A System for Control and Test of the ISOLDE Isotope Separator

Diploma Thesis
by
Petter Broman

Supervisor: Dr. H. Haas
Examiner: Prof. B. Jonson

Chalmers University of Technology
Department of Physics
Gothenburg
Sweden

and

CERN
EP-ISOLDE Group
Geneva
Switzerland

October 30, 1989

ABSTRACT

During three months of the summer 1989 I worked as a "summer student" at the researche laboratory CERN in Geneva. The ISOLDE isotope separator is the facility were I made my contribution. This report is the account for what I did. The work is intended for my diploma thesis.

The report is devided into three parts:

* Part 1 describes the ISOLDE separator and related physics.

* Part 2 presents two computerized systems for control and test of the separator. The first system adjusts the beam lines that transport the separated isotopes, and the second steers the separator magnet and measures how large are the intensities (in 10^{-10} Ampere) for the different separated isotopes.

* Part 3 tells about a stepmotor, used as a driver for a collection device, and the electronics that was necessary to use to make it run in a desired way.

ACKNOWLEDGEMENTS

I would like to thank my supervisor at ISOLDE Heinz Haas, for letting me do my diploma thesis at CERN, and for giving me interesting tasks to work with. I would also like to thank Helge Ravn for showing me the wonders of Excel, and Ove Jonsson for showing me his HT system.

During the months of September and October I have been writing this report at Chalmers. I thank Göran Nyman and Björn Jonson for using their time to help me find errors in the text. Björn Jonson is also the examiner for my diploma thesis.

Göteborg, October, 1989

CONTENTS

Part 1

ISOLDE AT CERN

1 INTRODUCTION

1.1 Location of ISOLDE

CERN (Conseil Européenne pour la Recherche Nucléaire), situated some 8 km northwest of the city of Geneva (see figure 1.1), is a laboratory for experimental high energy physics. The laboratory and its famous accelerators; e.g. the SPS (Super Proton Synchrotron) and the newly built LEP (Large Electron Positron Collider) are built accross the French-Swiss border. CERN is at present (1989) shared between 14 european member states.

ISOLDE (Isotope Separator On-Line) - an experimental station dealing mainly with nuclear science, is one of CERN's experimental facilities, located on the main area: Sité de Meyrin (see figure 1.1 and 1.2), on the Swiss side.

There are three main parts of ISOLDE: the 600 MeV Proton Synchro-Cyclotron (SC), the ISOLDE-2 and the ISOLDE-3 isotope separators. In figure 1.3 these are shown. The 600 MeV protons[1] are delivered from the right in the figure and do either interact with a target to create isotopes as atoms that are ionized in an ion-source and led above ground into the so called Proton Hall, the ISOLDE-3, or the protons are brought under ground to a similar set-up, the ISOLDE-2, which serves three floors of experimental equipement: xUR-8, UR-9 and UR-10 (from bottom to top).

Except for these buildings, there are also some offices, smaller laboratories, a work shop and the SC and ISOLDE-3 control rooms.[2] ISOLDE is a part of the Experimental Physics Division (CERN-EP).

1.2 A Brief History of Nuclear Science and ISOLDE

1.2.1 The Beginning

The first observed natural phenomenon in which the atomic nucleus manifested itself was in the radioactive decays. In 1895 W. C. Röntgen discovered X-rays and only one year after H. A. Becquerel found

[1]There are also other particles available, e.g. 910 MeV ^3He.
[2]The ISOLDE-2 control room is located in UR-9.

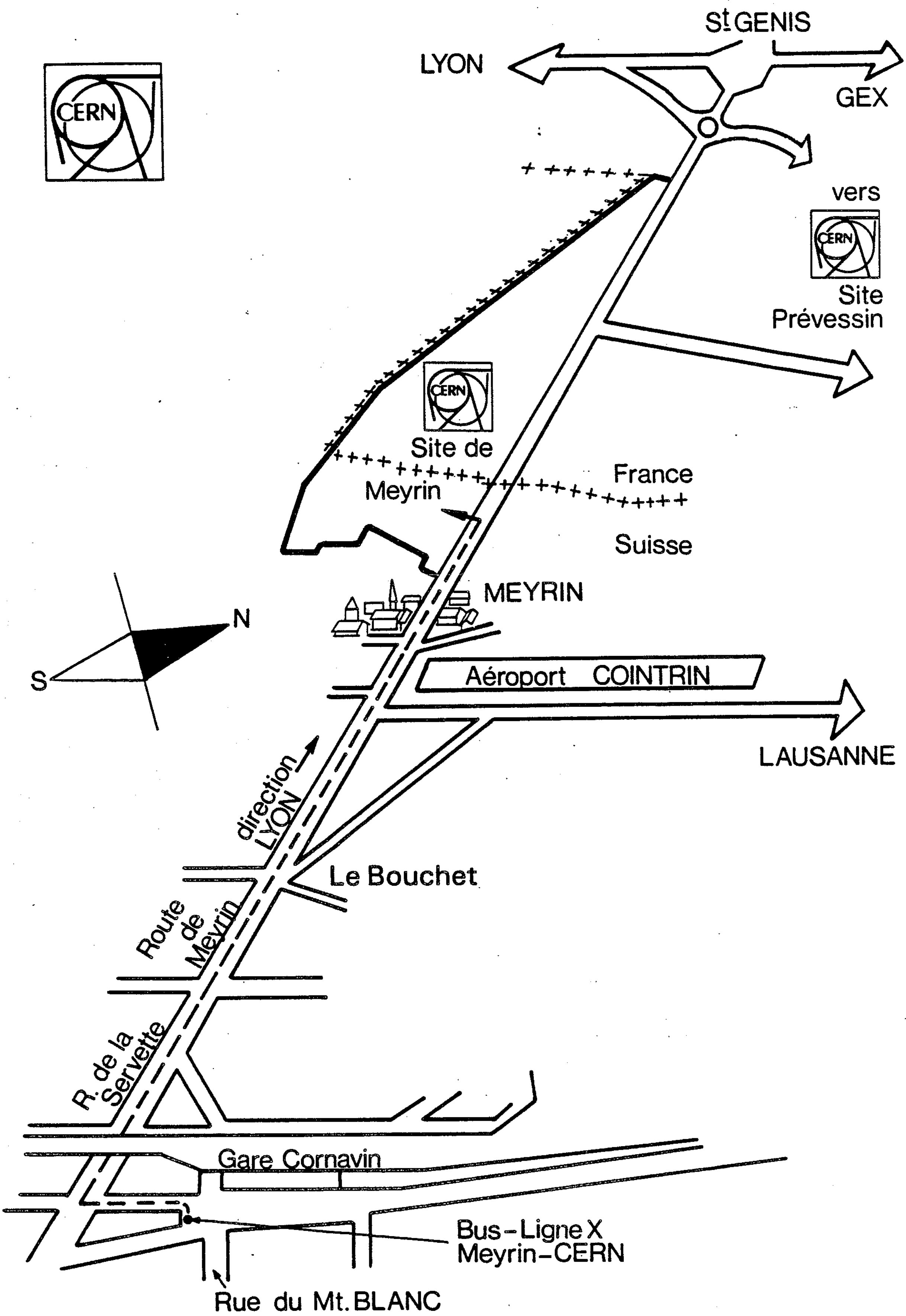


Fig 1.1 Location of CERN.

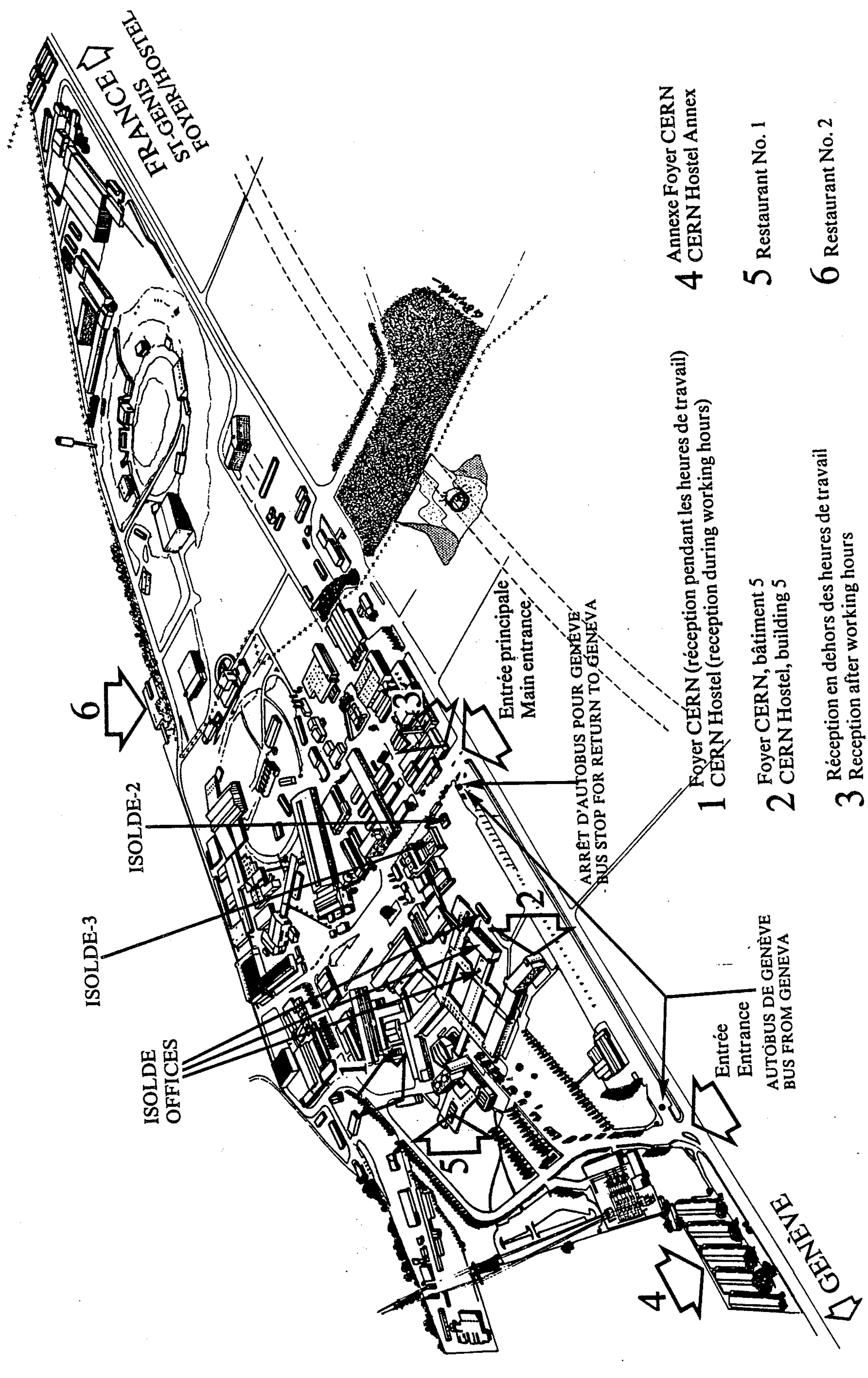


Fig 1.2 Location of ISOLDE.

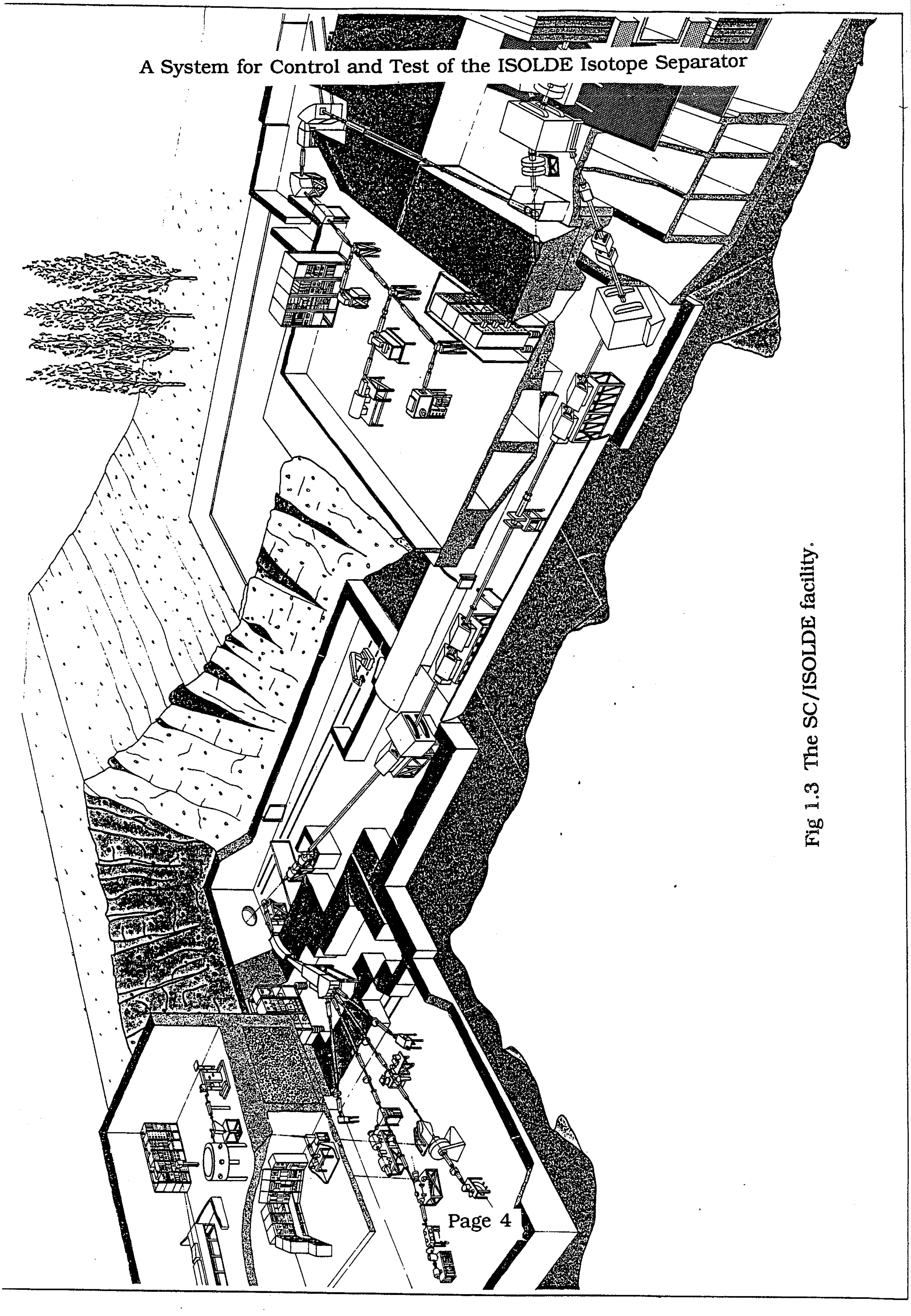


Fig 1.3 The SC/ISOLDE facility.

radiation from naturally occuring radioactivity. Lots of discoveries were made and in 1934 it was possible to demonstrate the occurence of artificial radioactivity.

1.2.2 The Chart of Nuclides

By now the knowledge was good enough to tell different properties about the nucleus. It is positively charged and located in the middle of the atom. The positive charge is due to the protons. The number of protons Z tells the chemical element that the atom belongs to. To account for the rest of the mass of the nucleus there are also the neutrons, without any charge, but with about the same mass as the protons. The number of protons Z together with the number of neutrons N are the only parameters that are necessary to distinguish one nucleus, i.e. a nuclide, from other nuclei. They can be plotted in a N-Z-diagram; the chart of nuclides, as in figure 1.4. In each grid point it is possible to place information about the various modes of decay, the half-lives and so on, for convenient reference.

The mass number A is defined as the total number of protons and neutrons in the nucleus, or simply Z+N. Nuclei with the same mass number are called isobars. They become diagonal lines in the chart of nuclides. In about the middle of these lines the elements are stable, but on both sides they tend to decay into this middle point through beta emission (electrons or positrons, with neutrinos, or electron capture).

Furthest away from the "line" joining the beta stable nuclides are the neutron and proton drip lines. There the binding energies are not big enough to bind another proton, in the case of the proton drip line, and another neutron, in the case of the neutron drip line.

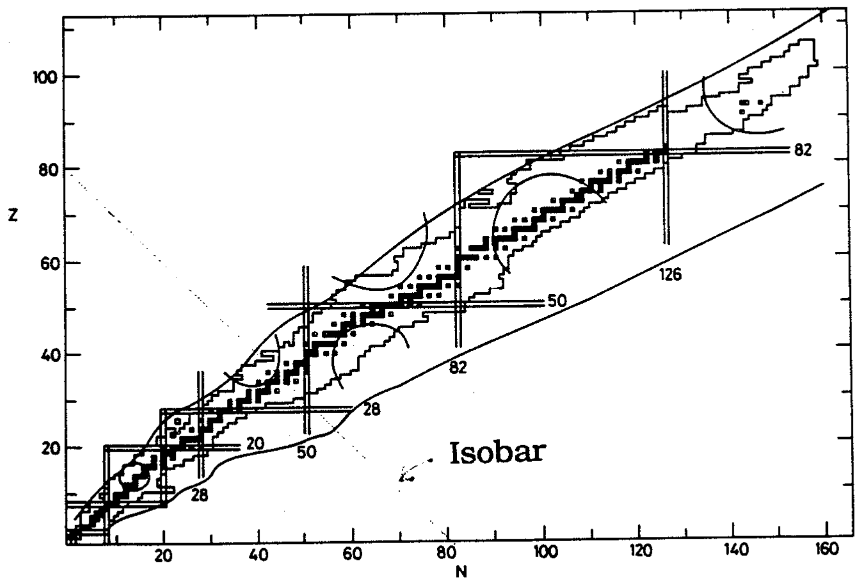


Fig. 1.4 The chart of nuclides. The stable isotopes are represented by the black squares. The region of known nuclides unstable against beta decay is enclosed by the broken contour. The diagonal smooth lines indicate the approximate limit of proton or neutron stability ($B_p \approx 0$ being the upper line; $B_n \approx 0$ being the lower line; B denoting nucleon binding energy). The circular lines are isodeformation curves showing the domain where nuclear deformation occurs.

1.2.3 Models

Radioactive studies in 1934-1955 led to the shell model proposed by Mayer and Jensen in 1949. According to this model a single nucleon (i.e. proton or neutron) is moving in a spherical symmetric potential regarded as an average over the other nucleons. This model explains the properties of nuclear levels, spin and parity. Bohr and Mottelson complemented this model with the collective model, describing how shape oscillations and rotations could occur due to the collective motion of many nucleons.

From experiments one found that nuclei near to the closed shells have spherical shape and vibrational spectra, while those far from the closed shells have deformed shape and rotational spectra.[1] Nilsson calculated the singel-particle states appropriate to the deformed potential. These investigations pointed out that the regions of nuclei far from stability are especially interesting. At the Lysekil conference in 1966 a number of physics questions where precisely formulated, and one experimental technique that seemed to make it possible to find the answers was that of on-line separation of ionic beams.

1.2.4 Separators

Soon after the application of accelerators around 1940, mass spectrometers[2] were constructed to identify mass numbers. In 1944 reactor-produced fission product mixtures with high specific activity became available for study. Dempster used a hot spark to ionize the fission products with an efficiency of 10^{-8}.

Separation of ^{235}U by war-time Calutrons gave the knowledge to construct laboratory machines for general pupose low-energy heavy ion acceleration with applications in many fields of physics and technology.

These techniques were off-line. They did not continuously deliver unstable nuclei created as secondary products from a target irradiated by some sort of primary particles. This so called on-line technique[3] together with the isotope separation was first tested by Kofoed-Hansen and Nielsen in 1950, using a target consisting of a mixture of uranium oxide and baking powder at room temperature. Thermal neutrons from the Copenhagen cyclotron were used as primary particles.

The on-line separation method was not drawing much attention in the fifties, mainly due to its technical problems, but by the advent of high resolution Ge and Li detectors in the mid-sixties, together with better accelerators, fast electronics and computers, the on-line separation technique was newly revised.

---

[1]These regions are marked in the chart of nuclides in figure 1.4.
[2]These mass spectrometers are actually mass separators, using accelerated ions bent in a magnet.
[3]The terms "off-line" and "on-line" were of cource invented later.

1.2.5 ISOLDE-1

In the autumn of 1961, the Nuclear Chemistry Group at CERN (later becoming the CERN ISOLDE Group) built an off-line electromagnetic mass separator, intended to complement radiochemical methods. But since an on-line separator would make it possible to detect short-lived nuclei, and thereby those in regions far from stability, and since both the theoretical and technical trend was supporting this, as mentioned above, it was decided in 1964 to build an on-line mass separator. It was ready for operation in the autumn of 1967 and used protons injected by the 600 MeV CERN Synchro-Cyclotron (SC), a facility which was already there - with all reasons for connecting it for on-line use.

The work with ISOLDE was from the beginning a cooperation between CERN and five other european countries: Denmark, France, Norway, Sweden and West Germany. They all contribute with ideas, manpower and money.

1.2.6 ISOLDE-2

In May 1973 the old ISOLDE installation, now known as ISOLDE-1, started to undergo a major reconstruction. The SC was modified at the same time. In late 1974 the new ISOLDE-2 was put into use, having a much higher extraction efficiency of the SC and a new technology for the important target and ion-source system. The beams and intensities were thereby greatly improved and the yields of separated nuclear products became higher.[1] New elements became available. The space for experiments had also been enlarged, and now not only strictly nuclear scientific experiments were possible but also applications in fields such as atomic, solid-state, and surface physics, elementary particle physics, and medicine.

