the light.

Valve electronics: The main frequency gives the start pulse to the nozzle drive electronics. The solenoid currents are controlled by three parameters: open and close pulse and the delay between them. These three times are set by flip flops externally adjustable with potentiometers. The relations between the settings affect the length and shape of the carrier pulse.

Vaporization laser: The vaporization laser has two firing modes, internal and external. The firing cycle can be divided into two parts. Firstly the flash lamp is discharged and secondly the Pockels cell is activated (after 200 μs). The delay is internally provided by a simple analog timer and thus the time jitter can be quite large but since the carrier gas pulse is so extended in time it is of no importance and that makes the triggering of the laser very simple.

Pump laser: Very high stability in the time domain is also required (typically 2 ns) when the pump laser of the tunable laser is operated. It is therefore fired externally. Two trigger pulses are required, one to fire the flash lamps and one to open the pockels cell.

Ionization laser: The ionization laser (excimer) can be triggered with enough accuracy with one pulse. This signal also activates the data acquisition system, which consists of two parts one digital oscilloscope and one AT computer.

8. Data acquisition

The signal from the detector must be digitized in some way allowing it to be analyzed. The A/D converter must be such that its sampling rate is high enough to enable it to catch events on the sub micro second scale since the difference between flight times is of that order. We use a fast digital oscilloscope (LeCroy 9400A) for this purpose. This oscilloscope has several powerful features. It has a sampling rate of 100 Ms/s which is quite adequate, it is possible to perform averaging of the signal internally and it has long (32 kB) memories which allow high resolution at long scans.

The A/D conversion is performed with 8 bits, which is unsatisfactory when making averaging of weak signals. It is therefore better to transfer the digitized one shot time scan to an external computer and do the averaging there. This procedure requires a sufficiently fast data transfer speed. For example if the experiment has a duty cycle of 10 Hz and the time scan consists of 2000 channels the system must be able to digitize and to transfer and receive 2 kB of data in 100 ms. It is obvious that the serial interface can not be used in this case instead we use a standard parallel GPIB interface between the oscilloscope and the computer.

The computer software is written in TurboPascal and is divided into three parts: TOFMS, SCANMS and LIFEMS. The TOFMS (App. 1) program is used when a mass spectrum is recorded. It reads the oscilloscope and averages for as many events as are necessary to build up the spectrum. The SCANMS (App. 2) program is used when doing R2PI spectroscopy. It transfers data from the oscilloscope to the computer and controls the scanning of the dye laser. It is possible to define a number of signal windows, which is very useful since spectra of different isotopomers can be recorded simultaneously. The widths of the signal windows are an important parameters which could affect the quality of the recordings. These widths can be set individually. The program also includes a calibration routine which is intended to work together with an uranium lamp using the optogalvanic effect. The LIFEMS (App. 3) program is used for lifetime measurements. It controls the programmable pulse generator and scans the delay time between the lasers.

Acknowledgments

This work was supported in part by the National Swedish Board for Technical Development (STU) and the Knut and Alice Wallenberg Foundation. The authors are grateful to prof. H. Kroto and prof. R. E. Smalley for valuable information and support. The authors also would like to express their gratitude to eng. K. Schmidt and techn. S. Hubenette and K. Hörnfeldt for manufacturing the equipment.

References

[1] R. E. Smalley, Laser Chem. 2 (1983).

[2] M. Doverstål, B. Lindgren, U. Sassenberg and H. Yu, Z. Phys. D 19 (1991).

[3] M. Doverstål, B. Lindgren, U. Sassenberg and H. Yu, Physica Scripta 43 (1991).

[4] M. Doverstål, B. Lindgren, U. Sassenberg and H. Yu, Chem. Phys. Let. 192 (1992) 2,3.

[5] M. Doverstål, B. Lindgren, U. Sassenberg, C. A. Arrington and M. D. Morse, J. Chem. Phys. 97 (1992) 10.

[6] M. Doverstål and P. Weijnitz, Molecular Physics 75 (1992) 6.

[7] L. R. Anderson, S. Maruyama and R. E. Smalley, Chem. Phys. Let. 176 (1991) 348.

[8] Methods of experimental physics vol. 7A, Academic press 1968.