1.2.7 ISOLDE-3

In April 1981 CERN's Scientific Policy Committee recommended a doubling of the scientific program at ISOLDE to 4000 hours. This needed a second facility. A plan for this; the ISOLDE-3, was finally acctepted in December 1984 by the CERN Directorate. The construction took place, and the first radioactive beam in ISOLDE-3 was separated in December1987. Beam optics and resolution was considerably improved with ISOLDE-3. Even isobars can be separated (up to around A=30), and singly charged ions are separated from doubly charged ions with twice the mass number.[2]

---

[1]The intensities became up to 10^{11} atoms per second.

[2]It is the ratio of charge to mass: q/m that decides the bending in the separating magnet. Thus e.g. $^{104}Cd^{+}$ and $^{208}Pb^{2+}$ come out together, but with a slight difference, due to the fact that mass and mass numbers are only approximately the same. This difference makes it possible to separate elements like $^{104}Cd^{+}$ and $^{208}Pb^{2+}$in ISOLDE-3 using its better resolution.

2 HOW ISOLDE WORKS

2.1 Introduction

ISOLDE is a "factory" producing pure isotope beams. It accomplishes the purfication, or with another term the "separation", both in A - the mass number, and under certain circumstances also in Z - the atomic number. The purification in Z is achieved by chemical means, through the choice of target and ion source system. Then the purification in A is brought out physically by bending the particles differently due to their charge to mass ratio in an applied variable magnetic field, and subsequent selection of those bent by a fixed angle.

The two facilities ISOLDE-2 and ISOLDE-3 work in parallel. ISOLDE-3 has the better resolution in mass and the higher yields, but ISOLDE-2 still seams to be easier to work with.

Both the separators are, as mentioned earlier, working "on-line". Radioactive isotopes are continuously created in the target by protons or lighter nuclei irradiating it (mainly ^3He). These in turn stem from the Synchro-Cyclotron. The use of these particles, together with a proper selection of target and ion source system, makes it possible to produce isotopes of elements - an advantage which has been utilized to produce a large amount of previously unknown nuclides. Different on-line separator systems data are displayed in table 2.1.

How ISOLDE works is depicted in figure 2.1.

Table 2.1

Essential parameters for comparing on-line separators

	Nuclear reactor	Heavy-ion accelerator	High-energy proton accelerator
Particle	n_{th}	HI	p
Energy	Thermal	~10 MeV/A	~600 MeV
Maximum available intensity (particles/s)	10^{15}	6×10^{12}	10^{15}
Nuclear reaction	Thermal fission of ^{235}U	(HI, $yp x n$)	(p, $yp x n$)
Cross-section (cm^2)	$<2 \times 10^{-23}$	$<2 \times 10^{-25}$	$<2 \times 10^{-26}$
Effective target thickness ($atoms/cm^2$)	1×10^{18}	10^{19}	6×10^{23}
Highest beam intensity used in an on-line separator experiment (particles/s)	10^{11}	10^{12}	10^{13}
Maximum production rates (atoms/s)	10^{8}	10^{6}	10^{11}

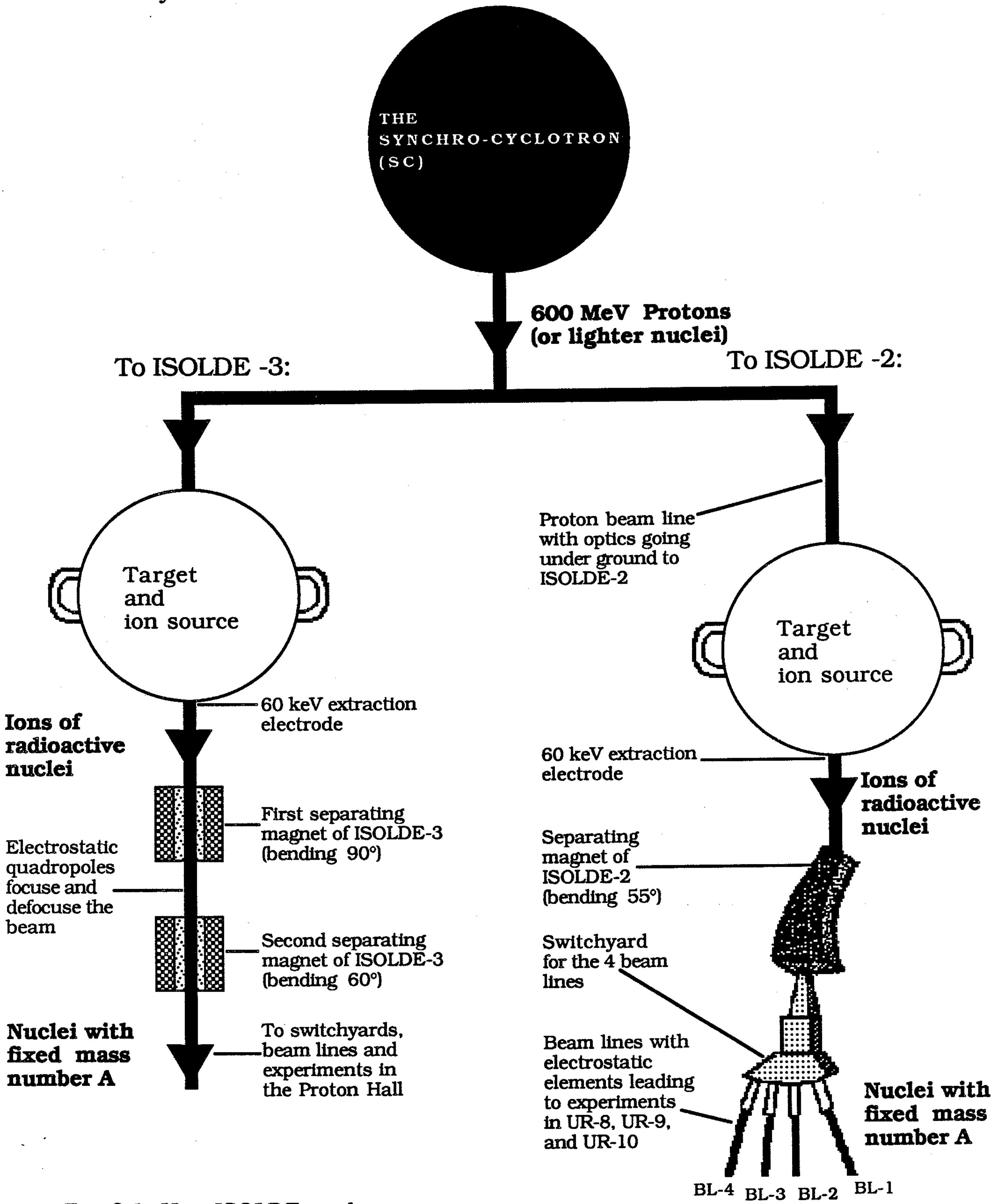


Fig. 2.1 How ISOLDE works.

2.2 The Synchro-Cyclotron

The Synchro-Cyclotron (SC) is a fixed-field resonance accelerator. It has got the following dimensions: 11 x 6.5 meters wide, 6 meters high, a pole diameter of 5 meters, and a pole gap of 36 to 45 centimeters. The total weight is 2600 metric tons. The electromagnet uses a power of 1.25 MW and is fed with a radiofrequency (RF) generated by two rotating condenser systems (ROTCO).

The machine is accelerating, except for protons, a number of light particles with charge to mass ratio from 1/3 to 1/4. Table 2.2 presents the available ions together with their intensities. The 600 MeV protons are the most important ones. Internally in the SC they have a maximum beam intensity of 7 μA, but with an extraction efficiency of 70%, they are ejected towards the two ISOLDE target areas with a maximum intensity of 5 μA .

The accelerated particles in the SC are created in a pulsed Penning ion source placed in the centre of the SC electromagnet. The lifetimes of these filaments are 150 hours, 50 hours, and 22 hours respectively for producing protons, helium, and carbon ions. It takes around one hour to change the filament.

The ISOLDE-2 target is located 60 meters from the SC and lies 8 meters underground, as shown in figure 1.3. ISOLDE-3 on the other hand is situated on both sides of the SC wall. This is shown in figure 2.16.

Table 2.2

List of the Synchro-cyclotron (SC) beams

Particle	Energy (MeV/N)	Extracted intensity (particles per second)	Frequency range (MHz)
Proton	602	$> 3 \times 10^{13}$	30.1-16.8
$^{3}He^{++}$	303	$> 3 \times 10^{12}$	20.3-13.9
$^{3}He^{+}$	85	$> 10^{13}$	10.1- 8.5
$^{12}C^{4+}$	85	$> 10^{12}$	10.1- 8.5
$^{15}N^{5+}$	85	$> 10^{11}$	10.1- 8.5
$^{18}O^{6+}$	85	$\sim 3 \times 10^{11}$	10.1- 8.5
$^{16}O^{6+}$	107	5×10^{9}	11.6- 9.4
$^{14}N^{5+}$	97	$\sim 10^{9}$	10.8- 9.0
$^{20}Ne^{7+}$	94	$\sim 5 \times 10^{9}$	10.6- 8.9
$^{20}Ne^{6+}$	70	$\sim 3 \times 10^{10}$	9.1- 7.7
$^{20}Ne^{5+}$	49	$\sim 3 \times 10^{11}$	7.6- 6.6
$^{12}C^{3+}$	49	$> 10^{12}$	7.6- 6.6

2.3 Target and Ion Source

2.3.1 Creation of Radioactive Nuclei

The creation of radioactive nuclei takes place when e.g. 600 MeV protons are colliding with the target material. Then, for these light bombarding projectiles with energies close to 1 GeV, there are three principal reactions taking place: spallation, fission and fragmentation. They are illustrated in figure 2.2.

2.3.2 Spallation

The spallation reaction occurs when the incomming particle more or less "knocks" off one or more nucleons from the target nucleus. Then in the next step, since the projectile also deposites kinetic energy, neutrons are evaporated from the nucleus.

The fact that the mass does not decrease that much; only relatively few nucleons are lost, makes it obvious that no lighter elements can be created with spallation reactions. Neutron deficient samples can on the other hand become produced.

The cross-section for the spallation reaction is about 20 mb near to stability and decreases towards regions far from stability. This is quite low, but the high penetrating power of 600 MeV protons, together with thick targets, still allow high production yields. A target thickness of 150 g/cm^2 and a mass around 140 corresponds to one interaction lenght for the ISOLDE protons, i.e. 67% of the protons are used up in the target. The production rate of the target can become up to 10^{11} atoms/s at cross-section 20 mb and incomming beam intensity of 1 μA.

Figure 2.3 shows cross-sections for elements produced in a spallation reaction with La as a target and 600 MeV protons as projectiles. The distribution can be described with a semi-empirical calculation using a formula of Rudstam.[1] For each element the curves show the dependence of the cross-section to the mass. They do also illustrate how the separation process is nececery to reduce the number of created particles to make elements like ^{114}Cs, with the very small cross-section of 1 nb, detectable. Otherwise they would disappear in the "soup" of produced nuclei.

If one uses ^{3}He, instead of the protons, interesting spallation yields are gained in regions far from stability.

2.3.3 Fission

The process when a heavy nucleus releases energy by falling apart into two new nuclei is called fission. Some neutrons may also become released as the rightmost stage in figure 2.2, under the tiltle "FISSION", displays. The fission process is induced, by the proton

[1]The curves for Te, Cs and Ba in the figure should be cut off, since the heaviest isobars are impossible to form at all.

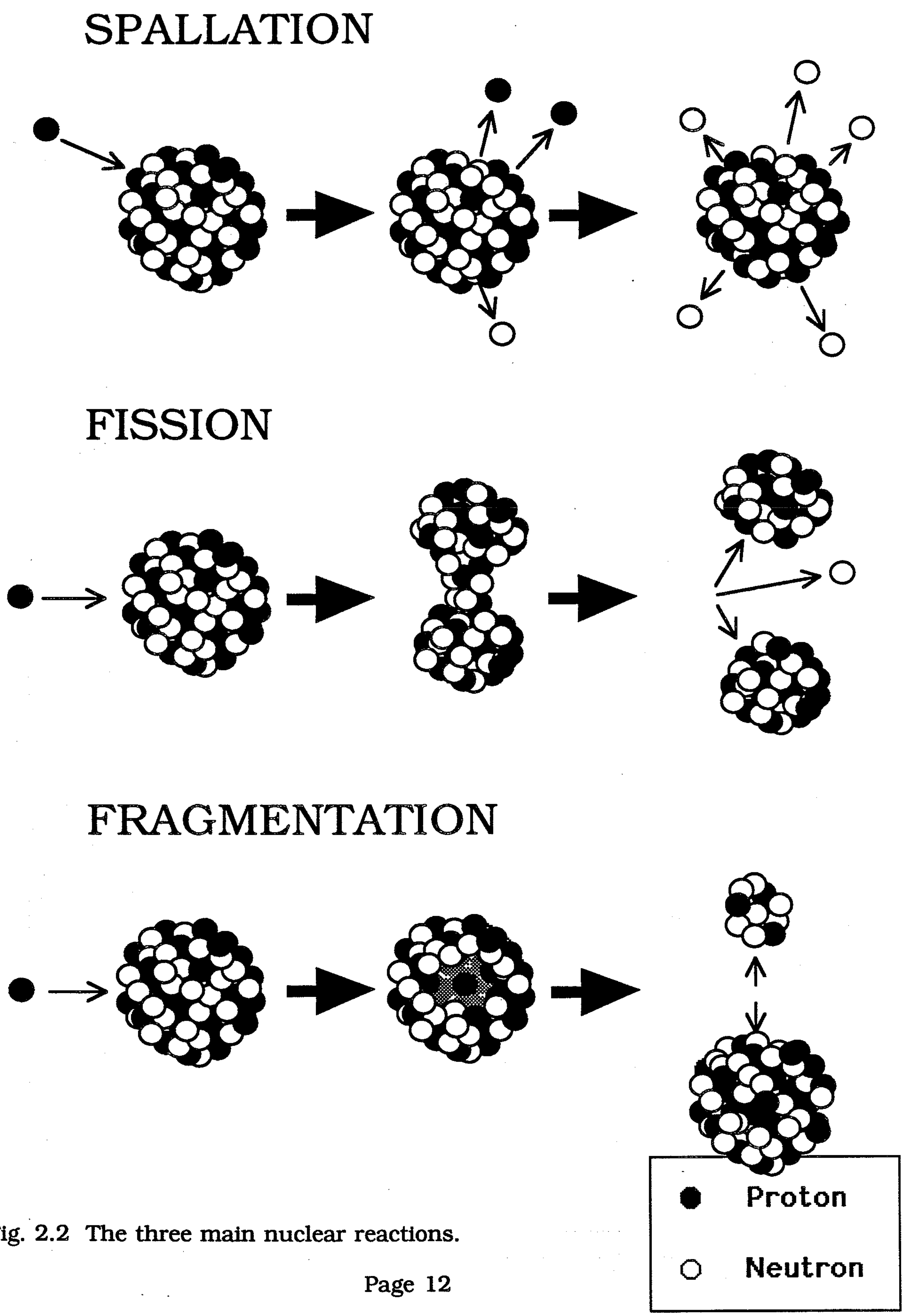


Fig. 2.2 The three main nuclear reactions.

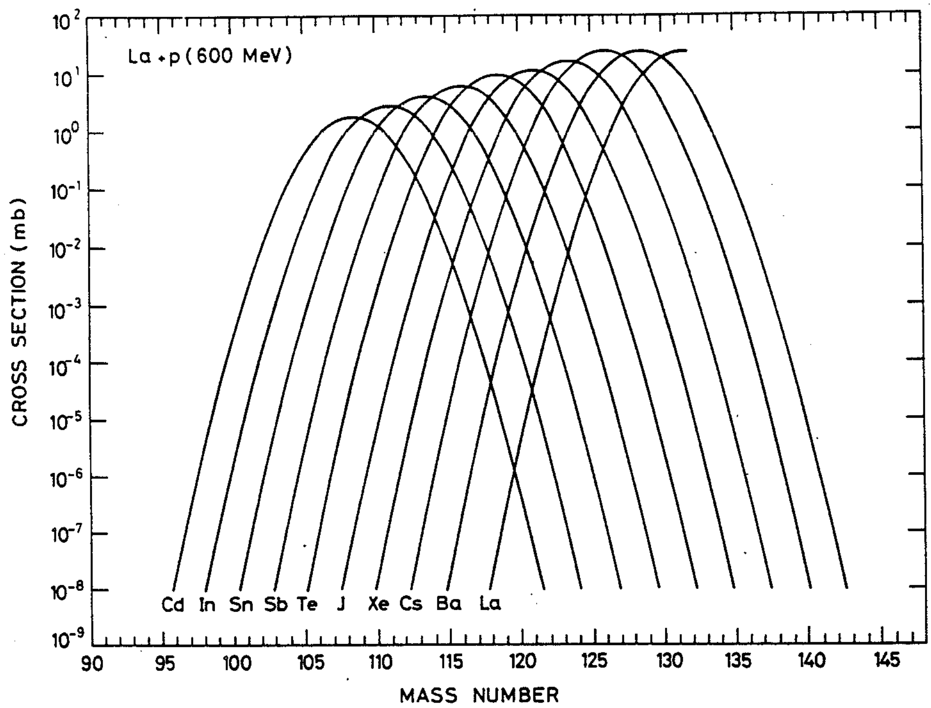


Fig. 2.3 Isotopic yield distribution for the spallation of La by 600 MeV protons, calculated according to the semi-empirical formula of Rudstam

which first joins the target nucleus to create an unstable element, which in turn split into two parts. The use of 600 MeV protons makes the fission symmetric. This is seen from a peak in the medium mass region. Lighter nuclei are therefore unprobable.

The cross-section for producing fission products is about 3 mb close to stability and lower far from it. Neutron rich nuclei are very efficiently produced.

2.3.4 Fragmentation

When the projectile penetrates into the center of the nucleus and deposits its energy, the nucleus behaves much like a drop of water reaching the boiling point; it explodes. This process is called fragmentation.[1] Probable products are one heavier and one lighter nucleus. The fragmentation is in other words responsible for the elements occuring in the $Z < 20$ region. Here the interesting ^{11}Li is formed. This reaction is interesting since it seams to have two neutrons orbiting at great distance from the core, thus forming a neutron halo. ^{11}Li is obviously lying just next to the neutron drip line.

The cross-section for fragmentation close to stability is about 10 mb.

[1] It is also possible to see the fragmentation as the process when a bullit (the proton) with high momentum penetrates through the nucleus, thereby splitting it into parts.

Just like ^3He induced spallation, ^3He fragmentation enhances the production yield.

2.3.5 The Target

There is no universal target or ion source that can produce all elements. Most elements, or groups of elements have to have a separate unit developed. This unit has to meet the requirements of high yield and selectivity. Depending on the half-lives of the nuclides to be produced, the release time must also be short. The sintering process should be slow and the vapour pressure of the target material at least less than 10^{-3} torr. Radiation damage should not affect the target and its container must not corrode.

Figures 2.4-2.7 and 2.9-2.12 all show the configuration of the target and ion source system. The nuclear reaction products are boiled out of target material using a heating current. This way they are made volatile and diffuse out of the target bulk material. Through a transfer line they are then impinging on a suitable ion source were they become charged. Afterwards an extraction electrode kept at a potential of 60 kV accelerates them further towards the mass separator magnet and the rest of ISOLDE. Figure 2.4 illustrates this schematically. Here the yield losses are shown too. They can be estimated by means of off-line tests.[1]

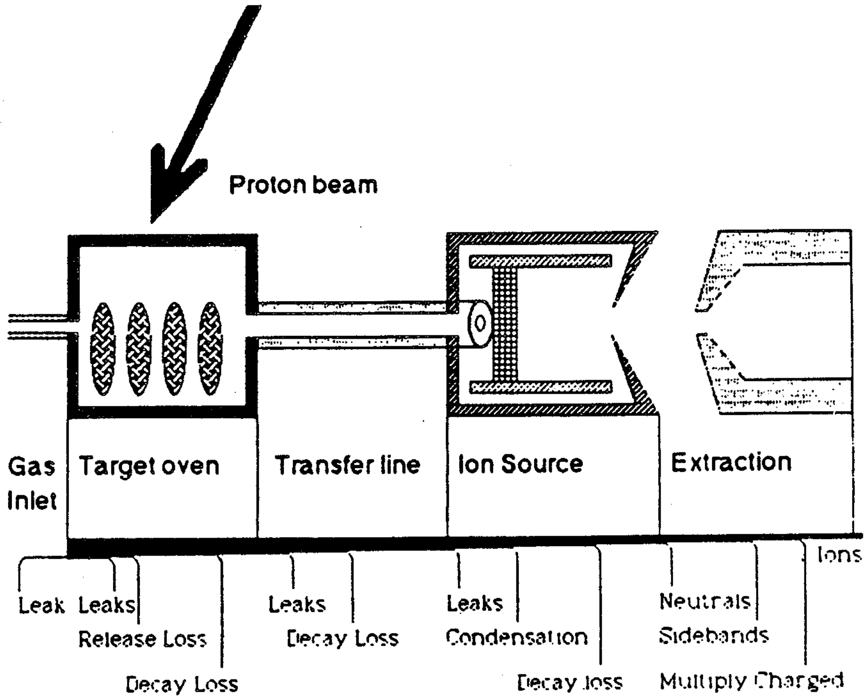


Fig. 2.4 Schematic view of the transfer steps and corresponding losses between target and extraction electrode.

[1]One example of this is my program for 'mass scanning', thoroughly explained in part 2 of this work.

The target container and the transfer line to the ion source is normally made of tantalum. These are then kept in an aluminium cylinder (see e.g. figure 2.6). Due to the high level of radioactivity induced in the target this equipment requires elaborate safety and handeling. Therefore new target units are remotely mounted or dismounted by an industrial robot. A target change takes 24 hours. The target lifetime is between 24 and 300 hours, with an average of 64 hours. The cost is 10 000 Swiss Francs, plus two weaks of man-work in off-line tests.

2.3.6 Off-Line Tests and Target Materials

Before a target assembly is put into use it has been tested off-line to see that if fulfills the demands. Tests like thermochemical stability and release rate of the desired element take place. The release rate is defined as the fraction of the reaction product which has left the target material after a given heating time.

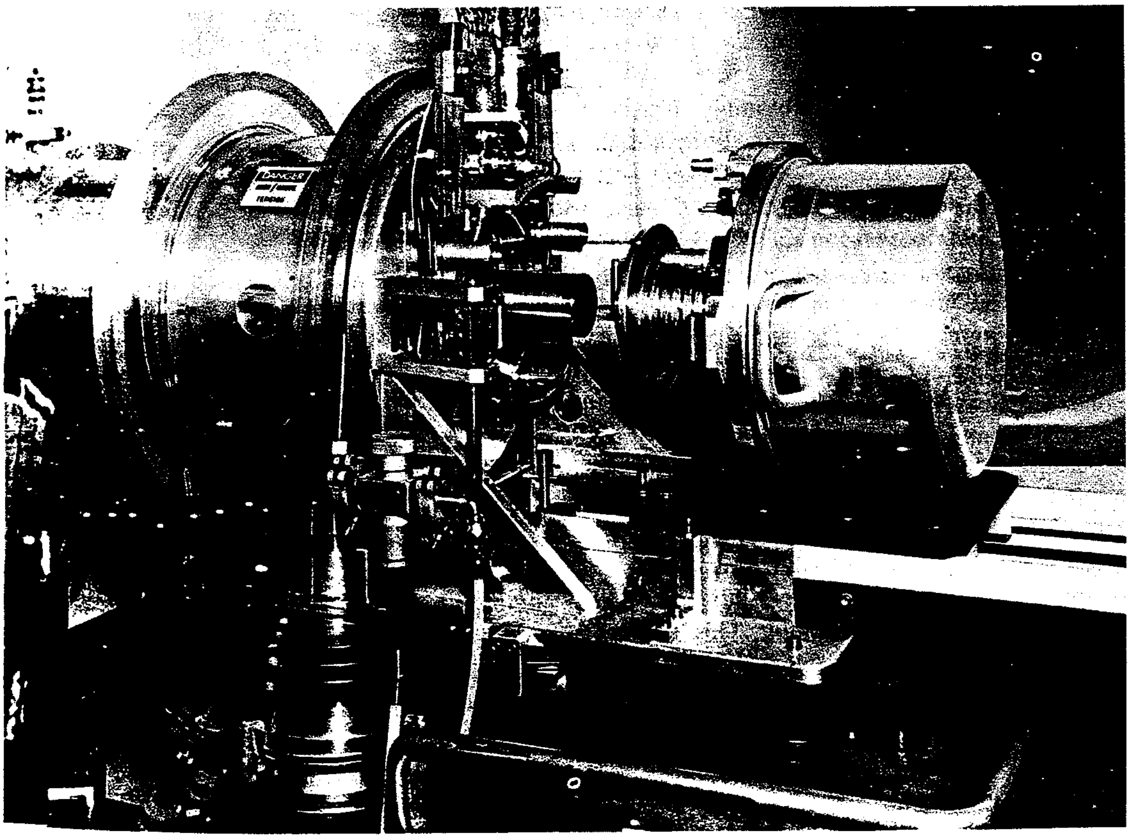


Fig. 2.5 Coupling of a target and ion-source unit to the separator acceleration chamber. The compact assembly consists of a water-cooled base plate (similar to that of a standard vacuum evaporation plant) with feedthroughs, pumping port, and quick connectors. It carries the target and ion source and is enclosed in an aluminium bell jar (for economical reasons a commercial cooking casserole is used). By means of an insulating rail and spindle the unit is moved to the acceleration tank, where the coupling force is taken over by pneumatically operated levers.

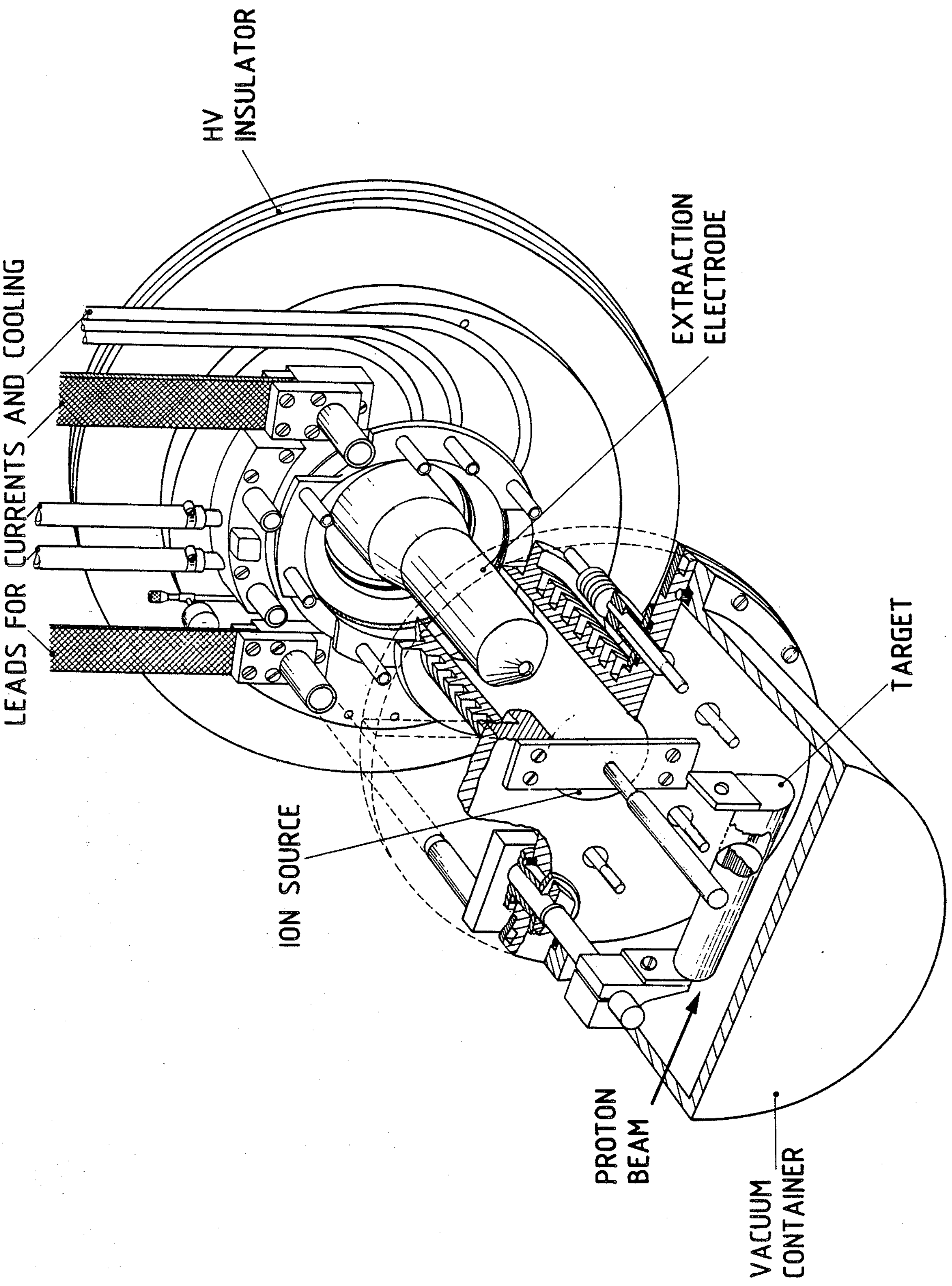


Fig. 2.6 Exploded view of target ion source system

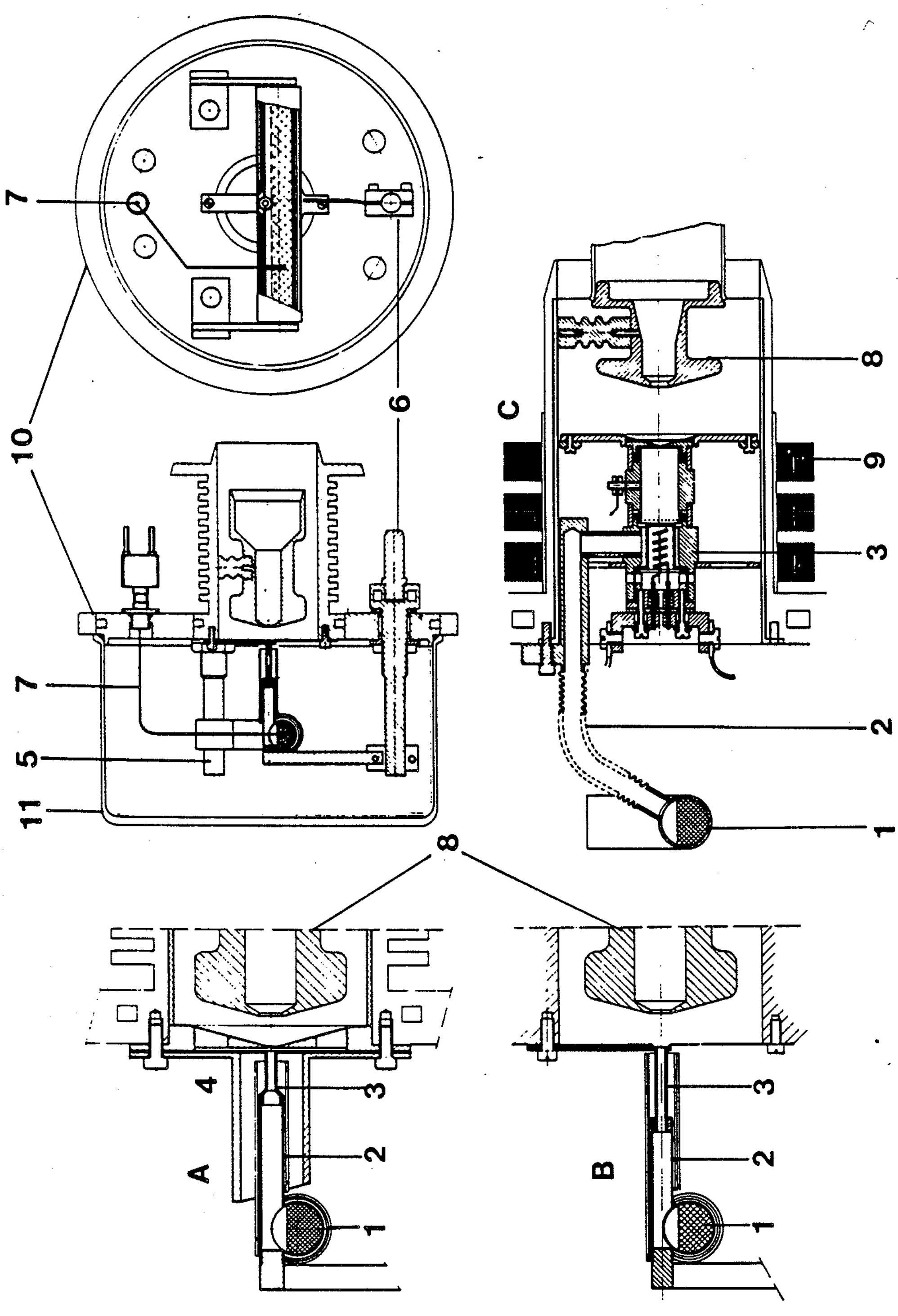


Fig. 2.7 The mechanical construction of the ISOLDE-2 target and ion-source unit. 1) Target container surrounded by heat screens. 2) Transfer line. 3) Ion source. 4) Copper cone. 5) Terminals for target support and heating. 6) Terminal for the line heating. 7) Thermocouple. 8) Extraction electrode. 9) Magnet coil. 10) Base plate with feedthroughs. 11) Vacuum chamber.
Inset A: Low-temperature Ta surface ionizer.
Inset B: High-temperature W surface ionizer.
Inset C: Forced electron-beam-induced arc-discharge source.

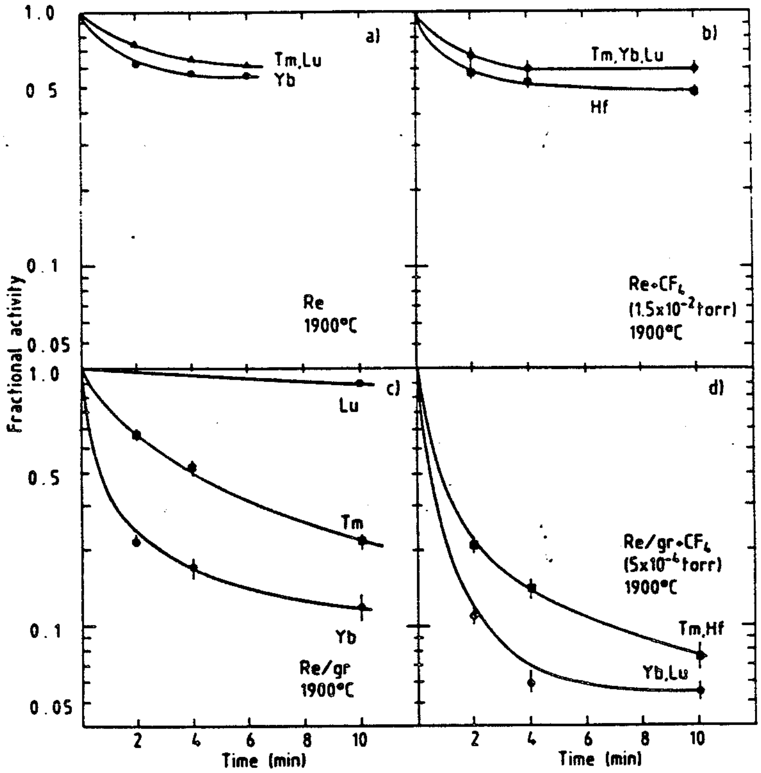


Fig. 2.8 Release of nuclear reaction products from a rhenium powder and from a rhenium/graphite mixture.

Figure 2.8 displays diagrams of fractional activity of Tm, Yb, Lu and Hf from Re targets in different forms and with different additives as a function of heating time. The slopes of these curves can be regarded as the release rates. The targets have first been irradiated with protons to create isotopes of lanthanide elements and Hf. Then the fractional activity is mesured off-line, after the beam has been turned off. Pure Re metal powder at 1900 °C has been used in figure 2.4a. Here the release rate is very slow and Hf is not even observable. If an inlet of reactive CF_4 gas is used, the properties are considerably improved and now Hf also becomes visible (see figure 2.4b). Still the conditions are not good enough, likely due to sintering of the Re powder. This sintering can be inhibited if a mixture of Re powder together with graphite powder is used. Usually the release properties are made even worse through the addition of graphite, due to formation of carbides, but platinum-like metals, like Re, do not react with carbon at high temperatures. Figure 2.4c shows the improvement. In combination with CF_4 gas, as is displayed in figure 2.4d, an optimum is reached and this target is now interesting for on-line applications.

There are two principal release dependent properties to consider: ingrain diffusion and surface desorption. If the former is too slow the release can be made faster by making the surface larger, by using powders, like the Re powder examplified above. The surface desorption is on the other hand improved by an inlet of reactive gas, just like the CF_4 gas, mentioned above.

Metal foils are prefered if the delay of an element is mainly due to slow desorption from the surface and not to slow in-grain diffusion. Ti foil for the production of neutron-deficient Sc isotopes at temperatures of 1600 °C and above, seems to be a promissing target material of this kind. The problem with metal foils is their poor mechanical stability.

Refractory carbides are often used as target materials at ISOLDE. Most of the possible carbides have already been tested, and uranium carbide together with a positive surface ion source seems to be one of the most successful. The release rate of the carbides is improved by the addition of reactive gases.

Even refractory oxides like ZrO_2, HfO_2 and ThO_2 can be used as target materials. The problem with these is their tendency to sinter. This is solved if Ta_2O_5, CaO or MgO are used instead, but they have on the other hand to be shielded from contact with the target container to avoid corrosion.

Already since the start of ISOLDE over 20 years ago, molten metals like Ge, Sn and Pb, have been efficiently used as target materials.

2.3.7 The Ion Source

The purpose of the ion source is to ionize the atoms produced in the target so that they can be accelerated. At ISOLDE there are three kinds of ion sources: positive surface ionizers, negative surface ionizers and plasma-discharge ion-sources. They do also exist in several versions, depending of the chemical properties of the desired elements.

2.3.8 Positive Surface Iionizers

Figure 2.9 shows a positive surface ionizer. It is the type of ion source that is most simple and stable. Many elements can be produced. The atom is required to have a small ionization potential, while the surface work function of the ionizer is choosen to be high. This way single electrons are easily stripped of the atom, and the atom turns into an ion.

The efficency is defined as the ratio between the number of ionized atoms n_+ and the total number of atoms n_{tot} at the surface. It is given by the Saha-Langmuir equation:

$$n_+/n_{tot}= (1 + C\, e^{(W-F)/kT})^{-1}$$

where W is the ionization potential of the atom, F is the surface work function of the ionizer, T the absolute temperature, and C a statistical constant.

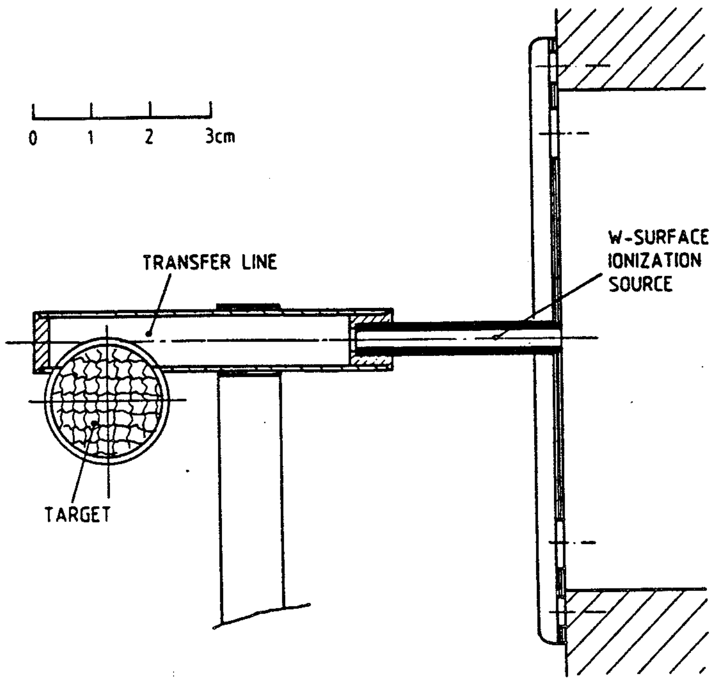


Fig 2.9 Positive surface ionizer.

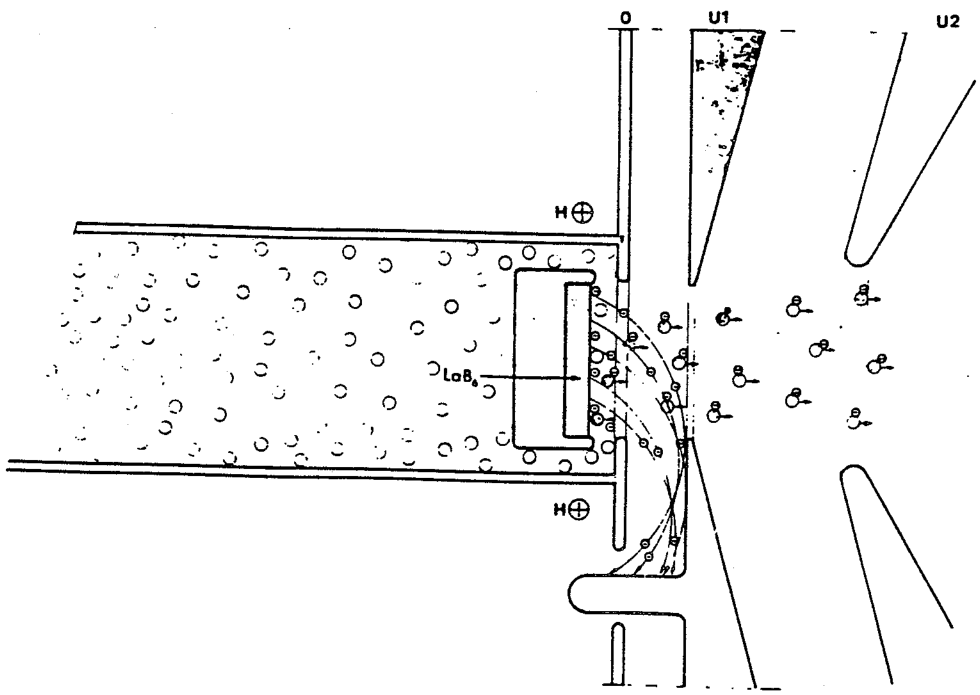


Fig 2.10 Negative surface ionizer.