Appendix 1

```
{$n+}
PROGRAM tofms;
uses TPDECL,CRT,GRAPH;

const MaxPointsD=3050;
      MaxCmdLength=25;
      Loop=true;
      PortA=$77C;
      PortB=$77D;
      PortD=$77F;
type Vector=array[1..MaxPointsD] of single;       {Data acq. vector}
     VectorR=array[1..MaxPointsD] of byte;        {Scope read vector}
     CMD_LENGTH=array[1..MaxCmdLength] of char;

var ResVec1:Vector;
    RD:VectorR;
    CMD:array[1..5] OF CMD_LENGTH;
    ReadCmd:CMD_LENGTH;
    DevName:NBUF;
    FactorN,FactorI:real;
    COUNT,Dev,Y,NShots,NChan,BChan,L,IV,FV:integer;
    Ch1,Ch2,Ch3,Ch4,Ch5,Ch6:char;
    DyeAnswer,InPosition:byte;

function OffSet:real;
var n,CODE:integer;
    L:real;
    ANSWER:array[1..15] OF CHAR;
    QUEST:array[1..6] OF CHAR;
    PART_OF_ANSWER:array[1..5] OF CHAR;
    S:string[5];
begin
  QUEST:='C1OF ?';
   IBWRT(Dev,QUEST,6);                          {Write QUEST to LECROY}
     IBRD(Dev,ANSWER,15);                       {Read ANSWER}
      for n:=1 to 5 do PART_OF_ANSWER[n]:=ANSWER[n+6];
     S:=PART_OF_ANSWER;
    VAL(S,L,CODE);
   OffSet:=L;
end;  {END OffSet}

procedure ClearVecD(var A:Vector;B:integer);
var n:integer;
begin
  for n:=1 to B do A[n]:=0;
end;  {END ClearVecD}

procedure ComputeFactors;                       {CALLS FUNCTION OffSet}
begin
  FactorN:=OffSet*32+128;
  FactorI:=OffSet*32-128;
end;  {END ComputeFactors}

procedure DeclareConstants;
begin
  port[PortD]:=$82;                             {SET A AS OUTPUT, B AS INPUT}
   DevName:='LECROY ';                          {SELECT LECROY}
    Dev:=IBFIND(DevName);
     CMD[3]:='CFMT,A,BYTE               ';
      ReadCmd[1]:='R';
       ReadCmd[2]:='E';
        ReadCmd[3]:='A';
         ReadCmd[4]:='D';
         ReadCmd[5]:=',';
        ReadCmd[8]:='.';
       ReadCmd[9]:='D';
      ReadCmd[10]:='A';
     ReadCmd[11]:=',';
    ReadCmd[12]:='1';
   ReadCmd[13]:=',';
  ReadCmd[18]:=',';
end;   {END DeclareConstants}

procedure InitializeDevice;
begin
   IBTMO(Dev,12);                              {SET TIMEOUT TO 3 SECONDS}
    IBEOT(Dev,1);
    ibwrt(Dev,cmd[3],11);
end;   {END InitializeDevice}

procedure MakeDataCmd;
var U:string[5];
    N:integer;
begin
{$I-}
repeat
  write('(N)ormal (=channel 1)   (I)nverted (=function F) acquisition: ');
   Ch5:=ReadKey;
  writeln(Ch5);
until (Ch5='N') OR (Ch5='I');
repeat
  write('How many shots?: ');
  readln(NShots);
until IORESULT=0;
repeat
  write('Number of points?: ');
  readln(NChan);
until (NChan<MaxPointsD) and (IORESULT=0);
  str(NChan:4,U);
  for N:=14 to 17 do ReadCmd[N]:=U[N-13];
repeat
  write('Begin at channel?: ');
  readln(BChan);
until IORESULT=0;
  str(BChan:4,U);
   for N:=19 to 22 do ReadCmd[N]:=U[N-18];
  if Ch5='N' then
begin                                          {NORMAL ACQUISITION}
  ReadCmd[6]:='C';
  ReadCmd[7]:='1';
end else
begin                                          {INVERTED ACQUISITION}
  ReadCmd[6]:='F';
  ReadCmd[7]:='F';
end;
{$I+}                                          {ENABLE I/O CHECKING}
end;   {END MakeDataCmd}

procedure MakeScopeCmd;
var U:string[5];
    N:integer;
begin
{$I-}                                          {DISABLE I/O CHECKING}
repeat
  write('Number of points?: ');
  readln(NChan);
until (NChan<MaxPointsD) AND (IORESULT=0);
  STR(NChan:4,U);
  for N:=14 to 17 do ReadCmd[N]:=U[N-13];
repeat
  write('Begin at channel?: ');
  readln(BChan);
{$I+}                                             {ENABLE I/O CHECKING}
until IORESULT=0;
  STR(BChan:4,U);
   for N:=19 to 22 do ReadCmd[N]:=U[N-18];
    writeln('C - Memory C');
     writeln('D - Memory D');
      writeln('E - Function E');
     writeln('F - Function F');
    write('Scope source?: ');
   Ch6:=ReadKey;
  writeln(Ch6);
  iv:=BChan;
  fv:=NChan;
CASE Ch6 OF
  'C':
begin
  ReadCmd[6]:='M';
  ReadCmd[7]:='C';
end;
  'D':
begin
  ReadCmd[6]:='M';
  ReadCmd[7]:='D';
end;
  'E':
begin
  ReadCmd[6]:='F';
  ReadCmd[7]:='E';
end;
  'F':
begin
  ReadCmd[6]:='F';
  ReadCmd[7]:='F';
end;
end;
end;

procedure ModeSelect;
begin
  CLRSCR;
repeat
  write('(D)ata acq.  (O)ld spectrum  (R)ead scope  (Q)uit: ');
   Ch2:=ReadKey;
  writeln(Ch2);
until (Ch2='D') or (Ch2='Q') or (Ch2='R') or (Ch2='O');
end;  {END ModeSelect}

procedure NormSpec(A:integer);  {A IS THE NUMBER OF POINTS TO BE NORMALIZED}
var n:integer;
    C:real;
begin
  C:=ABS(ResVec1[1]);
   for n:=2 to A do if C<ABS(ResVec1[n]) then C:=ABS(ResVec1[n]);
  for n:=1 to A do ResVec1[n]:=ResVec1[n]/C;
end;  {END NormSpec}

procedure PlotSpec(VAR V:VECTOR;NELEMENT:INTEGER);

var Number:string[10];
    DUM,SV,EV,Min,Max,YScale,XSCALE,INCREMENT,STEP,IIV,IIIV,FFV,Z:REAL;
    GD,GM,K,I,X,Y,XI,XF,YI,YF,KK1,K1,K2,NNELEMENT:integer;
    FUNCKEY,ESCAPE:boolean;
    ARROW:CHAR;
    PIXELX:array[1..400] OF integer;
    PIXELY:array[1..30] OF integer;

BEGIN
  GD:=DETECT;
   INITGRAPH(GD,GM,'\TUPA\BGI\');
    ESCAPE:=FALSE;
   K1:=1;
    K2:=NELEMENT;
    FFV:=FV;
   IIV:=IV;
  NNELEMENT:=NELEMENT-1;
REPEAT
  NNELEMENT:=K2-K1;
   MAX:=-1000;
   MIN:=+1000;
   FOR K:=K1 TO K2 DO
   BEGIN
    IF V[K]<MIN THEN MIN:=V[K];
    IF V[K]>MAX THEN MAX:=V[K];
   END;
   IF MAX=MIN THEN YSCALE:=300/MAX ELSE YSCALE:=300/(MAX-MIN);
   INCREMENT:=(FFV-IIV)/NNELEMENT;         {DISPERSION - ANTAL PIXELAVSTND/CM-1}
    XSCALE:=550/(FFV-IIV);
     STEP:=1;
      IF XSCALE < 10 THEN STEP:=(FFV-IIV)/55;
       SV:=INT(IIV+0.999999);
       I:=40+ROUND((SV-IIV)*XSCALE);
      STR(SV:5:0,Number);
     OUTTEXTXY((I-10),340,Number);
    DUM:=0;
   X:=I;
  LINE(X,330,X,339);
REPEAT
  LINE(X,330,X,335);