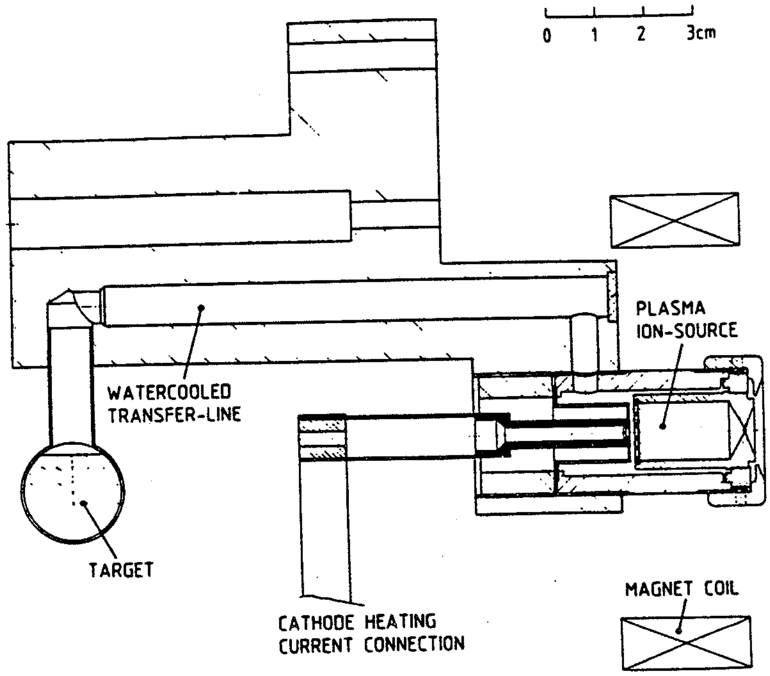


Fig 2.11 Water-cooled plasma-discharge ion source.

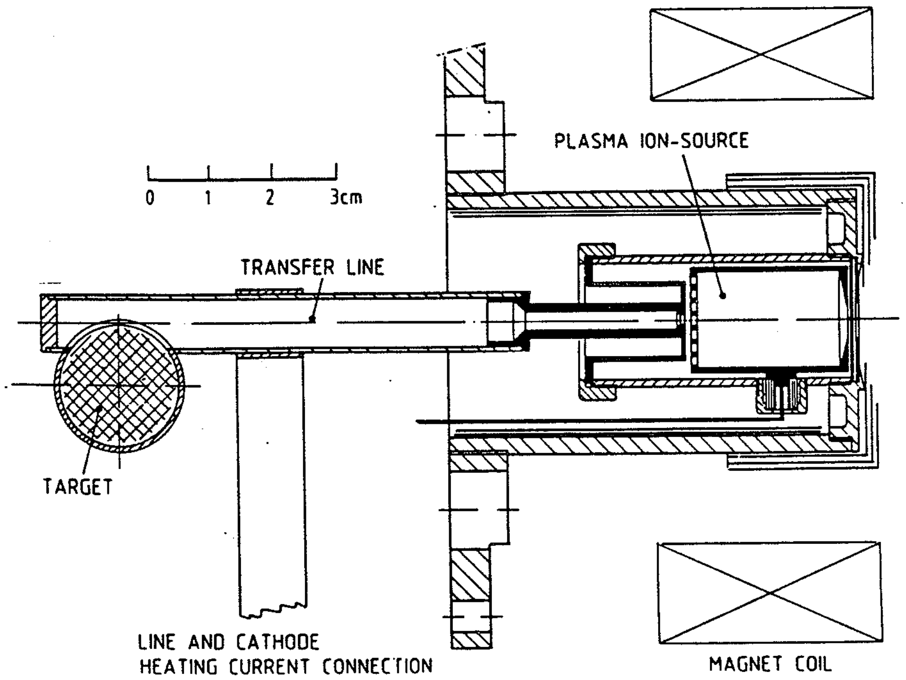


Fig 2.12 High temperature plasma-discharge ion source.

Fr, Cs Rb and K are the most easily ionized elements, having an efficiency close to 100%, and are selectively ionized with a Ta or Re surface ionizer at 1700 ^0C. Group 1 to 3 elements are ionized at 2300 ^0C with a W surface, not selectively but with efficiecies between 5% and 100%.

Molecular ions like LaO^+ and RaF^+ are also produced by positive surface ionizers.

2.3.9 Negative Surface Ionizers

If the atom has high electronegativity, a surface with low work function can transfer its electron to the atom. This is what is done in a negative surface ion source. LaB_6 pellets are useful here as surface. A seperate electrode and a permanent magnet make sure that the excess electrons are taken care of, as shown in figure 2.10. Strong outgasing from targets can lower the efficiency of the ionizer, therefore only targets with little outgasing, like ThO_2 can be used.

2.3.10 Plasma Discharge Ionizers

The majority of elements that cannot emit or absorb electrons at a hot surface have to be ionized by other means. At ISOLDE modified versions of the plasma discharge ionizer FEBIAD (forced electron beam arc discharge) made by Kirchner, are in operation. Here normally the only heating comes from the transfer line between the target and the ion source, and the only way to achieve selectivity of the outcomming elements is to regulate the temperature of this transfer line. If e.g noble gas isotopes are to be produced the transfer line is water-cooled, so that elements which are not gaseous at room temperature will condense before reaching the ion source (see figure 2.11).[1] The efficiencies are only 0.5% for He and 40% for Rn.

Connecting the water-cooling through a small heat-conducting resistor, permits the transfer line to reach temperatures about 200-400 ^0C, and volatile metals like Zn, Cd and Hg can become separated, though not selectively, but with efficiencies between 10% and 60%.

A hot plasma ion source where the whole system has a temperature of 1800-1900 ^0C can take care of the most difficult elements, like the less volatile ones, but it has poor selectivity (see figure 2.12). Reactive gases can be used, and molecular ions can be emitted.

[1]Here there is a separate cathode heating, since there is no heating from the transfer line.

2.4 Separation and Beam Handeling at ISOLDE-2

The mass separator at ISOLDE-2, illustrated in figure 2.13 below, was designed as a low-current Scandinavian electromagnetic isotope separator. The ion beam from the target and ion source is first accelerated by the 60 kV applied to the extraction electrode. This electrode also acts as a divergent lens. It is followed by three other electrodes, forming a convergent lens, directing a parallel ion beam into the focusing-type magnetic analyser. A beam spot with a diameter of 2 mm ±15% over the mass range, is obtained in the focal plane. 5% of the mass range is collected. Thus the separation in A is achieved.

The switchyard, of which a photo is shown in figure 2.15, can direct up to four different beams simultaneously into the beam lines BL-1 to BL-4, using four electrostatic deflectors. BL-1 to BL-4 deliver the beams to experiments in UR-8. BL-2 and BL-3 are also extended up to UR-9 and UR-10, to experiments requiring low background conditions. Those experiments which do not suffer from the background are performed in UR-8.[1]

When BL-2 is deflected up from UR-8, it takes use of a second analysing magnet. Something that further separates and purifies the elements in the beam.

Some important parameters concerning ISOLDE-2 are listed in table 2.3. Figure 2.14 shows some experimental stations at ISOLDE-2.

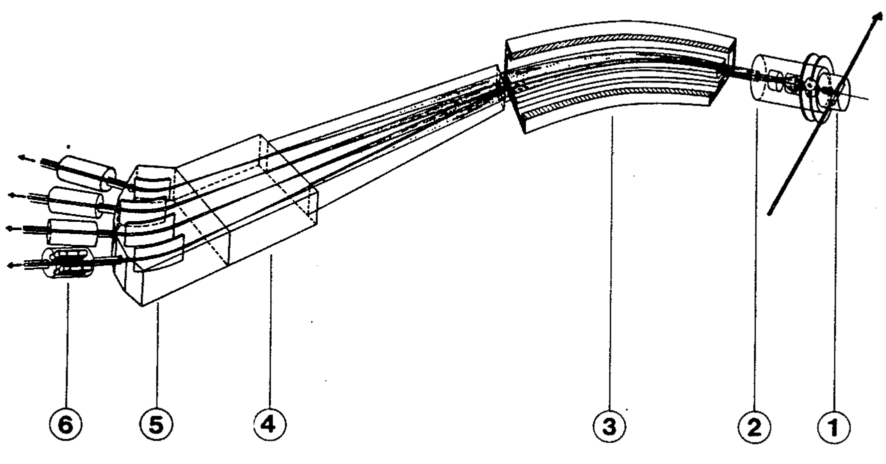


Fig. 2.13 Schematic view of an on-line isotope separator experiment. 1) Target and ion-source unit traversed by the particle beam. 2) Acceleration chamber. 3) Analysing magnet. 4) and 5) Deflector chamber with electrostatic switchyard. 6) External beam line.

[1]All along BL-1 to BL-4 there are electrostatic elements. To adjust these I have written a special computer program, which is fully explained in part 2 of this work.

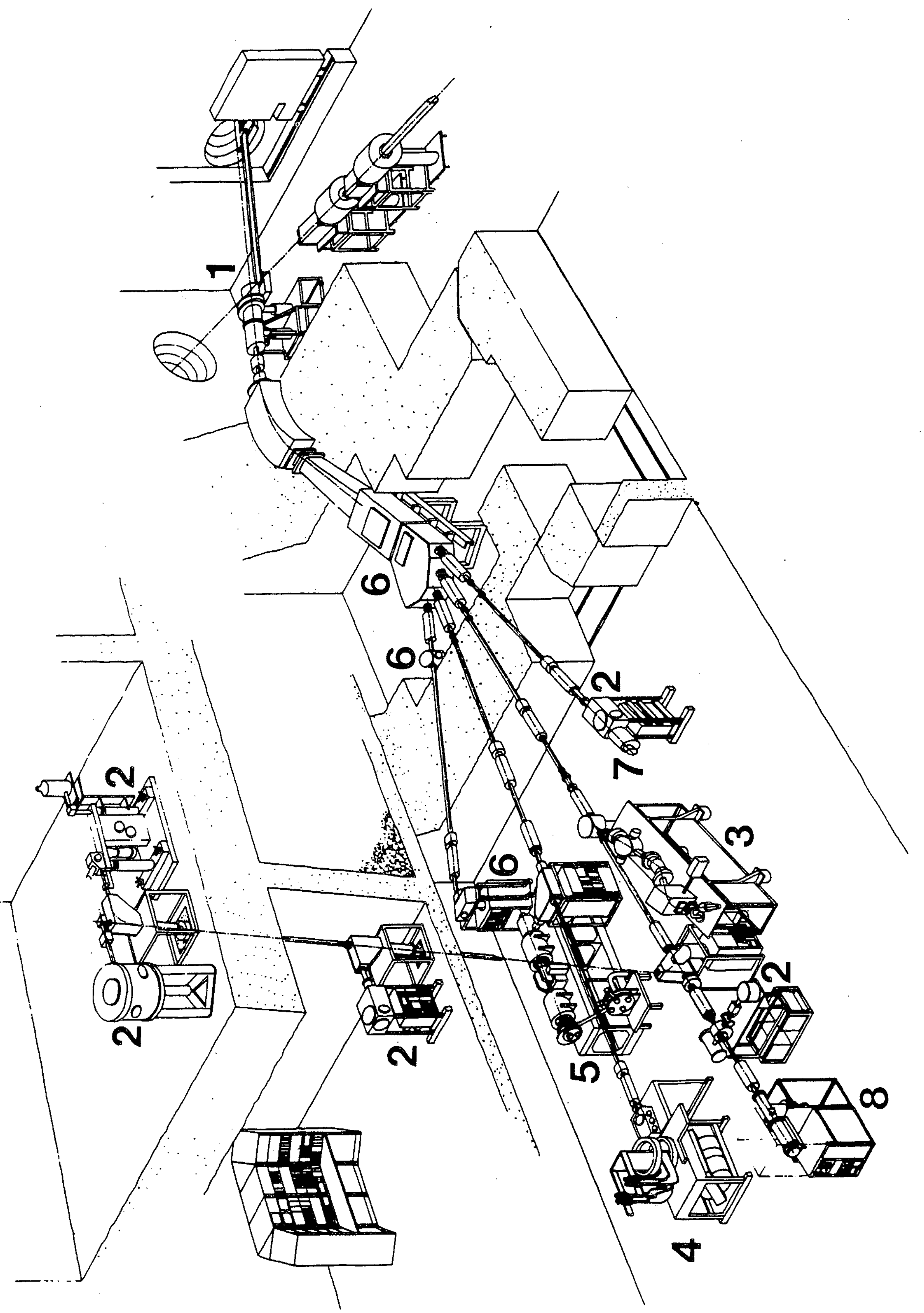


Fig. 2.14 Perspective view of the experiments at ISOLDE-2. 1) Target and ion-source development. 2) Nuclear spectroscopy (α, β, γ). 3) High-resolution mass spectrometer. 4) Optical pumping and laser spectroscopy. 5) Atomic-beam magnetic resonance. 6) Collection of radioactive sources for off-line work (hyperfine interactions in solids, determination of shifts in X-ray energies, targets for nuclear-reaction studies). 7) Beta-delayed particles. 8) Range measurements of ions in gases.

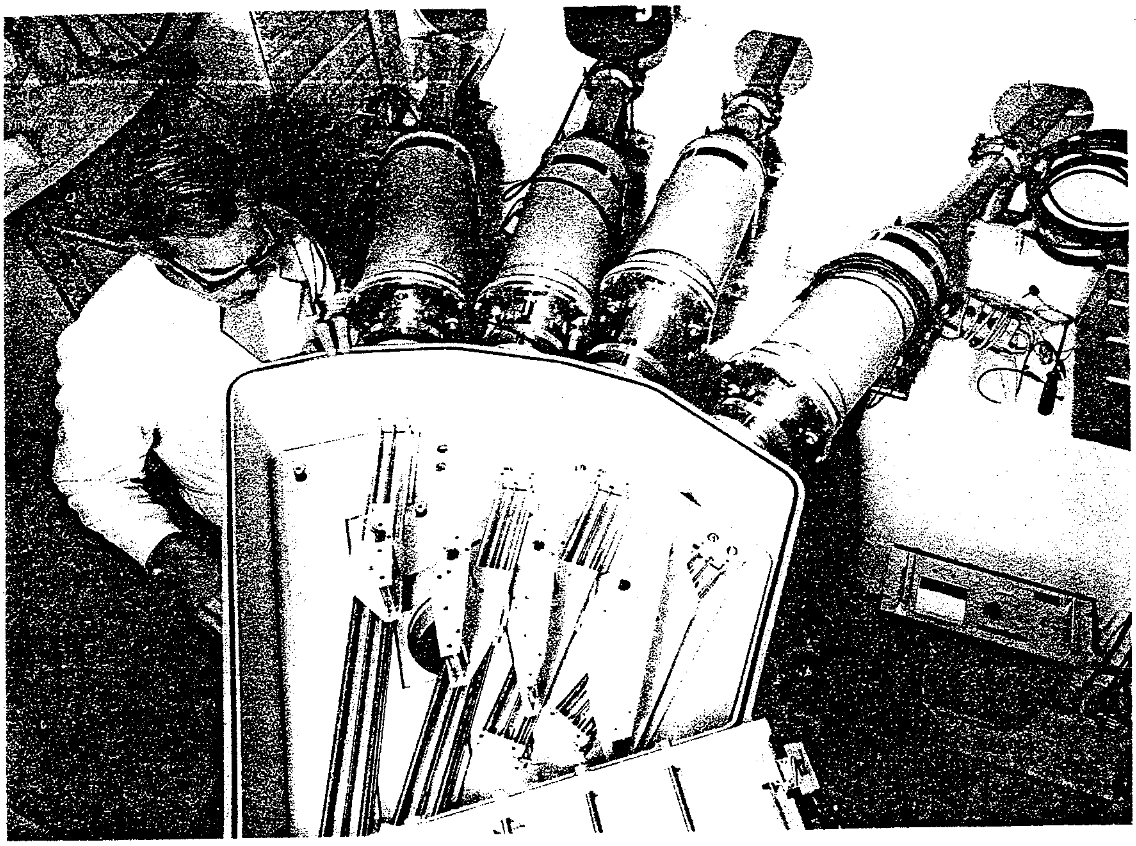


Fig. 2.15 The arrangement of the four electrostatic deflector plates of the switchyard, viewed from above and in the direction of the external beam lines.

2.5 Separation and Beam Handling at ISOLDE-3

The ISOLDE-3 target and ion source is located in the SC vault. After the extraction electrode the beam is transported 10 meters through the SC shielding wall (see figure 2.16). When the beam enters the Proton Hall it has been separated in two magnetic analysers, both bending the beam to the left. It has also passed two focal points, accomplished by several electrostatic quadropoles (see figure 2.17).

Switchyards deliver the beam to different experiments in the Proton Hall. There are two nuclear spectroscopy areas, one collinear laser area, and one solid-state physics area.[1]

Table 2.3 shows some parameters for ISOLDE-3. Here are also ISOLDE-2 parameters listed, so a comparison is possible. The higher beam intensity; 3 mA at ISOLDE-3 is notable.

[1]At one of the experimental stations in the Proton Hall, a step motor is used. The regulation system I made for this motor, is described in part 3.

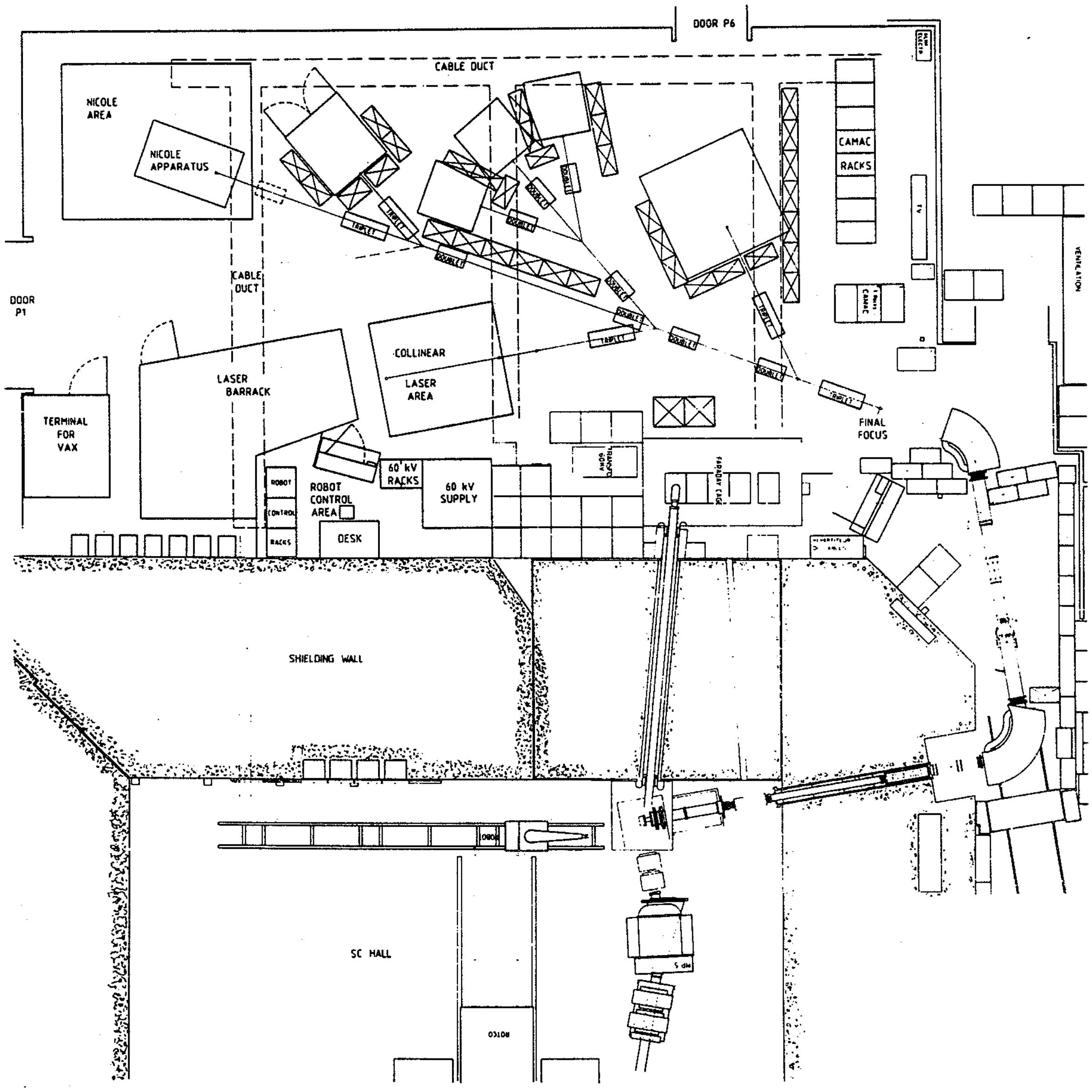


Fig. 2.16 Layout of the ISOLDE-3 mass separator and of the experimental area in the Proton Hall