   DUM:=DUM+STEP;
  X:=I+ROUND(DUM*XSCALE);
until X>590;
  DUM:=DUM-STEP;
   X:=I+ROUND(DUM*XSCALE);
    LINE(X,330,X,339);
     EV:=INT(FFV);
      STR(EV:5:0,Number);
      OUTTEXTXY((X-10),340,Number);
     LINE(40,330,590,330);
    XF:=40;
   YF:=330-ROUND(YScale*(V[K1]-Min));
  for K:=(K1+1) TO K2 do
BEGIN
  XI:=XF;
   YI:=YF;
    XF:=40+ROUND((K-K1)*INCREMENT*XSCALE);
   YF:=330-ROUND(YScale*(V[K]-Min));
  LINE(XI,YI,XF,YF);
END;
  OUTTEXTXY(430,0,'POSITION:');
   OUTTEXTXY(430,10,'INTENSITY :');
    SETFILLSTYLE(1,0);
    SETTEXTSTYLE(0,0,0);
   X:=300;
  Y:=100;
```

```
REPEAT
  FUNCKEY:=TRUE;
   for K:=Y-10 TO 330 do PIXELX[K]:=GETPIXEL(X,K);
    for K:=1 TO 21 do PIXELY[K]:=GETPIXEL(X-11+K,Y);
     Z:=(X-40)/XSCALE+IIV;
      STR(Z:5:2,Number);
       OUTTEXTXY(550,0,Number);
        STR((330-Y)/30:2:1,Number);
       OUTTEXTXY(550,10,Number);
      LINE(X,Y-10,X,330);
     LINE(X-10,Y,X+10,Y);
    ARROW:=READKEY;
   IF ARROW=#27 THEN ESCAPE:=TRUE;
  IF ARROW=#0 THEN
BEGIN
  ARROW:=READKEY;
   for K:=Y-10 TO 330 do PUTPIXEL(X,K,PIXELX[K]);
   for K:=1 TO 21 do PUTPIXEL(X-11+K,Y,PIXELY[K]);
  BAR(549,0,640,19);
CASE ARROW OF
  #73:dec(Y);
  #75:dec(X);
  #77:inc(X);
  #81:inc(Y);
  #82:
BEGIN
  KK1:=K1+ROUND((X-40)/550*NNELEMENT);
  IIIV:=Z;
END;
  #83:
BEGIN
  K2:=K1+ROUND((X-40)/550*NNELEMENT);
   FFV:=Z;
    IIV:=IIIV;
   FUNCKEY:=FALSE;
  K1:=KK1;
END;
  #115:X:=X-10;
  #116:X:=X+10;
  #118:Y:=Y+10;
  #132:Y:=Y-10;
  #71:
BEGIN
  K1:=1;
   K2:=NELEMENT;
    FFV:=FV;
   IIV:=IV;
  FUNCKEY:=FALSE;
END;
END;
END;
  until (FUNCKEY=FALSE) OR (ESCAPE=TRUE);
 CLEARDEVICE;
 until ESCAPE=TRUE;
   CLOSEGRAPH;
END;

procedure ProcVec;
var n:integer;
    Term:real;
begin
  if Ch5='N' then Term:=-NShots*FactorN else Term:=NShots*FactorI;
  for n:=1 to NChan do ResVec1[n]:=ResVec1[n]+Term;
end;   {end ProcVec}
```

```
procedure ReadSpec;
var InFile:string[30];
    F:text;
    C:integer;
begin
{$I-}
repeat
  write('Name of file: ');
   readln(InFile);
   assign(F,InFile);
  reset(F);
until IORESULT=0;
  NChan:=1;
while not eof(F) do
begin
  read(F,C);
   readln(F,ResVec1[NChan]);
  inc(NChan);
end;
  dec(NChan);
  close(F);
{$I+}
end;   {end ReadSpec}

procedure SaveSpec(A:integer;B:Vector);         {A IS THE NUMBER OF POINTS TO SAVE}
var n:integer;
    OutFile:string[30];
    F:text;
    Ch:char;
begin
{$I-}
repeat
  write('Save the spectrum? (Y/N): ');
   Ch:=ReadKey;
  writeln(Ch);
until (Ch='Y') OR (Ch='N');
  if Ch='Y' then
begin
repeat
  write('Name the output file: ');
   readln(OutFile);
  assign(F,OutFile);
  rewrite(F);
until ioresult=0;
  for n:=1 to A-1 do writeln(F,n,' ',B[n]);
   write(F,A,' ',B[A]);
  close(F);
end;
{$I+}
end;   {end SaveSpec}

procedure SmoothSpec;
var n:integer;
    P1,P2:real;
begin
repeat
  write('Smooth spectrum? (Y/N): ');
   Ch3:=ReadKey;
  writeln(Ch3);
until (Ch3='Y') OR (Ch3='N');
  if Ch3='Y' then
begin
  n:=2;
```

```
   P1:=ResVec1[n-1];
  WHILE n<NChan do
begin
  P2:=ResVec1[n];
   ResVec1[n]:=P1/4+ResVec1[n]/2+ResVec1[n+1]/4;
    inc(n);
    P1:=ResVec1[n];
   ResVec1[n]:=P2/4+ResVec1[n]/2+ResVec1[n+1]/4;
  inc(n);
end;
end;
end;   {end SmoothSpec}

procedure ReadAndAdd;
var M,N:integer;
begin
  for M:=1 to NShots do
begin
  IBWRT(Dev,ReadCmd,22);                        {SEND READ COMMAND TO LECROY}
   IBRD(Dev,RD,NChan+8);                 {READ NChan BYTES FROM DEVICE TO RD}
  for N:=1 to NChan do ResVec1[N]:=ResVec1[N]+RD[N+4];
end;
end;   {END ReadAndAdd}

procedure ReadScope;
var N:integer;
begin
  IBWRT(Dev,ReadCmd,22);                        {SEND READ COMMAND TO LECROY}
   IBRD(Dev,RD,NChan+8);                  {READ NChan BYTES FROM LECROY TO RD}
  for N:=1 to NChan do ResVec1[N]:=ResVec1[N]+RD[N+4];
end;   {END ReadScope}

begin                                           {MAIN PROGRAM}
  DeclareConstants;
repeat                                          {ETERNAL LOOP}
  ModeSelect;
case Ch2 of
  'R':
begin
  MakeScopeCmd;
   InitializeDevice;
    ClearVecD(ResVec1,NChan);
     ReadScope;
    PlotSpec(ResVec1,NChan-4);
   SaveSpec(NChan,ResVec1);
  IBONL(Dev,0);                                 {Release LECROY}
end;
  'O':
begin                                           {OLD SPECTRUM PROGRAM}
  ReadSpec;
   IV:=0;                                       {Initial x value for plot}
    FV:=Nchan-1;                                {Final x value for plot}
    PlotSpec(ResVec1,NChan);
   SmoothSpec;
  if Ch3='Y' then
begin
  PlotSpec(ResVec1,NChan);
  SaveSpec(NChan,ResVec1);
end;
end;                                            {END OLD SPECTRUM PROGRAM}
  'Q':halt;
  'D':

{***************************** DATA SECTION ********************************}
```

```
begin
  MakeDataCmd;
   InitializeDevice;
    ComputeFactors;                                  {COMPUTE OFFSET}
     IV:=Bchan;                                      {Initial x value for plot}
      FV:=Bchan+Nchan-1;                             {Final x value for plot}
       ClearVecD(ResVec1,NChan);                     {RESET RESULT VECTOR}
        ReadAndAdd;                                  {ADDING LOOP}
       ProcVec;
     PlotSpec(ResVec1,NChan-4);
    SaveSpec(NChan,ResVec1);
   SmoothSpec;
  if Ch3='Y' then
begin
  PlotSpec(ResVec1,NChan-4);
  SaveSpec(NChan,ResVec1);
end;
end;                                                 {END DATA ACQUISITION PROG.}

end;   {END CASE}
IBLOC(Dev);                                          {LECROY to local mode}
until Loop=false;                                    {LOOP NEVER FALSE}
end.
```