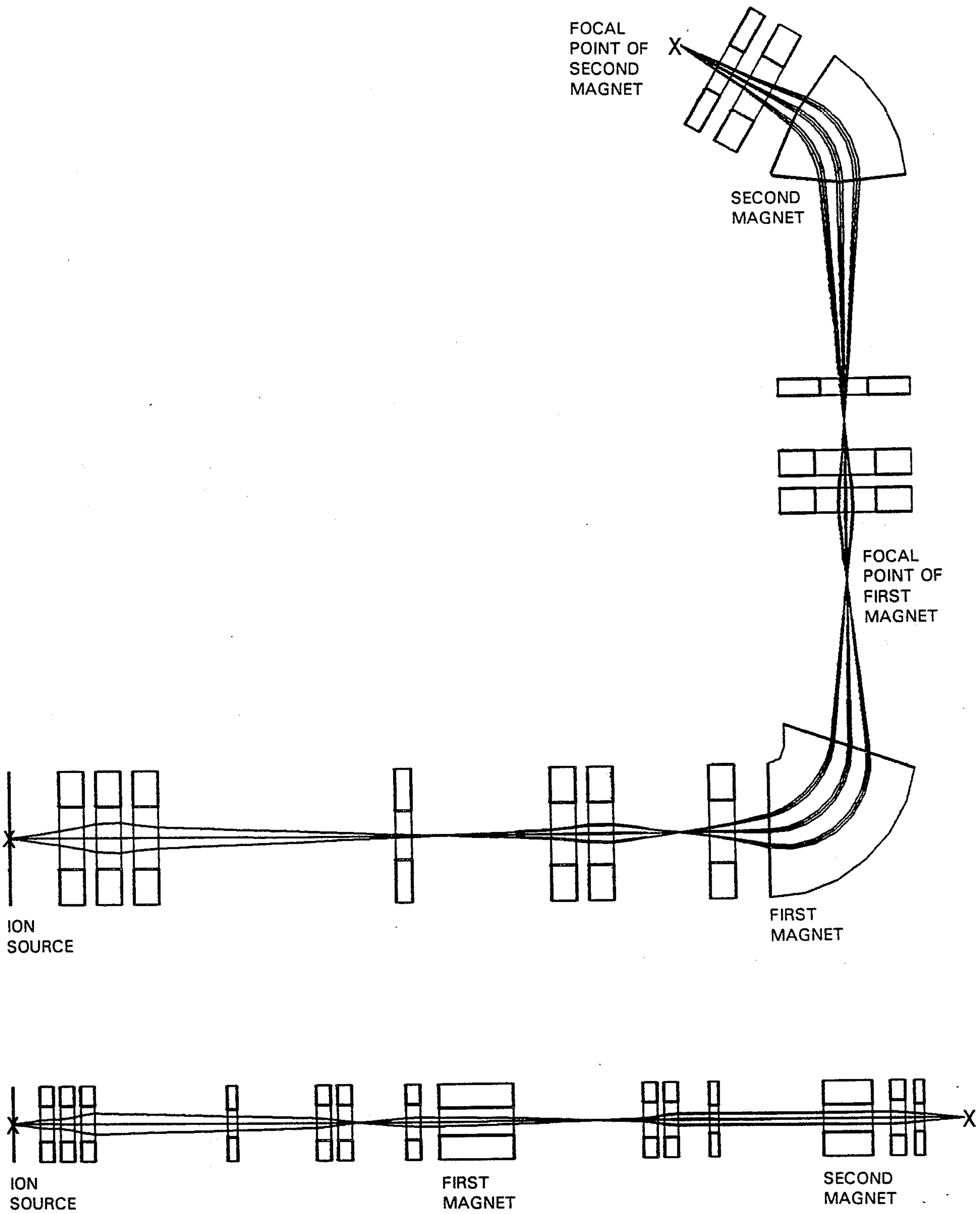


Fig. 2.17 Beam optics of the ISOLDE-3 mass separator using a point-like ion source.
Top: Horizontal plane of the separator.
Bottom: Vertical plane.

Table 2.3

IS-2 and IS-3 parameters (status at November 1985)

	IS-2	IS-3
Acceleration voltage (kV)	60	60
HT stability (V)	± 1	± 1
Maximum stable ion current (mA)	0.5	3
Source emittance (π mm mrad)	0.04–0.13	25
Mass dispersion (mm)	15 at mass 100	12 at mass 100
Resolving power	500–1500	5000–30000
Enhancement factor	10^4–10^6	10^6–10^8
600 MeV proton beam (μA)	2.6	4
910 MeV ^{3}He beam (μA)	1	5
Vacuum system	Turbomolecular pumps	Turbomolecular pumps
Analysing magnets number	1	2
Analysing magnets angle (deg.)	55	90, 60
Analysing magnets radius (m)	1	1, 1
Beam size in focal plane (mm)	3	< 4
Mass range in focal plane (%)	± 15	± 12
Beam transport	Electrostatic	Electrostatic
Control system	Hard-wired	Computer
Area for experiments (m^2)	295	200[a)]
Lift capacity (kg)	750[b)]	Overhead crane
Number of elements available	66	> 66

a) In the Proton Hall (total area in Proton Hall is 330 m^2).
b) For heavier equipment, special arrangements are possible.

3 PHYSICS AT ISOLDE

3.1 Beam Research

The number of elements that have been directly produced at ISOLDE are 66 (see figure 3.1). Table 3.1 summarizes the characteristics of targets and ion sources used for production of light elements. It is estimated that there are about 2000 undiscovered unstable nuclei with half-lives longer than 1 ms. The detection of any of these is one of the reasons for having a continuous development program on targets and ion sources.

At present the most important parts of the work are to improve ion sources and to invent new methods for element selectivity.

Collecting yields, and plot them as in figure 3.2 is routine.

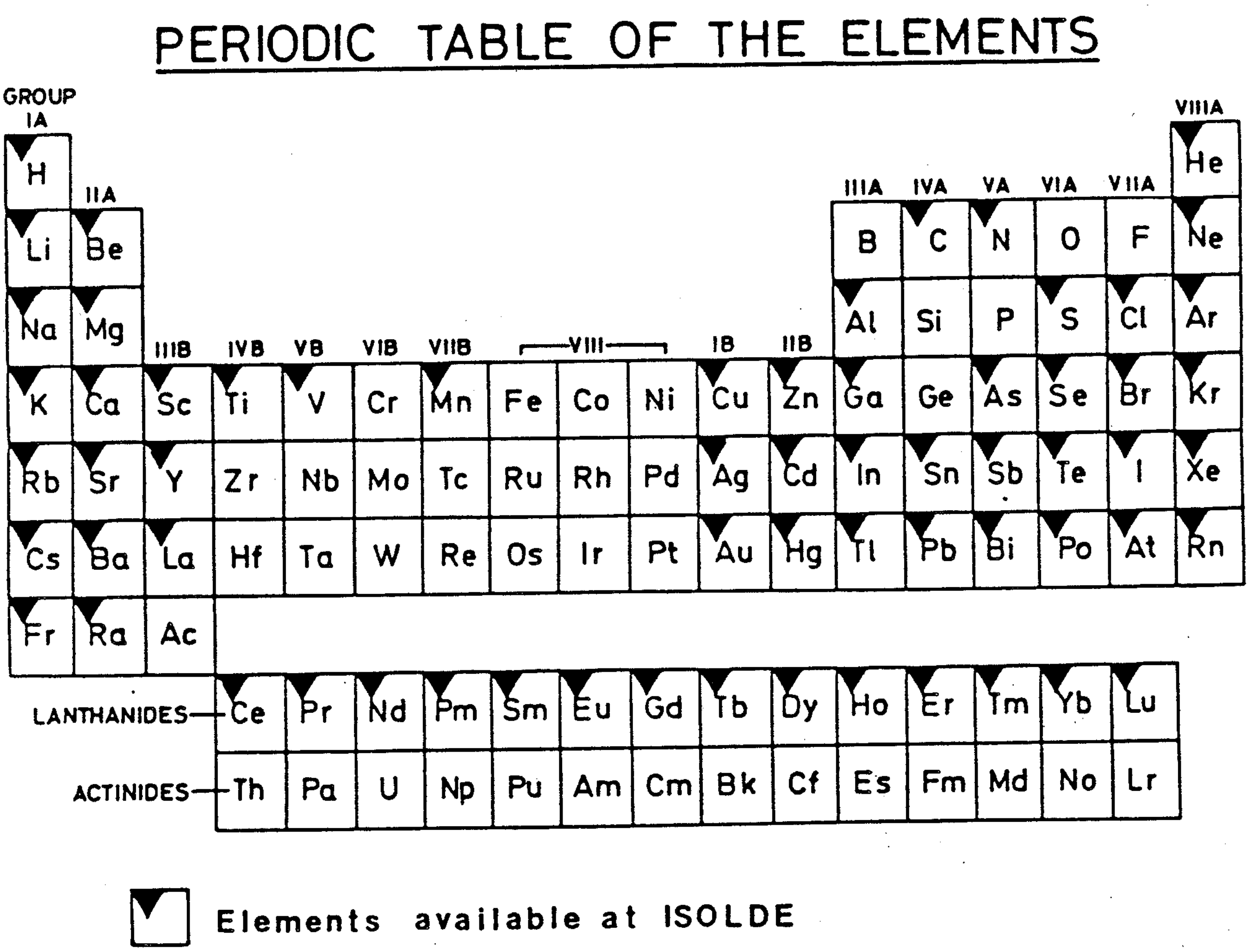


Fig. 3.1

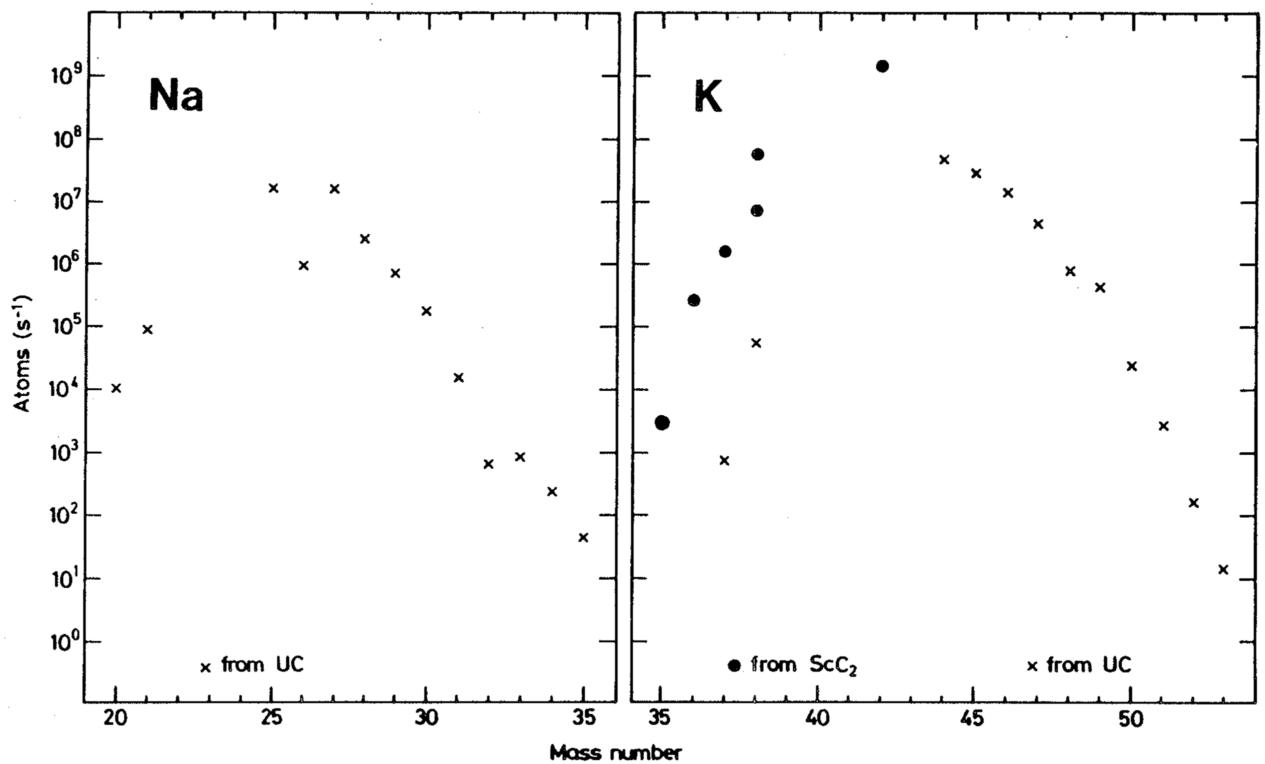


Fig. 3.2 Observed yields of sodium and potassium isotopes, corresponding to a $1\,\mu A$ 600 MeV proton beam impinging on a $14\,g/cm^2$ uranium–carbide target and a $3.2\,g/cm^2$ scandium carbide target. The yields below ^{27}Na and ^{50}K are measured by means of a $4\,\pi sr$ beta counter, those above are estimated from measurements of beta-delayed neutrons and normalized to the beta yields of ^{29}Na and ^{50}K, respectively.

Table 3.1

Status of target materials and suggestions for further developments

Element	Target material n-deficient	Target material n-rich	Evaporated species	Ionization	Status and remarks
Helium	UC ThC Ta	$Be^{18}O$	He	Plasma discharge ionization of He	Operational at ISOLDE but could be improved if an ion source with higher efficiency can be found.
Lithium		UC ThC Ta	Li	High-temperature surface ionization of Li^+	Operational at ISOLDE with element selectivity and high efficiency.
Beryllium	C B_4C TiC VC	UC Ta ZrC Hf_5Si_3 $Ir+CCl_4$	Be, BeO $BeCl_2$	High-temperature plasma discharge ionization of Be^+ or BeO^+	Release studies are positive. A beam of low yield was made at ISOLDE. Chemical evaporation with CCl_4 or CF_4 seems to be a possibility.
Boron	$C+CF_4$ $C+CCl_4$	$Ir+CF_4$ $Ir+CCl_4$ ThO_2	BF_3 BCl_3 B_2O_3	Plasma discharge BO^-	B seems to be a very difficult element to bring on line. The listed targets are only suggestions. The BO^- beam is normally emitted by the LaB_6 cathodes.
Carbon	BeO B_2O_3 MgO	CaO Ta_2O_5	CO CO CO	Plasma discharge ionization of preferably CO^+	A number of oxides release CO, and a beam exists at ISOLDE but only with a low yield presumably because of low ionization efficiency.
Nitrogen	BeO B_2O_3 MgO	CaO Ta_2O_5	N and N_2 N and N_2 N and N_2	Plasma discharge ionization of N^+ and $N_2{}^+$. A CN^- as observed from surface ionization sources may be of interest.	At ISOLDE, several oxide targets have given beams of N^+ and $N_2{}^+$. The yields are low and an ion source with higher efficiency should be found.
Oxygen	AlC_2	UC ThC TaC	CO CO CO	Plasma discharge ionization of C^+ or CO^+	The release of CO from these target materials has only been observed off line.
Fluorine	AlC_2	Ta+Ba Re+Ba UC+Ba	AlF_3, BaF_2 BaF_2 BaF_2	Plasma discharge ionization of F^+ or surface ionization of BaF^+	Fluorine has not been observed on line. It is a suggestion to add Ba vapour in order to form the often observed ion BaF^-.
Neon	MgO Al AlC_2	UC ThC Ta	Ne Ne Ne	Plasma discharge ionization of Ne^+	Operational but could be improved if an ion source with higher efficiency can be found.
Sodium	Al AlC Hf_5Si_3 LaSi	UC ThC Ta	Na Na Na Na	Surface ionization of Na^+	Operational with element selectivity and high yield.
Magnesium	Al AlC_2 Hf_5Si_2 LaSi	UC ThC Ta	Mg Mg Mg Mg	Plasma discharge ionization of Mg^+	Mg has only been observed from ScC_2 and Ta targets under non-optimized conditions. The presence of Zr vapour may help to keep the Mg in the volatile elemental form.

Table 3.1 (cont.)

Element	Target material n-deficient	Target material n-rich	Evaporated species	Ionization	Status and remarks
Aluminium	$Si+CF_4$ Hf_5Si_3	UC ThC $Ta+CF_4$ $Ir+CF_4$	AlF_3, Al Al AlF_3 AlF_3	Plasma discharge ionization of Al^+	Al has been seen from UC and Ta targets combined with a surface ionization source. Owing to its high ionization potential, a plasma source should be used and tested for its Al yield. Chemical evaporation is an attractive possibility since AlF_3 has a low boiling point.
Silicon	$CeS+CF_4$ $CaCl_2$ CeS	$Ir+CF_4$	SiF_4 $SiCl_4$ SiS	Plasma discharge ionization	Si is a difficult element to bring on line. The targets mentioned are only suggestions to be tested. Since TaS is strongly bound, a system made entirely of graphite may be needed.
Phosphorus	$CaCl_2$		PCl_3	Plasma discharge ionization	No target material has been found that releases P. This is mainly due to lack of a suitable tracer nuclide.
Sulfur	VC Hf_5Si_3	Hf_5Si_3 $Ir+O_2$	S, SiS SO_2	Plasma discharge or surface ionization	The only target material which has been found to release S is VC. Since S was not observed on line, it is suspected that it was lost as TaS. In that case an all-graphite system should be tested.
Chlorine	V ScC_2	UC ThC ThO_2 TaC Ta	Cl Cl	Plasma discharge ionization	The best target for Cl production seems to be Ta foils. The beam was made by means of surface ionization. A plasma source may give higher yields.
Argon	CaO	UC ThC Ta	Ar Ar Ar	Plasma discharge ionization of Ar^+	The best target found is CaO. Better yields may be obtained if a more efficient source can be found.
Potassium	ScC_2 ScLa Ti V	UC ThC Ta	K K K K	Surface ionization	Operational with high efficiency and element selectivity. Since ScC_2 tends to decompose, Ti and V targets should be tested.
Calcium	LaSc Ti V	UC ThC $Ta+Zr$ $Ir+CH_3Br$	Ca Ca Ca $CaBr_2$	Plasma discharge ionization	Element only observed in surface ionization. For efficiency increase a plasma source should be tested. Reduction with Zr or chemical evaporation with Br are interesting possibilities.

3.2 The Experimental Program

The experiments at ISOLDE aim to explore the properties of atomic nuclei in the regions far from stability. There are experiments going on in atomic, solid-state, and surface physics, elementary particle physics, and in medical science. Over 200 researchers from about 50 laboratories are involved.

Here follows a list of some of the ISOLDE experiments that have been done or are in progress:

* Ground-state spins, moments, and isotope shifts.

* Atomic and nuclear effects in the energies of K X-rays.

* Nuclear masses and decay energies.

* Beta-delayed particle emission; nuclear structure and statistical decay properties.

* Exotic decays.

* Light elements, isospin symmetries.

* Ground-state alpha widths.

* Detailed nuclear level schemes; beta-strength functions.

* Determination of the rest mass of the electron-neutrino.

* Preparation of radioactive targets for accelerators and intense neutron beams.

* Solid-state physics by perturbed angular correlations, Mössbauer spectroscopy, channeling, and diffusion studies.

* Surface physics.

* Atomic physics of unstable elements.

* Intense positron sources.

* Nuclear physics data for astrophysical applications.

Part 2

THE SYSTEM FOR CONTROL AND TEST

1 INTRODUCTION

1.1 The Beam Adjustment System

The four beam lines of ISOLDE-2 leading to different experimental stations are shown in figures 1.1-1.3. The different beam line elements, such as deflectors, lenses and correction elements, are adjusted by applying voltages from their respective high tension power supplies. Since 1980 these are the LeCroy HV4032A, 32 photomultiplier high voltage system crates, whose tension values may either become selected locally by buttons or remotely by a computer. In 1984 Ove C. Jonsson constructed a system to take advantage of this remote setting possibility by using a CAMAC crate, an IEEE bus and a PET computer.

The wish to make easy adjustments of the high tension values by using a computer mouse, to move a cursor over the computer screen and "click" on a special bar for every special beam line element, resulted in my new version of the beam adjustment system using the Olivetti PC.

The hardware is illustrated in figure 1.6.[1] The beam adjustment system consists of the following parts:

*An Olivetti Personal Computer M380/C equiped with a GPIB Elite card for IEEE bus communication, a hard disk , and a Logitech Mouse (section 2.1).

*The IEEE bus (section 2.2).

*A CAMAC crate (section 2.3).

*A LeCroy 8901 CAMAC to IEEE interface (section 2.4).

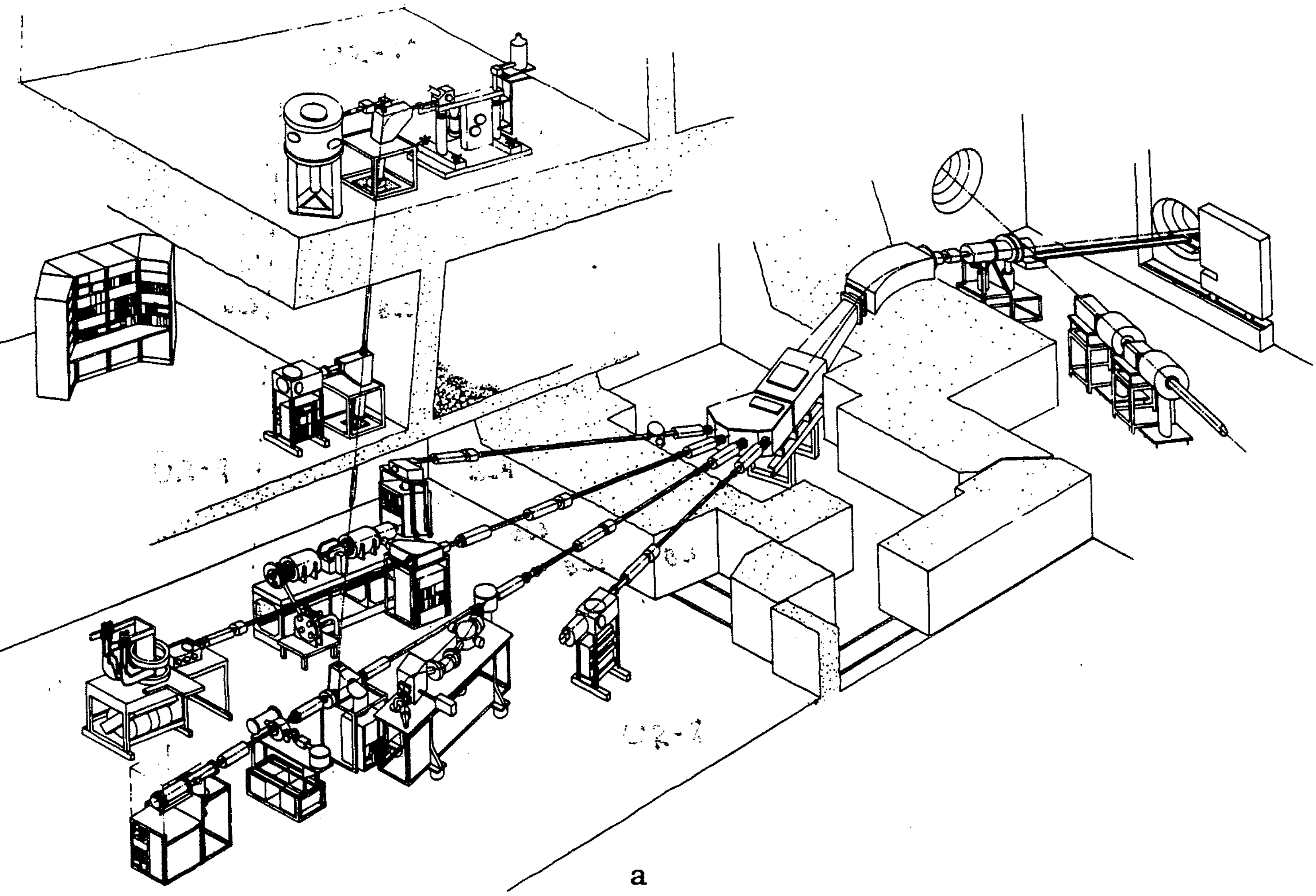
*A LeCroy 2132 HV4032 to CAMAC interface (section 2.6).

*Five (or six) LeCroy HV 4032A, 32 photomultiplier high voltage system crates (section 2.7).

*The different ISOLDE-2 beam line elements whose settings are affected (section 2.8).

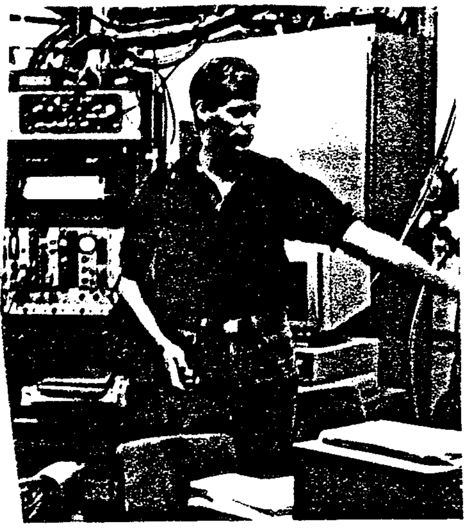
---

[1]To give a convenient survey of of the whole ISOLDE-2 application that I have constructed, the mass spectrometer system (next section) is also included in figure 1.6.



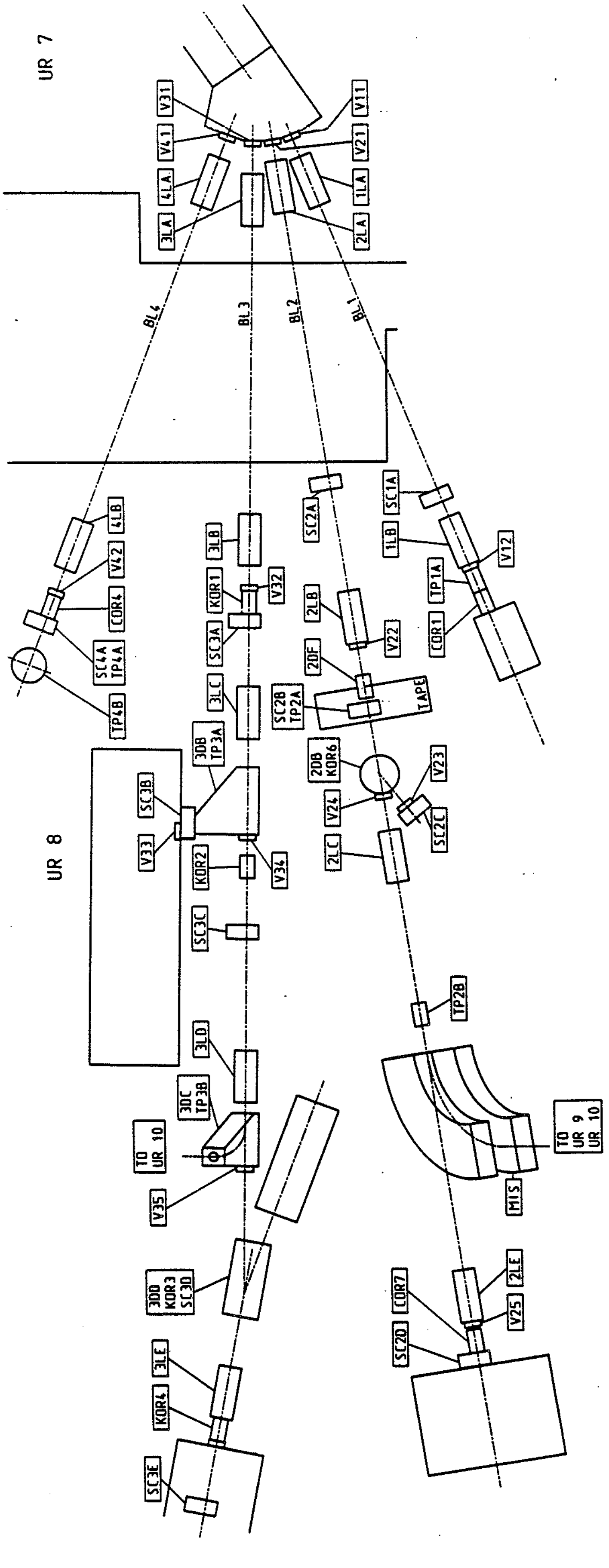
a

b



c

Fig. 1.1 a) The four ISOLDE-2 beam lines. b) and c) Myself in UR-9 where the equipment that I used is situated.



BL	beam line
V	valve
L	electrostatic lens
KOR and COR	electrostatic x-y deflector for beam correction
D	electrostatic beam deflector
SC	beam scanner and Faraday cup for beam diagnostic
TP	turbomolecular pump
MIS	magnetic beam deflector in beam line 2
TAPE	tape transport system for yield measurements

Fig. 1.2 Layout of experimental area (UR8) at ISOLDE-2

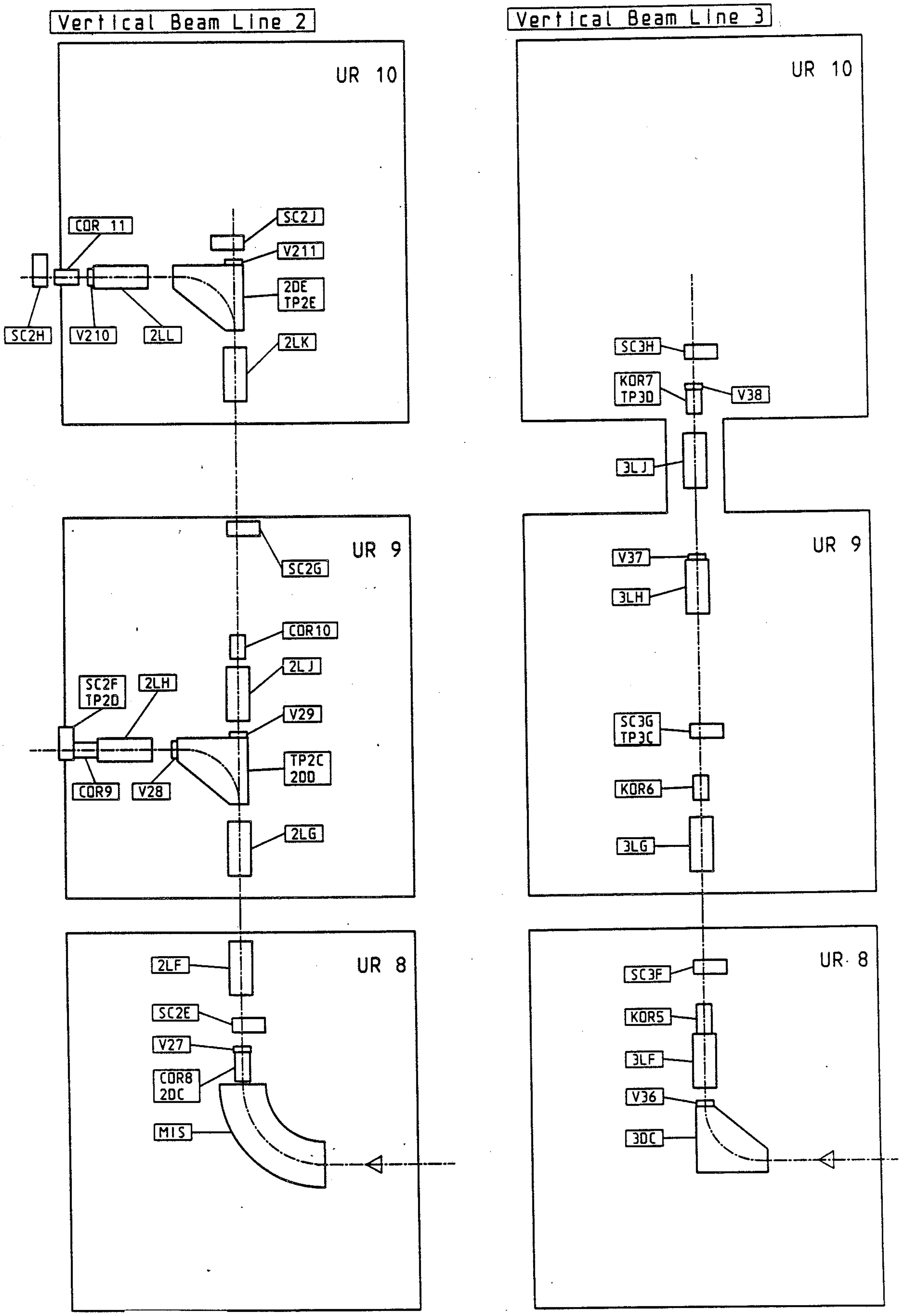


Fig. 1.3 Beam lines 2 and 3 going from UR8 to UR9 and UR10 at ISOLDE-2

The operation of the hardware units is explained in the sections mentioned within brackets. The beam adjustment system software follows in chapter 3.

1.2 Mass Spectrometry

To analyse which isotopes are present in a beam, ISOLDE-2 has an oscillating needle scanner that measures the current across the beam after the separator magnet (figure 1.4). The current is then displayed on an oscilloscope screen (figure 1.5). What is shown is a mass spectrum, that may be used for mass calibration. Alternatively it shows which masses are in the beam, if the calibration is already made.

The disadvantages with this "mass spectrometer" are that it only registrates the peaks in the neighbourhood of the selected mass and that the spectrum disappears when the beam is turned off, i.e. one would like to have some sort of documentation of the mass spectrum.

The solution to this problem is to use a computer that can both stear the analysing magnet and handle measured data. The choice is again the Olivetti PC with accessories (section 2.1).

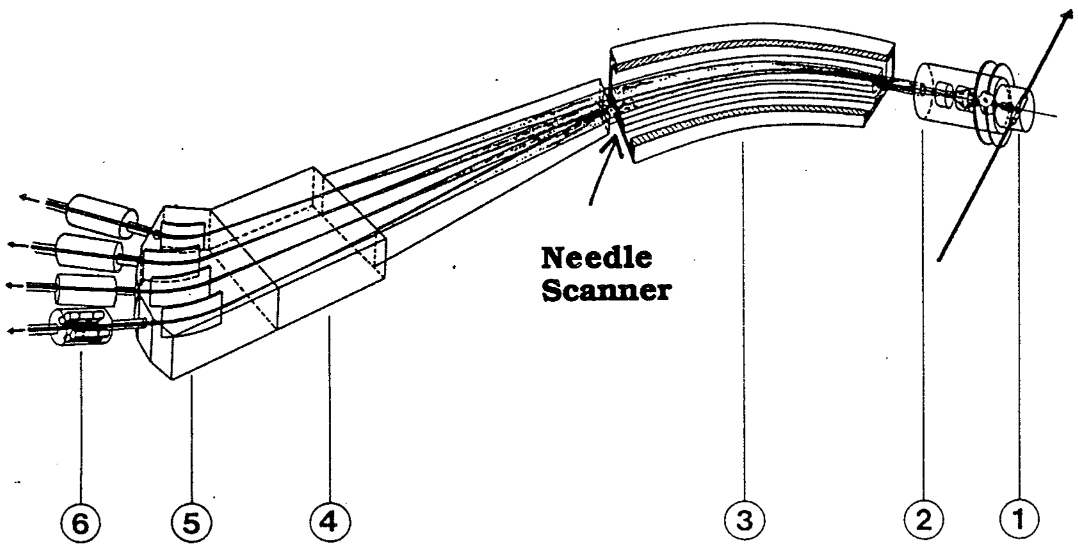


Fig. 1.4 Schematic view of an on-line isotope separator experiment. 1) Target and ion-source unit traversed by the particle beam. 2) Acceleration chamber. 3) Analysing magnet. 4) and 5) Deflector chamber with electrostatic switchyard. 6) External beam line.

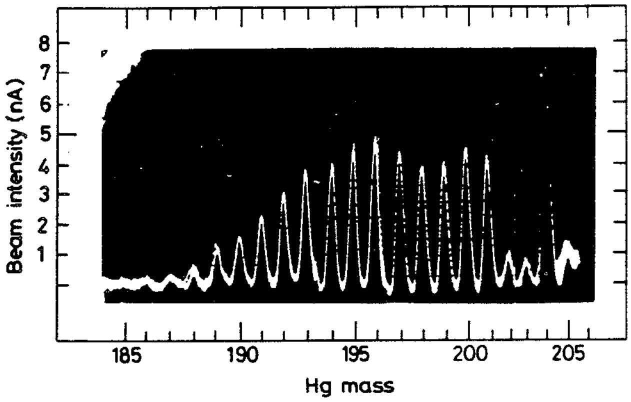


Fig. 1.5 Mass spectrum of the most intense radioactive Hg beams and ^{204}Pb as measured with an oscillating needle scanner.

The complete system is shown in figure 1.6. The instrumentation is partly shared with the beam adjustment system. This is the case with the PC, the IEEE bus and the CAMAC crate with 8901 interface (sections 2.1-2.4). Exclusively belonging to the mass spectrometer system are:

*A Nuclear Enterprises Multidac Type 9091 Digital/Analogue Converter (section 2.5).

*The analogue electronics controlling the analysing magnet (section 2.9).

*A Keithley 485 Electrometer that reads out the beam current, using a 4853 IEEE interface card for remote control (see section 2.10).

The software for the mass spectrometer is described in chapter 3. Results from measurements are devoted to chapter 4.

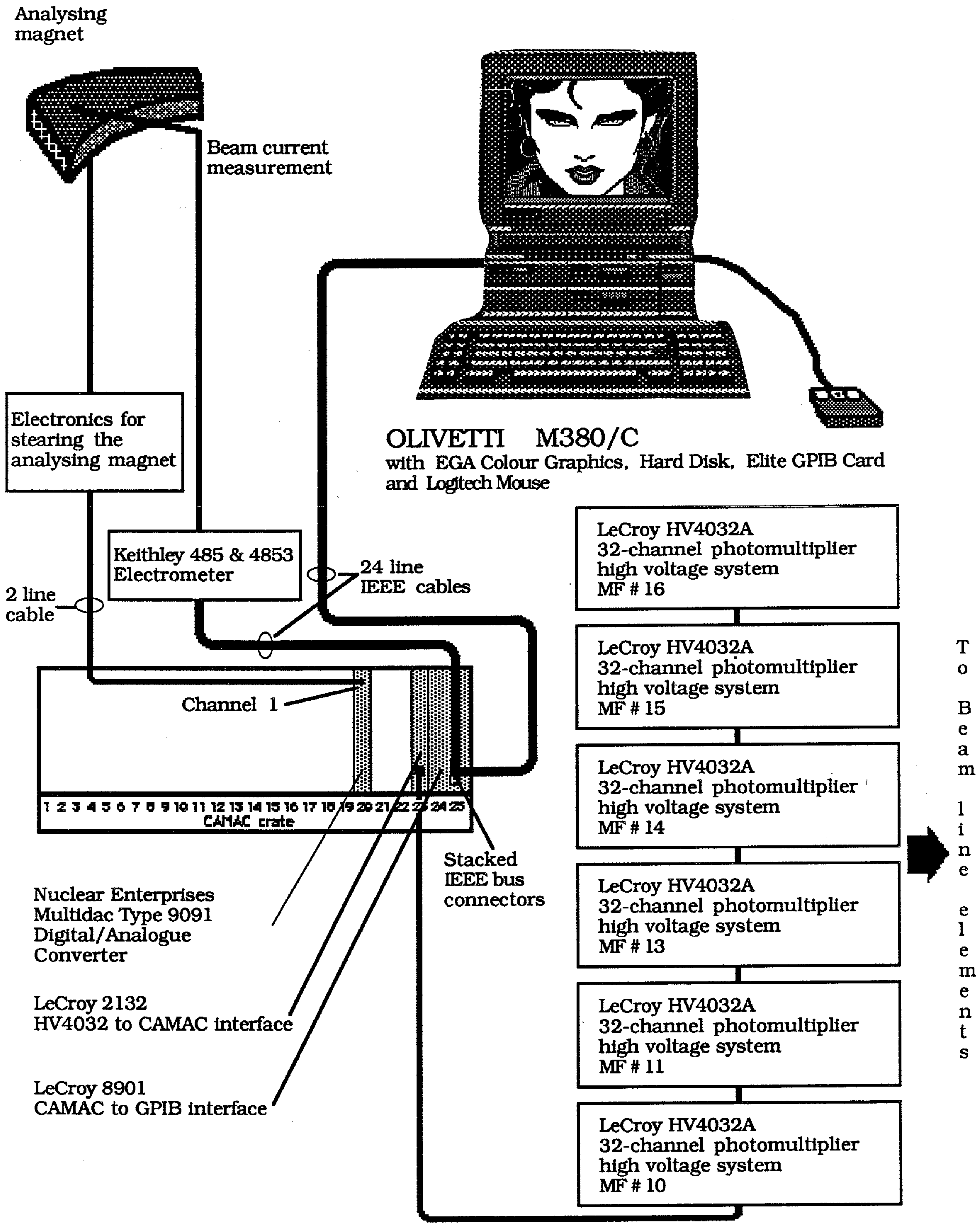


Fig. 1.6 The control and test system. It consists of the beam adjustment system and the mass spectrometer.

2 THE HARDWARE

2.1 The Olivetti Computer

The computer used in the system is an Olivetti Personal Computer M 380/C. It uses a colour screen for EGA graphics and is equiped with a hard disk with several Mbytes of memory, which makes it comfortable to save computer codes. There are also two disk drives; one for 3 1/2 inches diskettes and one for 5 1/4 inches diskettes. To handle a great deal of the computer interactions more easily, a Logitech mouse with three buttons is also connected to the Olivetti PC.

The Olivetti is not built with an interface for IEEE bus communication, but there is a special plug-in card available made by Brain Boxes Elite that serves this purpose. This card is used in our system and has address 02E1 Hex.

You are refered to the Olivetti manuals in the litterature list for further technical details on the Olivetti computer.

2.2 The IEEE Bus

2.2.1 Introduction

The IEEE bus is an instrumentation data bus standardized by the Institute of Electronic and Electrical Engineers in 1975. In 1978 it was revised to become the ANSI/IEEE-488-1978 standard. The bus is also referred to as the General Purpose Inteface Bus or GPIB.

A "bus" may be defined as a bunch of conducting lines that communicate information, such as addresses, data and commands, and the hardware and software for sending, receiving and controlling this information.

2.2.2 The Cables and Connectors

The "conducting lines" for the IEEE bus is mainly one or a series of 24-line cables that connect one or several instruments and a controlling device (e.g. an Olivetti PC) together as shown in figure 2.1.

The connectors, pictured in figure 2.2-2.3, are easily stacked over each other as the zoomed-in picture in figure 2.1 points out. You just have to plug a connector over the instrument output ditto and tighten the screws by hand or with a screwdriver. Then since an IEEE connector has two connecting sides; one male and one female, you can plug in and screw tight another one and so forth. Figure 2.4 and 2.5 display two possible configurations: linear and star configuration systems.