Appendix 2

```
{$N+}
PROGRAM scanms;
uses dos,TPDECL,CRT,GRAPH;

const MaxPoints=3500;
      MaxCmdLength=25;
      Loop=true;
      PortA=$30B;
      PortB=$303;

type Vector=array[1..MaxPoints] of single;     {Data acq. vector}
     VectorR=array[1..MaxPoints] of byte;
     CMD_LENGTH=array[1..MaxCmdLength] of char;

var ResVec1,ResVec2,ResVec3,PeakVec:Vector;
    RD:VectorR;
    CMD3,ReadCmd:CMD_LENGTH;
    DevName:NBUF;
    Fv,Iv,FactorN,FactorI:real;
    Dev,Y,NShots,NChan,BChan,AShots,L,LoSig1,LoSig2,LoSig3,
    HiSig1,HiSig2,HiSig3,Code:integer;
    Ch1,Ch2,Ch3,Ch4,Ch5,Ch6,Ch7,Ch9:char;
    DyeAnswer,InPosition:byte;
    S1:array[1..6] of char;
    S:string[8];
    Buffer:string[30];
    Regs:registers;

function IntegratePeak(A,B:integer):real;
var n:integer;
    Sum:real;
begin
  Sum:=0;
   for n:=A to B do Sum:=Sum+PeakVec[n];
  IntegratePeak:=Sum;
end;  {END IntegratePeak}

function OffSet:real;
var n,CODE:integer;
    L:real;
    ANSWER:array[1..15] OF CHAR;
    QUEST:array[1..6] OF CHAR;
    PART_OF_ANSWER:array[1..5] OF CHAR;
    S:string[5];
begin
  QUEST:='C1OF ?';
   IBWRT(Dev,QUEST,6);
    IBRD(Dev,ANSWER,15);
     for n:=1 to 5 do PART_OF_ANSWER[n]:=ANSWER[n+6];
    S:=PART_OF_ANSWER;
   VAL(S,L,CODE);
  OffSet:=L;
end;  {END OffSet}

procedure ClearVec(var A:Vector;B:integer);
var n:integer;
begin
  for n:=1 to B do A[n]:=0;
end;  {END ClearVecD}

procedure ComputeFactors;                         {CALLS FUNCTION OffSet}
begin
  FactorN:=OffSet*32+128;
  FactorI:=OffSet*32-128;
```

```
end;   (END ComputeFactors)

procedure DeclareConstants;
begin
   DevName:='LECROY ';                              (SELECT GPIB-BOARD #0)
    Dev:=IBFIND(DevName);
     CMD3:='CFMT,A,BYTE              ';
      ReadCmd[1]:='R';
       ReadCmd[2]:='E';
        ReadCmd[3]:='A';
         ReadCmd[4]:='D';
         ReadCmd[5]:=',';
        ReadCmd[8]:='.';
       ReadCmd[9]:='D';
      ReadCmd[10]:='A';
     ReadCmd[11]:=',';
    ReadCmd[12]:='1';
   ReadCmd[13]:=',';
  ReadCmd[18]:=',';
end;   (END DeclareConstants)

procedure InitializeDevice;
var s:array[1..7] of char;
begin
  IBTMO(Dev,12);                                    (SET TIMEOUT TO 3 SECONDS)
   IBEOT(Dev,1);
    IBWRT(Dev,CMD3,11);                          (SELECT OUTPUT FORMAT FOR DATA)
   s:='CAL OFF';
  ibwrt(Dev,s,7);
end;   (END InitializeDevice)

procedure INTERRUPT;
BEGIN
Repeat
  If Length(Buffer)<25 then Buffer:=Buffer+#0;
Until Length(Buffer)>=25;
  With Regs do
Begin
  DS:=Seg(Buffer);    {  Set the DS and      }
   DX:=Ofs(Buffer);    {  DX register Pair  }
  INTR($62,Regs);
End;               { Interrupt called and completed, now get the response }
  S:=COPY(BUFFER,4,8);       { from the buffer.}
END;                         { Peal off the bytes and display them }

procedure MakeCaliberCmd;
var U:string[5];
    N:integer;
begin
{$I-}
repeat
  write('How many shots?: ');
  readln(NShots);
until IORESULT=0;
repeat
  write('How many shots per average?: ');
  readln(AShots);
until IORESULT=0;
repeat
  write('Signal window from: ');
   readln(LoSig1);
   write('to: ');
  readln(HiSig1);
until IORESULT=0;
```

```
repeat
  write('Background window from: ');
   readln(LoSig2);
   write('to: ');
  readln(HiSig2);
until IORESULT=0;

begin                                              {AQC. WITHOUT REFERENCE}
  NChan:=HiSig2-LoSig1+1;
   BChan:=LoSig1;
    LoSig1:=LoSig1-BChan+1;
     LoSig2:=LoSig2-BChan+1;
    HiSig1:=HiSig1-BChan+1;
   HiSig2:=HiSig2-BChan+1;
end;                                               {END AQC. WITHOUT REFERENCE}
  str(NChan:4,U);
   for N:=14 to 17 do ReadCmd[N]:=U[N-13];
    STR(BChan:4,U);
    for N:=19 to 22 do ReadCmd[N]:=U[N-18];
   ReadCmd[6]:='C';                                {NORMAL ACQUISITION}
  ReadCmd[7]:='2';
{$I+}
end;   {END MakeCaliberCmd}

procedure MakePeakCmd;
var U:string[5];
    N:integer;
begin
{$I-}
repeat
  write('How many shots?: ');
  readln(NShots);
until IORESULT=0;
repeat
  write('How many shots per average?: ');
  readln(AShots);
until IORESULT=0;
repeat
  write('Signal window #1 from: ');
   readln(LoSig1);
   write('to: ');
  readln(HiSig1);
until IORESULT=0;
repeat
  write('Signal window #2 from: ');
   readln(LoSig2);
   write('to: ');
  readln(HiSig2);
until IORESULT=0;
repeat
  write('Signal window #3 from: ');
   readln(LoSig3);
   write('to: ');
  readln(HiSig3);
until IORESULT=0;
begin                                              {AQC. WITHOUT REFERENCE}
  NChan:=HiSig3-LoSig1+1;
   BChan:=LoSig1;
    LoSig1:=LoSig1-BChan+1;
     LoSig2:=LoSig2-BChan+1;
     LoSig3:=LoSig3-BChan+1;
    HiSig1:=HiSig1-BChan+1;
   HiSig2:=HiSig2-BChan+1;
  HiSig3:=HiSig3-BChan+1;
```

```
end;                                          {END AQC. WITHOUT REFERENCE}