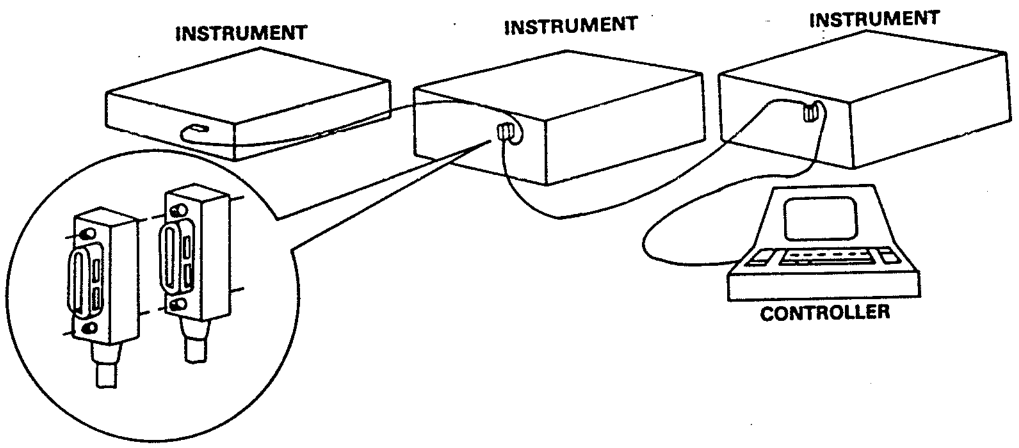


Fig. 2.1 IEEE-488 Connections

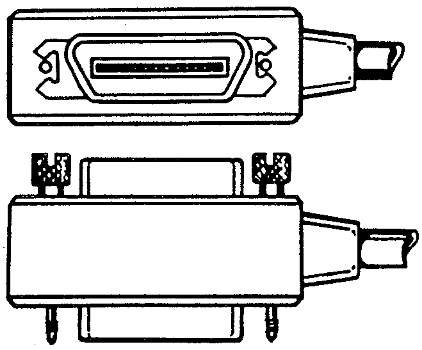


Fig. 2.2 IEEE-488 Connector

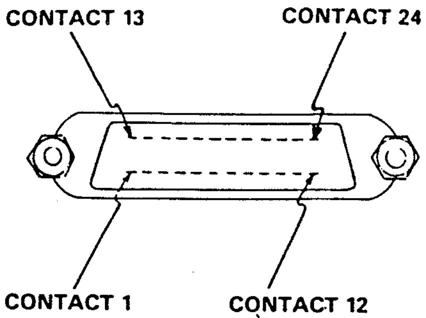


Fig. 2.3 Contact Assignments

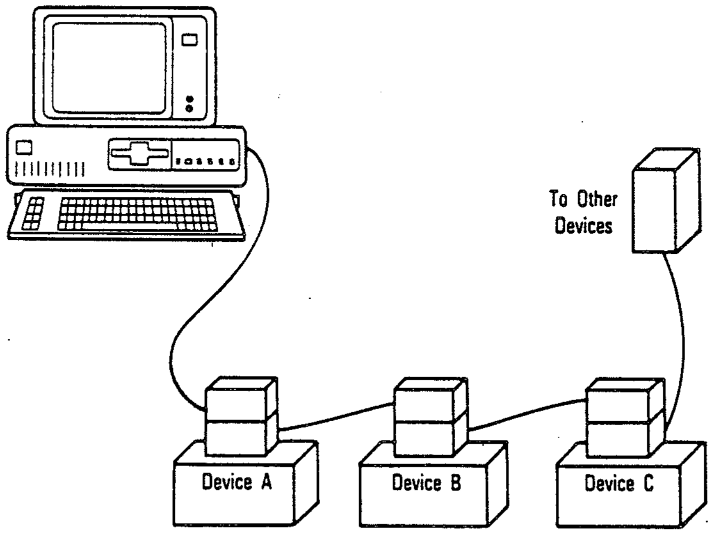


Fig. 2.4 Linear configuration IEEE system.

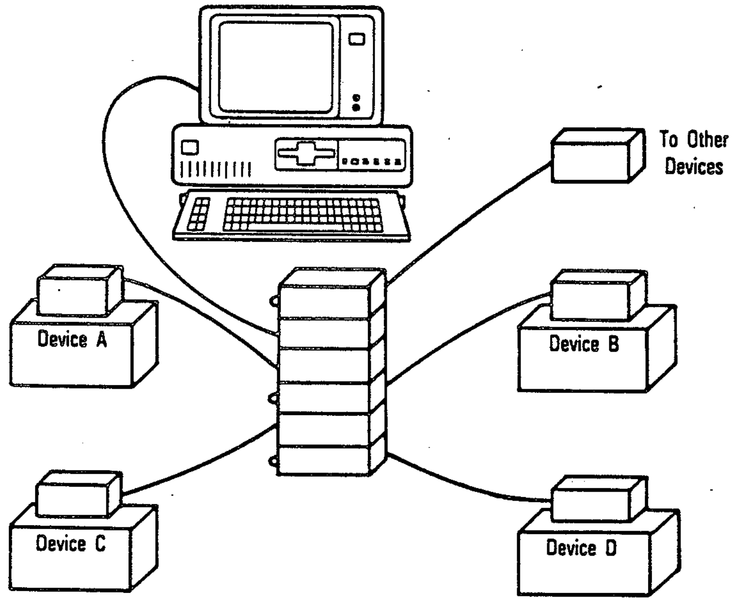


Fig. 2.5 Star configuration IEEE system.

There are some restrictions on the cables:

* A maximum of 15 devices, including the controller, on the bus.

* The maximum length of any single span between two devices is 4 meters.

* The maximum average length is 2 meters per device.

* The maximum total length for all interconnected instruments is 20 meters.

* The maximum voltage between IEEE common and ground must not exceed 30V or damage to the instruments may occur.

The bus lines are assigned to different pins in the connector as figure 2.3 tells. Table 2.1 lists what type of signal the different bus lines carry. There are a total of 24 signal lines used to implement the bus. Sixteen are signal lines, one is the cable shield, and six are twisted pair common for six of the signal lines.

Table 2.1

IEEE Contact Designations

Contact Number	IEEE-488 Designation	Type
1	DIO1	Data
2	DIO2	Data
3	DIO3	Data
4	DIO4	Data
5	EOI (24)*	Management
6	DAV	Handshake
7	NRFD	Handshake
8	NDAC	Handshake
9	IFC	Management
10	SRQ	Management
11	ATN	Management
12	SHIELD	Ground
13	DIO5	Data
14	DIO6	Data
15	DIO7	Data
16	DIO8	Data
17	REN (24)*	Management
18	Gnd, (6)*	Ground
19	Gnd, (7)*	Ground
20	Gnd, (8)*	Ground
21	Gnd, (9)*	Ground
22	Gnd, (10)*	Ground
23	Gnd, (11)*	Ground
24	Gnd, LOGIC	Ground

*Numbers in parentheses refer to signal ground return of reference contact number. EOI and REN signal lines return on contact 24.

2.2.3 The IEEE Signal Lines

The purpose of the IEEE bus was to make a parallel data transfer medium that optimized the number of lines needed. This is achieved by using 16 signal lines devided into the following three categories:

* 5 interface management lines: ATN, IFC, REN, EOI, SRQ

* 3 handshake lines: DAV, NRFD, NDAC

* 8 data input/output lines: DIO1-DIO8[1]

The lines use low (0.4V or lower) for true and high (2.4V or higher) for false, i.e. negative logic.[2]

The signal lines together with an example of a bus system is shown in figure 2.6.

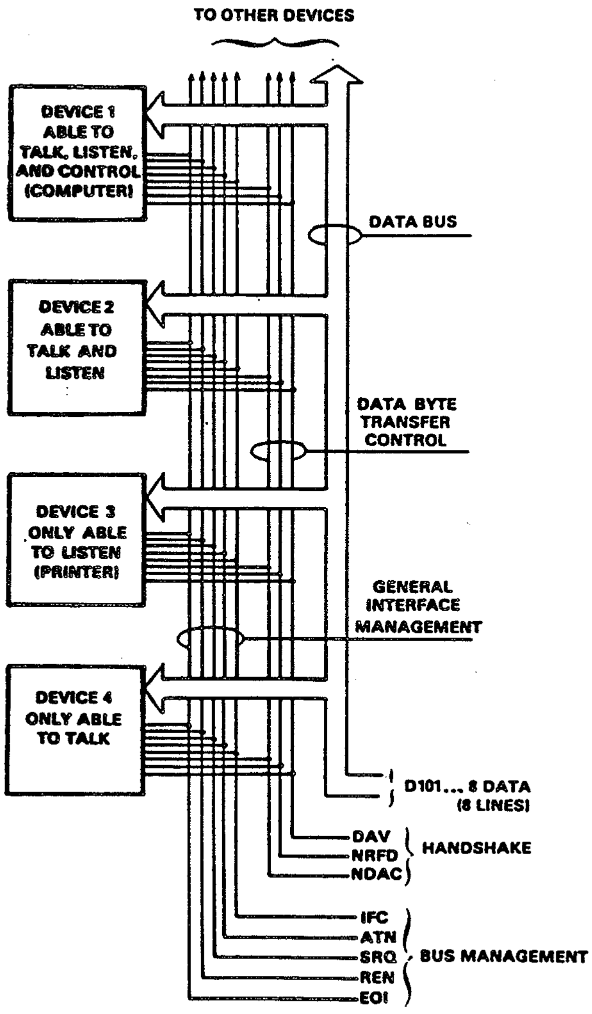


Fig. 2.6 IEEE Bus Configuration

[1] Se also table 2.1.
[2] The numbers in brackets are voltage levels for Transistor-Transistor Logic or abbreviated TTL.

Notice that all devices have the same information available on the bus simultaneously. The devices can operate in three modes: as controller; talker; listener.

The controller, also called system controller, controller-in-charge and adapter is only one at a time on the bus and it usually is the computer. This is at least the case in my system. The controller uses the interface management lines to execute commands by simply setting the corresponding line low.

The talker sends data on the data lines and the listener receives data. Several devices may be commanded to listen at once, but there can only be one talker at any given time. Otherwise the communication would be like picking up a single voice from a large crowd.

Before a device can talk or listen it must be appropriately addressed. How this is done is illustrated in section 2.2.6.

To make reliable transfer of the information on the 8 data lines a handshake procedure is taking place using the three handshake lines. This procedure is described in section 2.2.5.

2.2.4 The Bus Management Lines

2.2.4.1 Introduction - There are 5 bus management lines that ensure an orderly transfer of data. The commands that are sent are single-line commands or with another term; uniline commands. The following is a brief description of these commands.

2.2.4.2 ATN (Attention) - The attention line is one of the more important management lines. The state of ATN determines whether the data bus is to be considered as sending commands or data. If the controller sets ATN low the data bus transmits multiline commands. If on the other hand the controller sets ATN high the data bus contains device-dependent commands or data strings.

For an explanation of what "multiline" and "device dependent" commands mean, see section 2.2.6 about the data lines. Table 2.2 lists the various types of IEEE commands.

2.2.4.3 IFC (Interface Clear) - Setting the IFC line low makes the bus go to a known inactive state. This can only be done by the controller. IFC can be sent at any time.

2.2.4.4 REN (Remote Enable) - Setting the REN line low sends the REN command. This sets all instruments that use it in remote mode instead of local. They have to be in remote mode before it is possible to program them over the bus.

2.2.4.5 EOI (End Or Identify) - The EOI is set low by the talker simultaneously with the last byte of a multibyte transfer on the data lines to indicate to the listener that it should stop reading.
The controller may also set both ATN and EOI low to make a parallel poll on the instruments, i.e. making them send their status on the data lines.

2.2.4.6 SRQ (Service ReQuest) - The SRQ line is set low by an instrument when it requires service from the controller.

2.2.5 The Handshake Lines

2.2.5.1 Introduction - There are 3 handshake lines working in a sequence to ensure reliable data transfer regardless of the transfer rate. It is actually the slowest device on the bus that determines the transfer rate.

In the following let "source" mean the controller if ATN is low and the talker if ATN is high. Similarly let "acceptor" mean all the instruments on the bus if ATN is low and only the listener(s) if ATN is high. Here is a description of the handshake lines:

2.2.5.2 DAV (Data Valid) - The state of the DAV line is controlled by the source. When DAV is low it indicates that the signals on the data lines are really the wanted byte.

2.2.5.3 NRFD (Not Ready For Data) - The acceptor controls the state of the NRFD line. It is high (i.e. false) when the acceptor is ready to receive data and conversely low (i.e. true) when it is not.

2.2.5.4 NDAC (Not Data ACcepted) - The acceptor also controls the state of the NDAC line. When the acceptor has received the data byte properly it is set high (i.e. false).

2.2.5.5 The Handshake Sequence. - Figure 2.7 illustrates the hanshake sequence when one byte of data is transfered.

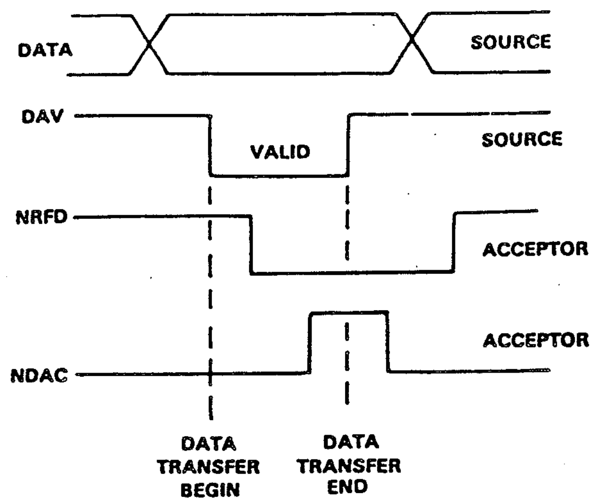


Fig. 2.7 **Handshake Sequence**

In the beginning of the transfer NRFD must be high indicating that all the accepting devices on the bus are ready to receive data. At the same time the NDAC line has to be low from the previously sent data byte. First when these conditions are met the DAV line can become set to low. Otherwise the source is waiting for the correct status. But because of the possibility of bus hang up, some controllers have time-out routines. So these may become activated and the data transfer stops, when the delay is to long to reach the correct status of the NRFD and NDAC lines.

After the DAV line has gone low the NRFD line also goes low. The NDAC then goes high when all the accepting devices on the bus have received the data, i.e. when the slowest of them have detected it. This is something that caused trouble in my program for scanning the isotope masses, since the large number of points (4096), that I tried to measure, had to take as short time as possible. But when I installed a teslameter, to measure the magnetic field, the total bus delay became more than 2 seconds per measurement. That made me skip recording the magnetic field and disconnect the teslameter.

2.2.6 The Data Lines

2.2.6.1 Introduction. - There are 8 data lines in the IEEE bus called DIO1 through DIO8. These are bidirectional lines for transmitting and receiving bit-parallel, byte-serial code. Low is true as before. The commands that may become sent, summarized in table 2.2, are grouped into uniline, multiline and device-dependent commands.

Uniline commands are achieved be setting the single corresponding bus management line low as already described in section 2.2.4.

Multiline commands are general bus commands that are sent by the controller with ATN low. As table 2.2 shows they can further be devided into universal, addressed and unaddressed commands.

Table 2.2

IEEE Bus Command Summary

Command Type	Command	State of ATN Line*	Comments
Uniline	REN (Remote Enable)	X	Set up for remote operation.
	EOI (End or Identify)	X	Sent by setting EOI low.
	IFC (Interface Clear)	X	Clears Interface
	ATN (Attention)	Low	Defines data bus contents.
	SRQ (Service Request)	X	Controlled by external device.
Multiline			
Universal	DCL (Device Clear)	Low	Returns device to default conditions.
	SPE (Serial Poll Enable)	Low	Enables serial polling.
	SPD (Serial Poll Disable)	Low	Disables serial polling.
Addressed	SDC (Selective Device Clear)	Low	Returns unit to default conditions.
	GTL (Go to Local)	Low	Returns to local control.
	GET (Group Execute Trigger)	Low	Triggers device for reading.
Unaddress	UNL (Unlisten)	Low	Removes all listeners from bus.
	UNT (Utalk)	Low	Removes all talkers from bus.
Device-dependent		High	Programs

*X = Don't Care

The universal commands: DCL, SPE, SPD, PPU, LLO and also the unaddressed commands: UNL, UNT are common for all bus devices and therefore do not need any addressing, whilst the addressed commands SDC, GTL, GET, PPC, TCT are only directed towards the selected talker or listeners.

Device-dependent commands are as the name says depending on what device is beeing used, e.g. the electrometer (see section 2.10) - telling it what range it should have, or the digital/analogue converter (see section 2.5) - demanding it to read the digital data.

Figure 2.8 shows how the multiline commands as bytes are interpreted in the ASCII code. Even addresses are command bytes. The code uses 7 bits, i.e. the data lines DIO1 to DI07. The 8:th bit, i.e. line DIO8 is don't care. This gives 128 different code words. They are devided into:

* PCG (Primary Command Group) with codes 00-5F Hex (0-95 Dec).

* SCG (Secondary Command Group) with codes 60-7F Hex (96-127 Dec)

* ACG (Addressed Command Group) with codes 00-0F Hex (0-15 Dec)

* UCG (Universal Command Group) with codes 10-1F Hex (16-31 Dec)

* LAG (Listen Address Group) with codes 20-3F Hex (32-63 Dec)

* TAG (Talker Address Group) with codes 40-5F Hex (64-95 Dec)

As you can see the PCG is devided into the ACG, UCG, LAG and TAG. Some devices need a secondary address, derived from the SCG, except for the ordinary primary address found in LAG or TAG. This is not the case for the equipment that I have used in my system.

Here comes a breif decription of the commands:

2.2.6.2 GTL (Go To Local) - The GTL command with code 01 Hex tells an addressed device to go to a local state, where you can control the instrument with the buttons on it instead of program if over the bus.

2.2.6.3 SDC (Selective Device Clear) - This command with code 04 Hex makes a specified unit go to default conditions. It is the same as DCL except for the fact that only the addressed unit(s) is (are) affected.

2.2.6.4 PPC (Parallel Poll Configure) - PPC with code 05 Hex tells an instrument to send its status byte.

Bits D_7 D_6 D_5 D_4					X 0 0 0	COMMAND	X 0 0 1	COMMAND	X 0 1 0	PRIMARY ADDRESS	X 0 1 1	PRIMARY ADDRESS	X 1 0 0	PRIMARY ADDRESS	X 1 0 1	PRIMARY ADDRESS	X 1 1 0		X 1 1 1	
D_3	D_2	D_1	D_0	COLUMN – ROW↓	0 (A)	0 (B)	1 (A)	1 (B)	2 (A)	2 (B)	3 (A)	3 (B)	4 (A)	4 (B)	5 (A)	5 (B)	6 (A)	6 (B)	7 (A)	7 (B)
0	0	0	0	0	NUL		DLE		SP	0	0	16	@	0	P	16			p	
0	0	0	1	1	SOH	GTL	DC1	LLO	!	1	1	17	A	1	Q	17	a		q	
0	0	1	0	2	STX		DC2		"	2	2	18	B	2	R	18	b		r	
0	0	1	1	3	ETX		DC3		#	3	3	19	C		S	19	c		s	
0	1	0	0	4	EOT	SDC	DC4	DCL	$	4	4	20	D	4	T	20	d		t	
0	1	0	[illegible]	5	ENQ	PPC*	NAK	PPU*	%	5	5	21	E	5	U	21	e		u	
0	1	1	0	6	ACK		SYN		&	6	6	22	F	6	V	22	f		v	
0	1	1	1	7	BEL		ETB		'	7	7	23	G	7	W	23	g		w	
1	0	0	0	8	BS	GET	CAN	SPE	(	8	8	24	H	8	X	24	h		x	
1	0	0	1	9	HT	TCT*	EM	SPD	)	9	9	25	I	9	Y	25	i		y	
1	0	1	0	10	LF		SUB		*	10	:	26	J	10	Z	26	j		z	
1	0	1	1	11	VT		ESC		+	11	;	27	K	11	[	27	k		{	
1	1	0	0	12	FF		FS		,	12	<	28	L	12	\	28	l		:	
1	1	0	1	13	CR		GS		–	13	=	29	M	13	]	29	m		}	
1	1	1	0	14	SO		RS		.	14	>	30	N	14	^	30	n		~	
1	1	1	1	15	SI		US		/	15	?	UNL	O	15	–	UNT	o		DEL	

ADDRESSED COMMAND GROUP (ACG)

UNIVERSAL COMMAND GROUP (UCG)

LISTEN ADDRESS GROUP (LAG)

TALK ADDRESS GROUP (TAG)

PRIMARY COMMAND GROUP (PCG)

SECONDARY COMMAND GROUP (SDC)

*PPC (Parallel Poll Configure), PPU (Parallel Pull Unconfigure), LLO (Local Lockout), and TCT (Take Control) not implemented by Model 4853.

NOTE: D_0 = DIO1...D_7 = DIO8
X = Don't Care

Fig. 2.8 Command Codes

2.2.6.5 GET (Group Excecute Trigger) - The GET command with code 08 Hex triggers instruments with this option[1] for sending bytes on the bus. These instruments must have been addressed as talkers.

2.2.6.6 TCT (Take ConTrol) - TCT with code 09 Hex can hand over the controller capability to another device. It has not been used in my system, since I only use the computer as a controller.