  str(NChan:4,U);
   for N:=14 to 17 do ReadCmd[N]:=U[N-13];
    STR(BChan:4,U);
    for N:=19 to 22 do ReadCmd[N]:=U[N-18];
   ReadCmd[6]:='C';                           {NORMAL ACQUISITION}
  ReadCmd[7]:='1';
{$I+}
end;  {END MakePeakCmd}

procedure ModeSelect;
begin
  CLRSCR;
repeat
  write('(P)eak acq.  (C)alib.  (O)ld spectrum   (Q)uit: ');
   Ch2:=ReadKey;
  writeln(Ch2);
until (Ch2='Q') or (Ch2='O') or (Ch2='P') or (Ch2='C');
end;  {END ModeSelect}

procedure NormSpec(A:integer);  {A IS THE NUMBER OF POINTS TO BE NORMALIZED}
var n:integer;
    C:real;
begin
  C:=ABS(ResVec1[1]);
   for n:=2 to A do if C<ABS(ResVec1[n]) then C:=ABS(ResVec1[n]);
  for n:=1 to A do ResVec1[n]:=ResVec1[n]/C;
end;  {END NormSpec}

procedure PlotSpec(VAR V:VECTOR;NELEMENT:INTEGER);

var Number:string[10];
    DUM,SV,EV,Min,Max,YScale,XSCALE,INCREMENT,STEP,IIV,IIIV,FFV,Z:REAL;
    GD,GM,K,I,X,Y,XI,XF,YI,YF,KK1,K1,K2,NNELEMENT:integer;
    FUNCKEY,ESCAPE:boolean;
    ARROW:CHAR;
    PIXELX:array[1..400] OF integer;
    PIXELY:array[1..30] OF integer;
begin
  GD:=DETECT;
   INITGRAPH(GD,GM,'\TUPA\BGI\');
    SETBKCOLOR(1);
     SETCOLOR(14);
      ESCAPE:=FALSE;
     K:=1;
    Max:=V[1];
   Min:=V[1];
  FOR K:=2 TO NELEMENT DO
begin
  IF V[K]<Min THEN Min:=V[K];
  IF V[K]>Max THEN Max:=V[K];
END;
  if Max=Min then YScale:=300/Max else YScale:=300/(Max-Min);
   K1:=1;
    K2:=NELEMENT;
    FFV:=FV;
   IIV:=IV;
  NNELEMENT:=NELEMENT-1;
 repeat
  NNELEMENT:=K2-K1;
   INCREMENT:=(FFV-IIV)/NNELEMENT;         {DISPERSION - ANTAL PIXELAVSTND/CM-1}
    XSCALE:=550/(FFV-IIV);
     STEP:=1;
      IF XSCALE < 10 THEN STEP:=(FFV-IIV)/55;
```

```
        SV:=INT(IIV+0.999999);
        I:=40+ROUND((SV-IIV)*XSCALE);
       STR(SV:5:0,Number);
      OUTTEXTXY((I-10),340,Number);
     DUM:=0;
    X:=I;
   LINE(X,330,X,339);
repeat
   LINE(X,330,X,335);
    DUM:=DUM+STEP;
   X:=I+ROUND(DUM*XSCALE);
until X>590;
   DUM:=DUM-STEP;
    X:=I+ROUND(DUM*XSCALE);
     LINE(X,330,X,339);
      EV:=INT(FFV);
       STR(EV:5:0,Number);
       OUTTEXTXY((X-10),340,Number);
      LINE(40,330,590,330);
     XF:=40;
    YF:=330-ROUND(YScale*(V[K1]-Min));
   for K:=(K1+1) TO K2 do
begin
   XI:=XF;
    YI:=YF;
     XF:=40+ROUND((K-K1)*INCREMENT*XSCALE);
    YF:=330-ROUND(YScale*(V[K]-Min));
   LINE(XI,YI,XF,YF);
END;
   OUTTEXTXY(430,0,'POSITION:');
    OUTTEXTXY(430,10,'INTENSITY :');
     SETFILLSTYLE(1,0);
     SETTEXTSTYLE(0,0,0);
    X:=300;
   Y:=100;
repeat
   FUNCKEY:=TRUE;
   for K:=Y-10 TO 330 do PIXELX[K]:=GETPIXEL(X,K);
    for K:=1 TO 21 do PIXELY[K]:=GETPIXEL(X-11+K,Y);
     Z:=(X-40)/XSCALE+IIV;
     STR(Z:5:2,Number);
      OUTTEXTXY(550,0,Number);
       STR((330-Y)/30:2:1,Number);
        OUTTEXTXY(550,10,Number);
       LINE(X,Y-10,X,330);
      LINE(X-10,Y,X+10,Y);
     ARROW:=READKEY;
    IF ARROW=#27 THEN ESCAPE:=TRUE;
   IF ARROW=#0 THEN
begin
   ARROW:=READKEY;
    for K:=Y-10 TO 330 do PUTPIXEL(X,K,PIXELX[K]);
    for K:=1 TO 21 do PUTPIXEL(X-11+K,Y,PIXELY[K]);
   BAR(549,0,640,19);
CASE ARROW OF
   #73:dec(Y);
   #75:dec(X);
   #77:inc(X);
   #81:inc(Y);
   #82:
begin
   KK1:=K1+ROUND((X-40)/550*NNELEMENT);
   IIIV:=Z;
END;
```

```
  #83:
begin
  K2:=K1+ROUND((X-40)/550*NNELEMENT);
   FFV:=2;
    IIV:=IIIV;
   FUNCKEY:=FALSE;
  K1:=KK1;
END;
  #115:X:=X-10;
  #116:X:=X+10;
  #118:Y:=Y+10;
  #132:Y:=Y-10;
  #71:
begin
  K1:=1;
   K2:=NELEMENT;
    FFV:=FV;
   IIV:=IV;
  FUNCKEY:=FALSE;
END;
END;
END;
until (FUNCKEY=FALSE) OR (ESCAPE=TRUE);
  CLEARDEVICE;
until ESCAPE=TRUE;
  CLOSEGRAPH;
END;

procedure ProcVec;
var n:integer;
    Term:real;
begin
  if Ch5='N' then Term:=-NShots*FactorN else Term:=NShots*FactorI;
  for n:=1 to NChan do ResVec1[n]:=ResVec1[n]+Term;
end;  {end ProcVec}

procedure ReadSpec;
var InFile:string[30];
    F:text;
    C:integer;
begin
{$I-}
repeat
  write('Name of file: ');
   readln(InFile);
   assign(F,InFile);
  reset(F);
until IORESULT=0;
  NChan:=1;
while not eof(F) do
begin
  read(F,C);
   readln(F,ResVec1[NChan]);
  inc(NChan);
end;
  dec(NChan);
  close(F);
{$I+}
end;  {end ReadSpec}

procedure SaveSpec(A:integer;B:Vector);        {A IS THE NUMBER OF POINTS TO SAVE}
var n:integer;
    OutFile:string[30];
    F:text;
    Ch:char;
begin
{$I-}
repeat
  write('Save the spectrum? (Y/N): ');
   Ch:=ReadKey;
  writeln(Ch);
until (Ch='Y') OR (Ch='N');
  if Ch='Y' then
begin
repeat
  write('Name the output file: ');
   readln(OutFile);
  assign(F,OutFile);
  rewrite(F);
until ioresult=0;
  for n:=1 to A-1 do writeln(F,n,' ',B[n]);
   write(F,A,' ',B[A]);
  close(F);
end;
{$I+}
end;   {end SaveSpec}

procedure SmoothSpec;
var n:integer;
    P1,P2:real;
begin
repeat
  write('Smooth spectrum? (Y/N): ');
   Ch3:=ReadKey;
  writeln(Ch3);
until (Ch3='Y') OR (Ch3='N');
  if Ch3='Y' then
begin
  n:=2;
   P1:=ResVec1[n-1];
  WHILE n<NChan do
begin
  P2:=ResVec1[n];
   ResVec1[n]:=P1/4+ResVec1[n]/2+ResVec1[n+1]/4;
    inc(n);
    P1:=ResVec1[n];
   ResVec1[n]:=P2/4+ResVec1[n]/2+ResVec1[n+1]/4;
  inc(n);
end;
end;
end;   {end SmoothSpec}

procedure ReadAndAdd;
var M,N:integer;
begin
  for M:=1 to NShots do
begin
  IBWRT(Dev,ReadCmd,22);                          {SEND READ COMMAND TO DEVICE}
   IBRD(Dev,RD,NChan+8);                  {READ NChan BYTES FROM DEVICE TO RD}
  for N:=1 to NChan do Res.ec1[N]:=ResVec1[N]+RD[N+4];
end;
end;   {END ReadAndAdd}

procedure ReadAAndAdd;
var M,N:integer;
    A1:array[1..50] of double;
    a1a:double;
    OS:REAL;
```

```
begin
  for M:=1 to AShots do
begin                                              {AVERAGE LOOP}
  IBWRT(Dev,ReadCmd,22);                           {SEND READ CMD TO DEVICE}
   IBRD(Dev,RD,NChan+8);  {READ NChan BYTES FROM DEVICE TO RD}
    A1[m]:=FACTORN-RD[5];
end;
A1a:=0;
     for n:=1 to AShots do  A1a:=A1a+A1[n];
A1a:=A1a/AShots;
for n:=1 to AShots do
begin
  if abs(A1[n]-A1a)<0.375*abs(a1a) then peakvec[1]:=peakvec[1]+a1[n];
end;                                               {END AVERAGE LOOP}
end;   {END ReadAAndAdd}

begin                                              {MAIN PROGRAM}
  TextBackground(1);
   TextColor(14);
  DeclareConstants;
repeat                                             {ETERNAL LOOP}
  ModeSelect;
case Ch2 of
  'O':
begin                                              {OLD SPECTRUM PROGRAM}
  ReadSpec;
   write('Start position: ');
    readln(IV);
     write('End position: ');
     readln(FV);
    PlotSpec(ResVec1,NChan);
   SmoothSpec;
  if Ch3='Y' then
begin
  PlotSpec(ResVec1,NChan);
  SaveSpec(NChan,ResVec1);
end;
end;                                               {END OLD SPECTRUM PROGRAM}
  'Q':halt;
  'C':
{***************************** Caliber SECTION ******************************}
begin
  Buffer:='G';          {SCAN command to HD500}
   Interrupt;
    MakeCaliberCmd;
     InitializeDevice;
    ComputeFactors;                               {COMPUTE OFFSET}
   ClearVec(ResVec1,NShots);
{***************************** Caliber FUNCTION *****************************}
  for L:=1 to NShots do
 begin
  ClearVec(PeakVec,NChan);
  port[PortA]:=$FF;
repeat
  delay(90);
   DyeAnswer:=port[PortB];
  InPosition:=DyeAnswer;
until InPosition<128;
  DyeAnswer:=port[PortB];
   ReadAAndAdd;
    for Y:=LoSig1 to HiSig2 do PeakVec[Y]:=PeakVec[Y]-AShots*FactorN;
   ResVec1[L]:=IntegratePeak(LoSig1,HiSig1);
  port[PortA]:=0;
 end;
```

```
{************************* END Caliber FUNCTION ***************************}
  write('Start position: ');
   readln(IV);
    write('End position: ');
     readln(FV);
     PlotSpec(ResVec1,NShots);
    SaveSpec(NShots,ResVec1);
   SmoothSpec;
  if Ch3='Y' then
begin
  PlotSpec(ResVec1,NShots);
  SaveSpec(NShots,ResVec1);
end;
end;

  'P':

{***************************** PEAK SECTION *******************************}

begin
  Buffer:='G';           {SCAN command to HD 500}
   Interrupt;
    MakePeakCmd;
     InitializeDevice;
      ComputeFactors;                           {COMPUTE OFFSET}
      ClearVec(ResVec1,NShots);
     Buffer:='0';
    Interrupt;
   val(s,Iv,Code);

{****************************** PEAK FUNCTION *****************************}

  for L:=1 to NShots do
begin
  ClearVec(PeakVec,NChan);
  port[PortA]:=$FF;
repeat
  DELAY(80);
  DyeAnswer:=port[PortB];
  InPosition:=DyeAnswer;
until InPosition<128;
  DyeAnswer:=port[PortB];
   ReadAAndAdd;
    for Y:=LoSig1 to HiSig3 do PeakVec[Y]:=PeakVec[Y];
     ResVec1[L]:=IntegratePeak(LoSig1,HiSig1);
    ResVec2[L]:=IntegratePeak(LoSig2,HiSig2);
   ResVec3[L]:=IntegratePeak(LoSig3,HiSig3);
  port[PortA]:=0;
end;

{************************** END PEAK FUNCTION *****************************}

  Buffer:='0';
   Interrupt;
    val(s,Fv,Code);
     PlotSpec(ResVec1,NShots);
      PlotSpec(ResVec2,NShots);
{******************* SELECT NUMBER OF SPECTRA TO PLOT *********************
       PlotSpec(ResVec3,NShots);
        PlotSpec(NShots,MarkVec);
********************************* END *************************************}
       SaveSpec(NShots,ResVec1);
      SaveSpec(NShots,ResVec2);
{********************* SELECT NUMBER OF SPECTRA TO SAVE *******************
```

```
     SaveSpec(NShots,ResVec3);
    SaveSpec(NShots,MarkVec);
********************************* END *************************************}
   SmoothSpec;
  if Ch3='Y' then
begin
  PlotSpec(ResVec1,NShots);
  SaveSpec(NShots,ResVec1);
end;
end;
end;   {END CASE}
  s1:='CAL ON';
   ibwrt(Dev,s,6);
  ibloc(Dev);                                        {RELEASE DEVICE}
until Loop=false;                                  {LOOP NEVER FALSE}
end.
```

Appendix 3

```
{$N+}
PROGRAM lifems;
uses TPDECL,CRT,GRAPH;

const MaxPoints=1000;
      MaxCmdLength=25;
      Loop=true;

type Vector=array[1..MaxPoints] of single;       {Data acq. vector}
     VectorR=array[1..MaxPoints] of byte;        {Scope read vector}
     CMD_LENGTH=array[1..MaxCmdLength] of char;

var ResVec1,PeakVec:Vector;
    RD:VectorR;
    v:array[1..100] of real;
    CMD:array[1..5] OF CMD_LENGTH;
    ReadCmd,WriteCmd:CMD_LENGTH;
    PCTime:string[6];
    DevName:NBUF;
    FactorN,FactorI:double;
    n,Dev1,Dev2,Y,NShots,NChan,BChan,AShots,L,LoSig1,
    PCStart,PCStop,PCStep,HiSig1,NResVec,NStep,NN:integer;
    Ch1,Ch2,Ch3,Ch4,Ch5,Ch6,Ch7,Ch9:char;
    Sum,Sum1,Fv,Iv:real;

function IntegratePeak(A,B:integer):double;
var n:integer;
    Sum:double;
begin
  Sum:=0;
   for n:=A to B do Sum:=Sum+PeakVec[n];
  IntegratePeak:=Sum;
end;  {END IntegratePeak}

function OffSet:double;
var n,CODE:integer;
    L:double;
    ANSWER:array[1..15] OF CHAR;
    QUEST:array[1..6] OF CHAR;
    PART_OF_ANSWER:array[1..5] OF CHAR;
    S:string[5];
begin
  QUEST:='C1OF ?';
   IBWRT(Dev1,QUEST,6);
    IBRD(Dev1,ANSWER,15);
     for n:=1 to 5 do PART_OF_ANSWER[n]:=ANSWER[n+6];
    S:=PART_OF_ANSWER;
   VAL(S,L,CODE);
  OffSet:=L;
end;  {END OffSet}

procedure ClearVec(var A:Vector;B:integer);
var n:integer;
begin
  for n:=1 to B do A[n]:=0;
end;  {END ClearVec}

procedure ComputeFactors;                        {CALLS FUNCTION OffSet}
oegin
  FactorN:=OffSet*32+128;
  FactorI:=OffSet*32-128;
end;  {END ComputeFactors}

procedure DeclareConstants;
```

```
begin
  Devname:='LECROY ';                            {SELECT LECROY}
   Dev1:=IBFIND(DevName);
    Devname:='DG535  ';                          {SELECT DG535}
     Dev2:=IBFIND(DevName);
      CMD[3]:='CFMT,A,BYTE            ';
       ReadCmd[1]:='R';
        ReadCmd[2]:='E';
         ReadCmd[3]:='A';
          ReadCmd[4]:='D';
           ReadCmd[5]:=',';
            ReadCmd[8]:='.';
             ReadCmd[9]:='D';
              ReadCmd[10]:='A';
             ReadCmd[11]:=';';
            ReadCmd[12]:='1';
           ReadCmd[13]:=',';
          WriteCmd[1]:='D';
         WriteCmd[2]:='T';
        WriteCmd[3]:='6';                           {Set D}
       WriteCmd[4]:=',';
      WriteCmd[5]:='3';                             {D=B+xxx}
     WriteCmd[6]:=',';
    WriteCmd[13]:='E';
   WriteCmd[14]:='-';
  WriteCmd[15]:='9';
end;   {END DeclareConstants}

procedure InitializeDevice;
begin
  IBTMO(Dev1,12);                                {SET TIMEOUT TO 3 SECONDS}
   ibtmo(Dev2,12);
    IBEOT(Dev1,1);
   ibeot(Dev2,1);
  IBWRT(Dev1,CMD[3],11);                           {SELECT OUTPUT FORMAT FOR DATA}
end;   {END InitializeDevice}

procedure MakeLifeCmd;
var U:string[5];
    N:integer;
begin
{$I-}
repeat
  write('How many shots per average?: ');
  readln(AShots);
until IORESULT=0;
repeat
  write('Signal window from: ');
   readln(LoSig1);
   write('to: ');
  readln(HiSig1);
until IORESULT=0;
repeat
  write('Scan P/C from (ns): ');
   readln(PCStart);
   write('to: ');
    readln(PCStop);
   write('step: ');
  readln(PCStep);
until IOResult=0;
begin                                            {AQC. WITHOUT REFERENCE}
  NChan:=HiSig1-LoSig1+1;
  NStep:=(PCStop-PCStart) div PCStep;
   BChan:=LoSig1;
   LoSig1:=LoSig1-BChan+1;
  HiSig1:=HiSig1-BChan+1;
end;                                              {END AQC. WITHOUT REFERENCE}
  str(NChan:4,U);
   for N:=14 to 17 do ReadCmd[N]:=U[N-13];
    ReadCmd[18]:=',';
     STR(BChan:4,U);
    for N:=19 to 22 do ReadCmd[N]:=U[N-18];
   ReadCmd[6]:='C';                               {NORMAL ACQUISITION}
  ReadCmd[7]:='1';
{$I+}
end;   {END MakeLifeCmd}

procedure ModeSelect;
begin
  CLRSCR;
repeat
  write('(L)ifetimes   (O)ld spectrum   (Q)uit: ');
   Ch2:=ReadKey;
  writeln(Ch2);
until (Ch2='Q') or (Ch2='O') or (Ch2='L');
end;   {END ModeSelect}

procedure PlotSpec(V:Vector;NElement:integer);
VAR Number:string[10];
    DUM,SV,EV,Min,Max,YScale,XScale,Increment,STEP:REAL;
    GD,GM,K,I,X,Y,XI,XF,YI,YF:integer;
    FUNCKEY:BOOLEAN;
    arrow:CHAR;
    PIXELX:array[1..400] OF integer;
    PIXELY:array[1..30] OF integer;
begin
  GD:=DETECT;
   INITGRAPH(GD,GM,'\tupa\bgi');
    Max:=V[1];
   Min:=V[1];
  FOR K:=2 TO NElement DO
begin
   IF V[K]<Min THEN Min:=V[K];
   IF V[K]>Max THEN Max:=V[K];
END;
  Increment:=(FV-IV)/(NElement-1);
   YScale:=300/(Max-Min);                 {DISPERSION - ANTAL PIXELAVSTND/CM-1}
    XScale:=550/(FV-IV);
     STEP:=1;
      IF XScale<10 THEN STEP:=(FV-IV)/55;
       SV:=INT(IV+0.999999);
       I:=40+round((SV-IV)*XScale);
      STR(SV:5:0,Number);
     OUTTEXTXY((I-10),340,Number);
    DUM:=0;
   X:=I;
  line(X,330,X,339);
REPEAT
  line(X,330,X,335);
   DUM:=DUM+STEP;
  X:=I+round(DUM*XScale);
UNTIL X>590;
  DUM:=DUM-STEP;
   X:=I+round(DUM*XScale);
    line(X,330,X,339);
     EV:=INT(FV);
      STR(EV:5:0,Number);
      OUTTEXTXY((X-10),340,Number);
     line(40,330,590,330);
    XF:=40;
   YF:=330-round(YScale*(V[1]-Min));
  for K:=2 TO NElement do
begin
  XI:=XF;
   YI:=YF;
    XF:=40+round((K-1)*Increment*XScale);
   YF:=330-round(YScale*(V[K]-Min));
  line(XI,YI,XF,YF);
END;
  OUTTEXTXY(430,0,'POSITION:');
   OUTTEXTXY(430,10,'INTENSITY :');
    SETFILLSTYLE(1,0);
    SETTEXTSTYLE(0,0,0);
   X:=300;
  Y:=100;
REPEAT
  for K:=Y-10 TO 330 do PIXELX[K]:=GETPIXEL(X,K);
   for K:=1 TO 21 do PIXELY[K]:=GETPIXEL(X-11+K,Y);
    STR(((X-40)/XScale+IV):5:2,Number);
     OUTTEXTXY(550,0,Number);
      STR((330-Y)/30:2:1,Number);
       OUTTEXTXY(550,10,Number);
        line(X,Y-10,X,330);
        line(X-10,Y,X+10,Y);
       arrow:=READKEY;
      IF arrow<>#0 THEN FUNCKEY:=FALSE ELSE FUNCKEY:=TRUE;
     arrow:=READKEY;
    for K:=Y-10 TO 330 do PUTPIXEL(X,K,PIXELX[K]);
   for K:=1 TO 21 do PUTPIXEL(X-11+K,Y,PIXELY[K]);
  BAR(549,0,640,19);
CASE arrow OF
  #116:X:=X+10;
  #77:X:=X+1;
  #115:X:=X-10;
  #75:X:=X-1;
  #73:Y:=Y-1;
  #81:Y:=Y+1;
  #132:Y:=Y-10;
  #118:Y:=Y+10;
END;
  UNTIL FUNCKEY=FALSE;
  CLOSEGRAPH;
END;

procedure ReadSpec;
var InFile:string[30];
    F:text;
    C:integer;
begin
{$I-}
repeat
  write('Name of file: ');
   readln(InFile);
   assign(F,InFile);
  reset(F);
until IORESULT=0;
  NChan:=1;
while not eof(F) do
begin
  read(F,C);
   readln(F,ResVec1[NChan]);
  inc(NChan);
```

```
end;
  dec(NChan);
  close(F);
{$I+}
end;   {END ReadSpec}

procedure SaveSpec(A:integer;B:Vector);    {A IS THE NUMBER OF POINTS TO SAVE}
var n:integer;
    OutFile:string[30];
    F:text;
    Ch:char;
begin
{$I-}
repeat
  write('Save the spectrum? (Y/N): ');
   Ch:=ReadKey;
  writeln(Ch);
until (Ch='Y') OR (Ch='N');
  if Ch='Y' then
begin
repeat
  write('Name the output file: ');
   readln(OutFile);
  assign(F,OutFile);
  rewrite(F);
until ioresult=0;
  for n:=1 to A-1 do writeln(F,n,' ',B[n]);
   write(F,A,' ',B[A-1]);
  close(F);
end;
{$I+}
end;   {END SaveSpec}

procedure SmoothSpec;
var n:integer;
    P1,P2:double;
begin
repeat
  write('Smooth spectrum? (Y/N): ');
   Ch3:=ReadKey;
  writeln(Ch3);
until (Ch3='Y') OR (Ch3='N');
  if Ch3='Y' then
begin
  n:=2;
   P1:=ResVec1[n-1];
  WHILE n<NChan do
begin
  P2:=ResVec1[n];
   ResVec1[n]:=P1/4+ResVec1[n]/2+ResVec1[n+1]/4;
     inc(n);
     P1:=ResVec1[n];
   ResVec1[n]:=P2/4+ResVec1[n]/2+ResVec1[n+1]/4;
  inc(n);
end;
end;
end;   {END SmoothSpec}

procedure ReadAAndAdd;                          {ONLY ONE READOUT IN THE SUM}
var N:integer;
begin
  IBWRT(Dev1,ReadCmd,22);                          {SEND READ CMD TO DEVICE}
   IBRD(Dev1,RD,NChan+8);                   {READ NChan BYTES FROM DEVICE TO RD}
  for N:=1 to NChan do PeakVec[N]:=PeakVec[N]+RD[N+4];
```

```
end;   (END ReadAAndAdd)

begin                                                  (MAIN PROGRAM)
  DeclareConstants;
repeat                                                 (ETERNAL LOOP)
  ModeSelect;
case Ch2 of
  'O':
begin                                                  (OLD SPECTRUM PROGRAM)
  ReadSpec;
   PlotSpec(ResVec1,NChan);
   SmoothSpec;
  if Ch3='Y' then
begin
  PlotSpec(ResVec1,NChan);
  SaveSpec(NChan,ResVec1);
end;
end;                                                   (END OLD SPECTRUM PROGRAM)
  'Q':halt;

  'L':

(*************************** LIFETIME SECTION *******************************)

begin
  MakeLifeCmd;
   InitializeDevice;
    ComputeFactors;                                   (COMPUTE OFFSET)
   ClearVec(ResVec1,NStep);

(**************************** LIFETIME FUNCTION *****************************)

   L:=PCStart;
   NResVec:=1;
repeat
  ClearVec(PeakVec,NChan);
   str(L:6,PCTime);
    for NN:=7 to 12 do WriteCmd[NN]:=PCTime[NN-6];
     ibwrt(Dev2,WriteCmd,15);

     for n:=1 to 100 do               (100 IS THE NUMBER OF SHOTS TO ADJUST)
begin
      ReadAAndAdd;
     for Y:=LoSig1 to HiSig1 do PeakVec[Y]:=PeakVec[Y]-FactorN;
     V[n]:=IntegratePeak(LoSig1,HiSig1);
end;
     sum:=0;
     sum1:=0;
    for n:=1 to 100 do sum:=sum+v[n];
    sum:=sum/100;
    for n:=1 to 100 do if (-v[n]<-2*sum) and (-v[n]>0*sum) then
    sum1:=sum1+v[n];

    ResVec1[NResVec]:=sum1;
   L:=L+PCStep;
  inc(NResVec);
until L>PCStop;
  dec(NResVec);
   Iv:=PCStart;
    Fv:=PCStop;

(*************************** END LIFETIME FUNCTION ***************************)

     PlotSpec(ResVec1,NResVec);
    SaveSpec(NResVec,ResVec1);
   SmoothSpec;
  if Ch3='Y' then
begin
  PlotSpec(ResVec1,NResVec);
  SaveSpec(NResVec,ResVec1);
end;
end;
end;   {END CASE}
  IBLOC(Dev1);                                       {RELEASE DEVICEs}
  ibloc(Dev2);
until Loop=false;                                    {LOOP NEVER FALSE}
end.
```