2.2.6.7 LLO (Local Lock Out) - LLO with code 11 Hex takes away the possibility to use the buttons on all the instruments, insted the controller has got the monopole.

2.2.6.8 DCL (Device Clear) - This command with code 14 Hex returns all units to default conditions.

2.2.6.9 PPU (Parallel Poll Unconfigure) - PPU with code 15 Hex rounds of all PPC's.

2.2.6.10 SPE (Serial Poll Enable) - SPE with code 18 Hex tells all devices to send their status.

2.2.6.11 SPD (Serial Poll Disable) - SPD with code 19 Hex finishes the SPE.

2.2.6.12 UNL (UNListen) - UNL with code 3F Hex or ASCII '?' removes all listeners from the bus.

2.2.6.13 UNT (UNTalk) - UNT with code 5F Hex or ASCII '_' removes the talker from the bus.

2.2.6.14 Addressing. - To illustrate the addressing let us take the Keithley Model 485 Electrometer (figure 1.6 and section 2.6). It has got primary address 22 and no secondary address. To address it to listen, you make the controller send its listener address code on the bus. This you find in figure 2.8, within the LAG area, in the right of the columns named "PRIMARY ADDRESS". 22 makes the ASCII '6' character or the bits X011 for D7-D4 and 0110 for D0-D3, i.e. 011 0110 binary, which converted becomes 36 hexadecimal or 54 decimal. This is the listener address for the electrometer. The talker address is found similarly in the TAG area to be ASCII 'V', 101 0110 Bin, 56 Hex or 86 Dec. Another way to derive the addresses code byte is to decimally add 32, for the listener address, and 64 for the talker address, to the primary address.

[1]The Keithley Model 485 Electrometer in my system has the GET option but others as well - to trigger it. See section 2.6.

2.3 The CAMAC Crate

A CAMAC crate is drawn in figure 1.6 among the other devices. It is an electronics rack with 25 slots, where you can plug in different kinds of electronic equipment. The contact pins in the back of each slot are both giving power support and are connected to pins in the other slots, thereby making up a sort of bus system.

The instruments in a CAMAC crate are specially developed modules for its standard, and include such devices as bus interfaces (see models 8901 and 2132 in sections 2.4 and 2.6 respectively), multiplexers, digital/analogue converters (like model 9091 in section 2.5), analogue/digital converters, e.t.c.

2.4 The CAMAC to IEEE Interface

2.4.1 Front Panel

The LeCroy model 8901 CAMAC to GPIB interface is a CAMAC device taking up the two slots 24 and 25 in the CAMAC crate (see figure 1.6). It requires a power of 6V at 1.2A. An illustration of the front panel is given in figure 2.9.

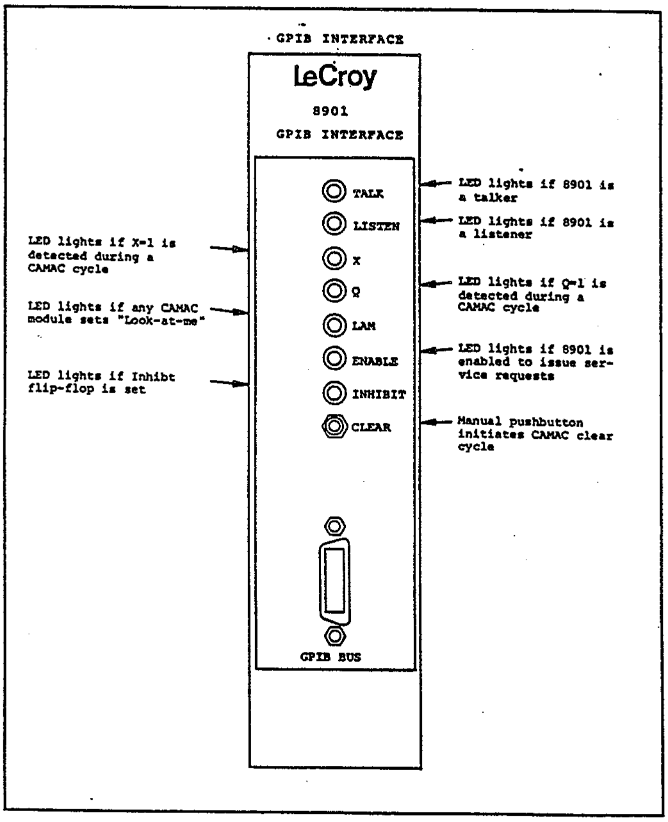


Fig. 2.9 Front panel of the 8901 CAMAC to GPIB interface.

Table 2.3

Front Panel LED Functions:	
TALK:	Lights when 8901 is a talker.
LISTEN:	Lights when 8901 is a listener.
X:	Lights when a command accepted signal is generated within the CAMAC crate.
Q:	Lights when a Data Valid signal is generated within the CAMAC crate.
ENABLE:	Lights when 8901 is enabled to carry out service requests.
LAM:	Lights when any CAMAC module sets "Look-at-me", a service request.
INHIBIT:	Lights when CAMAC dataway is inhibited.
Front Panel Controls:	
CLEAR:	Manual pushbutton for initializing data and state registers in all CAMAC modules.

In table 2.3, as well as in figure 2.9, the meaning of the indicators and the push-button on the model 8901 is explained. Especially the X and Q LED lights are displaying the handshake process of the internal bus system, solely working within the CAMAC crate.

There is also an IEEE connector on the the module's front. This has been coupled in my system as figure 1.6 shows: both to the Olivetti PC and to the Electrometer. The former acts as the controller and can address the 8901 to talk or listen. The TALK and LISTEN LED indicators are then correspondingly illuminated.

2.4.2 Addressing

Figure 2.10 gives a hint of the three levels of addressing.

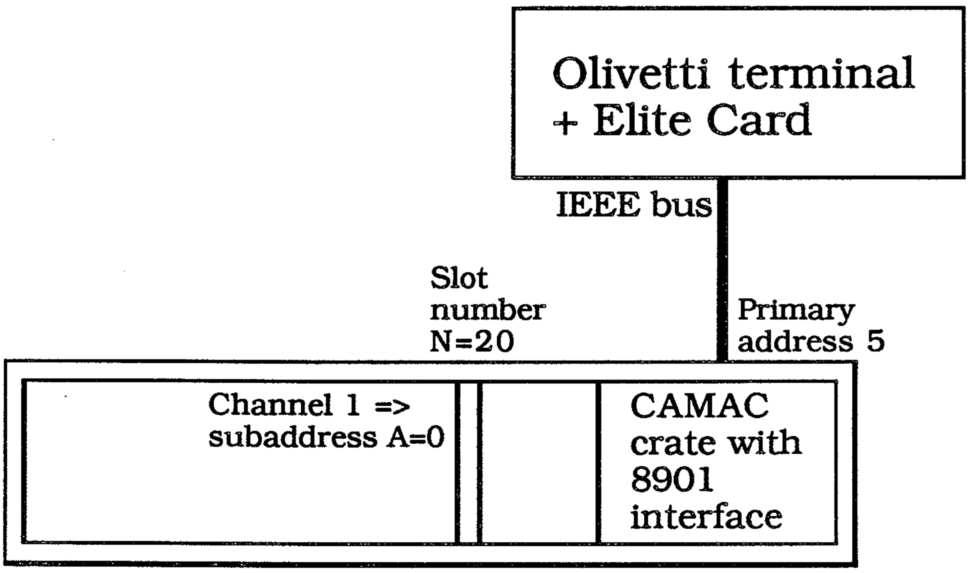


Fig. 2.10 The three CAMAC address levels.

First comes the IEEE primary address 5. Then the module dependent subaddress with the notation A.[1] Finally the third address is simply the slot number N from 1 to 23 (since 24 and 25 are used up by the 8901 interface).

2.4.3 The 8901 Module Registers

The 8901 module has 7 registers to generate the internal CAMAC cycles: N, A, F, W, C, Z, I. After the 8901 interface has been commanded to listen over the IEEE bus, the first sent device dependent command data byte is stored in the F register (see how this is done in table 2.4). This register contains the CAMAC control information, such as read from or write to a specific module in the crate. The next byte is accepted by the A subaddress register, and the third byte by the N station number register. Their functions have just been described in section 2.4.2. If the F register was programmed to contain the code 16 for write, another 3 bytes are accepted and stored in the 24 bit large W register (see table 2.5). Important to note is that different CAMAC modules may assign different meaning to the F and A codes (see tables 2.8 and 2.9).

The IEEE system controller can terminate the loading process after any number of bytes have been transferred by commanding the 8901 interface to enter talk mode, by issuing an IFC command, or by issuing an unlisten command.

The 8901 can send clear (C), initialize (Z), or inhibit (I) on the internal CAMAC dataways if the corresponding 1 bit registers are programmed to be set. This is done by sending one byte of data using the format 2 or 3 shown in table 2.4.[2] The C and Z registers are cleared after the completion of the the next CAMAC cycle, whilst the I register remains unaffected until it is programmed off.

2.4.4 The Read Procedure

If a read procedure is to take place, the first byte is sent on the IEEE bus as soon as the 8901 has been commanded to talk. The next byte then follows when the listeners have accepted the first one, signaled by the IEEE handshake line NDAC. The interface can be programmed for 2,3, or 4 byte return. Default is 3 bytes. Table 2.6 gives the read return formats.

---

[1]For instance the D.A.C. has 16 analogue output channels, so the corresponding subaddresses become 0 to 15.

[2]An example of this is the code $100011_2=35_{10}$ which sets both C and Z.

Table 2.4

FIRST BYTE

FORMAT	D8	D7	D6	D5	D4	D3	D2	D1
1	X	0	0	F16	F8	F4	F2	F1
2	X	1	0	X	Inhibit	No X Enable	No Q Enable	LAM Enable
3	X	0	1	X	X	X	C	Z
4	X	1	1	X	HI SPEED TRANSFER	4 Word Return	3 Word Return	2 Word Return

X: DON'T CARE

1: SET DESCRIBED STATE

Table 2.5

BYTE	D8	D7	D6	D5	D4	D3	D2	D1
2	X	X	X	X	A8	A4	A2	A1
3	X	X	X	N16	N8	N4	N2	N1
4	W8	W7	W6	W5	W4	W3	W2	W1
5	W16	W15	W14	W13	W12	W11	W10	W9
6	W24	W23	W22	W21	W20	W19	W18	W17

Table 2.6

	BYTE	D8	D7	D6	D5	D4	D3	D2	D1
2 WORD RETURN	1	R8	R7	R6	R5	R4	R3	R2	R1
2 WORD RETURN	2							Q	X
3 WORD RETURN	1	R8	R7	R6	R5	R4	R3	R2	R1
3 WORD RETURN	2	R16	R15	R14	R13	R12	R11	R10	R9
3 WORD RETURN	3							Q	X
4 WORD RETURN	1	R8	R7	R6	R5	R4	R3	R2	R1
4 WORD RETURN	2	R16	R15	R14	R13	R12	R11	R10	R9
4 WORD RETURN	3	R24	R23	R22	R21	R20	R19	R18	R17
4 WORD RETURN	4							Q	X
DEFAULT	1	R8	R7	R6	R5	R4	R3	R2	R1
DEFAULT	2	R16	R15	R14	R13	R12	R11	R10	R9
DEFAULT	3	R24	R23	R22	R21	R20	R19	R18	R17

Table 2.7

Service Request Status Bytes

BYTE	D8	D7	D6	D5	D4	D3	D2	D1
1	X	1					Q	X
2	X	1	L6	L5	L4	L3	L2	L1
3	X	1	L12	L11	L10	L9	L8	L7
4	X	1	L18	L17	L16	L15	L14	L13
5	X	1		L23	L22	L21	L20	L19

2.4.5 Special Features

Some special features with the model 8901 are that it monitors the status of 23 "Look-At-Me" (LAM) signals, it can issue service request, and it has got a high speed transfer mode.

The controller can program LAM to be set by a module under conditions that depend on that specific module beeing used.[1] The LAM LED light is shining when any LAM has been prompted.
The causes of the SRQ (Service ReQuest) signal to be transmitted are a detection of LAM, or no Q or X responses given after the execution of a CAMAC cycle. But this is only the case if the "No X Enable", "No Q Enable", or "LAM Enable" bits are set in the F register (see table 2.4). Serial request ends when the 8901 is polled. Then the status is sent (see table 2.7)

If high speed transfer mode is on and the 8901 is commanded to be a talker, it is automatically and continuously sending new series of data. A new cycle is initiated when the last byte of the last data series is under process, making the time between transmissions only about 2 μsec. The mode continues until a Q=0 is detected.

2.5 The Digital/Analogue Converter

The Nuclear Enterprises Multidac Type 9091 Digital/Analogue Converter (D.A.C.) is a single slot CAMAC module. It is doing what the name says; it converts a digital signal into an analogue output. The output is a voltage, which in my system (see figure 1.6) is in the range between 0V and +10V. This is proportional to the value of the digital input; made up of 12 bits, having the values 0 to 4095.

The D.A.C. has got 16 output channels, with subaddresses A(0) to A(15).[2] I used channel 1, with subaddress A(0). I also used slot position 20 in the CAMAC crate.

To send a digital input to the D.A.C., you first have to address the CAMAC 8901 interface to be a listener. Then since the command code for write is F(16) the following sequence makes up a digital input signal:

F(16)•A(0)•N(20)•byte1•byte2

Byte1 is loaded into registers W1-W8, whilst the first four bits of byte2 are stored in W9-W12 (see table 2.5). Now almost instantly the channel 1 output responds with the converted analogue voltage. An

---

[1]The HV 4032-to-CAMAC interface, encountered in section 2.6, may for instance set LAM when it has completed the storage of a HT value. This I use in my program (see section 3.2).

[2]'A(0)' is a notation meaning that the A register is programmed to contain 0.

indicator light on the front panel of the D.A.C. also becomes illuminated.

To read back the setting of channel 1, the following must be commanded over the bus:

F(0)•A(0)•N(20)

Then the talker address of the CAMAC has to be is put on the IEEE bus, after which the 12 digital input bits are sent as bytes, according to the format of table 2.6.

Table 2.8 summarizes some parameters for the D.A.C.

Table 2.8

MECHANICAL SPECIFICATION

Single width CAMAC module.

WEIGHT

0.75 kg approx.

OPERATING TEMPERATURE RANGE

0°C to +40°C

POWER REQUIREMENTS

+6 V at 550 mA +24 V at 125 mA -24 V at 150 mA	from the mandatory CAMAC power supply lines, excluding output current.

FRONT PANEL CONNECTORS

CH1 - CH16	Analogue Outputs 1 - 16
Connector type:-	Coaxial 50 series (e.g. Lemo 00250)

FRONT PANEL INDICATORS

LP101	'Enable'	LED illuminates when any of the analogue outputs is enabled by an overwrite function.

USE OF DATAWAY

Commands

F(16) (A(0) - A(15))	Overwrite analogue output programming register. CH1 corresponds to A(0). CH2 corresponds to A(1) etc. The data consists of 12 bits on W1 - W12. An overwrite command enables the respective channel.
F(0) (A(0) - A(15))	Read analogue output programming register. The data is read via R1 - R12. An optional link enables the enable status to be read via R16. A logic 1 indicates that the channel has not yet been enabled by an overwrite command.

Control

Z or C	Disable all outputs. All outputs are switched to zero volts until overwritten by an F(16) command. Strobed by S2 and OR gated with Power On Reset.

Status

X Response	Generated for F(16) (A(0) - A(15)) and F(0)(A(0) - A(15)). Inhibited for all other commands.
Q Response	Generated for F(16) (A(0) - A(15)) and F(0) (A(0) - A(15)). Inhibited for F(16) until any one channel is enabled.

Strobes

S1	Gated with F16 to perform overwrite and set Enable.
S2	Gated with Z or C to disable all channels.

ANALOGUE SIGNALS

Output Voltage Range:	0 V to +10 V 0V to +5 V 0 V to -5 V 0 V to -10 V -10 V to +10 V -5 V to +5 V +10 V to -10 V +5 V to -5 V Selectable by printed circuit board links.
Max. Output Current:	±20 mA
Short Circuit Protection:	indefinite
Programming Resolution:	12 bits
Max. Offset Variation Channel - Channel:	±500 μV at 20°C
Max. Voltage Drift:	10 ppm/°C of full scale
Typical Voltage Drift:	5 ppm/°C of full scale
Output Regulation:	150 μV per ma
Rise and Fall Time:	200 μs for 10 V transition measured 10% - 90% f.s.
Update Time:	500 μs for 10 V transition measured 10% - 90% f.s. from the time when the channel register is overwritten
Differential Non-linearity:	±½ LSB max.

2.6 The HV4032 to CAMAC Interface

2.6.1 The Serial Bus Driver

The LeCroy 2132 HV4032 to CAMAC interface is present as a link between the CAMAC and the five LeCroy HV 4032A, 32 photomultiplier high voltage system crates[1] (see next section), in turn driving the beam line elements. Actually it is another bus driver making the beam line control system (but not the mass spectrometer system - see figure 1.6) in total having three separate buses: the IEEE bus, the CAMAC dataway, and this 2132 to HV4032 serial bus. Figure 2.11 displays the trio.

The model 2132 takes one slot position in the CAMAC; number 23, next to the 8901 interface was the choice here. It can communicate both ways with up to fifteen HV4032 power crates, arranged in any combination in a so called daisy chain.

The bus consists of 10 bus lines, with 20 mA current loops, divided into the following pairs; two serial data pairs, two bus priority pairs, and one pair to differentiate between CAMAC and the so called TTY operation (the later is not used here). The maximum length of the cable before the first HV4032 mainframe[2] is 300 meters, and between the HV4032 crates 30 meters.

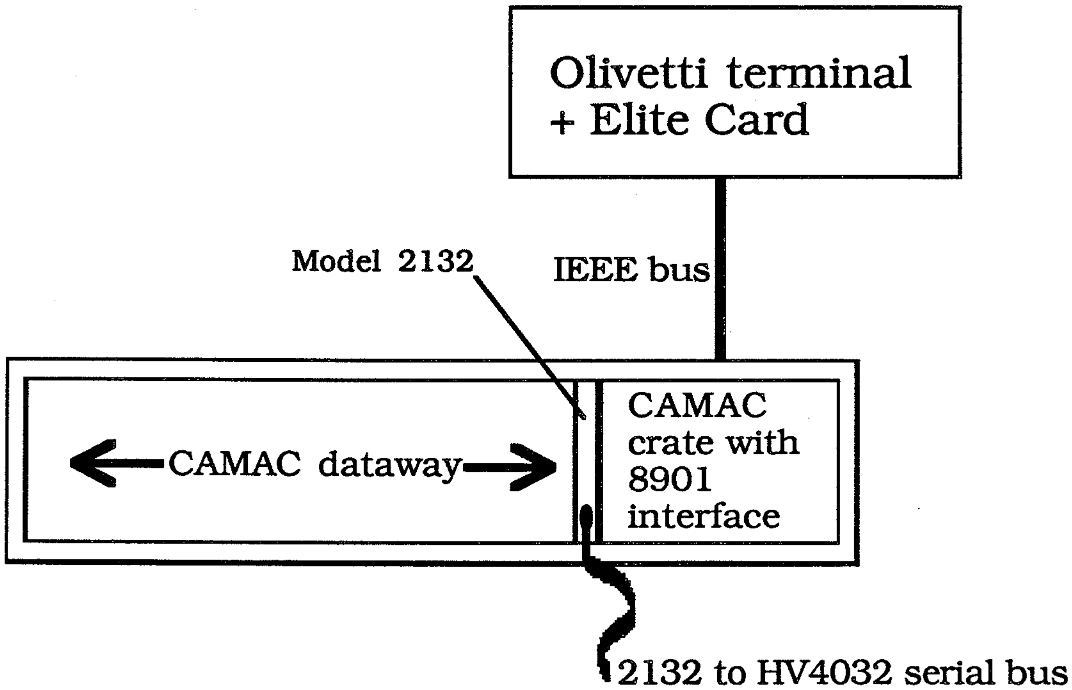


Fig. 2.11 The three separate buses.

[1]There are currently five such crates installed: MF #11, MF # 13, MF # 14, MF # 15 and MF # 16 (see figure 1.6). MF # 10 is a separate device currently not bus controlled.
[2]"Mainframe' is used as a synonyme for the HV4032 power crate.

Data are transmitted at 2400 BAUD with a modified ASCII code. 16 bit words are sent serially as three bytes, each with its own parity bit for checking transmission errors. There is also a "Look-At-Me" (LAM) error handling built in. High priority LAM (L1) is set to signal error, whereas low priority LAM (L2) is set after the completion of the data transmission to the 2132. These LAM's may be detected through the CAMAC read (R1=L1, R2=L2) which uses code F(0) (see table 2.9).

2.6.2 Front Panel

The 2132 front panel looks like figure 2.12. Two LEMO type connectors, one serial I/O connector (AMP Model 201298-1), and a LED light are provided.

The DISABLE input can externally be clamped to ground to inhibit low priority LAM (L2) and Q responses to be given. This eliminates computer overhead for routine high voltage maintainance at critical times.

ERROR is an open collector 300 mA clamp to ground when a failure is detected at an HV4032. It is intended for use with an external alarm. Maximum VCC is 30 volts.

The HVDC-14 cable to the HV4032 mainframes is connected to the Serial I/O connector at the bottom of the 2132 front panel.

A LED is illuminated each time the 2132 is addressed.

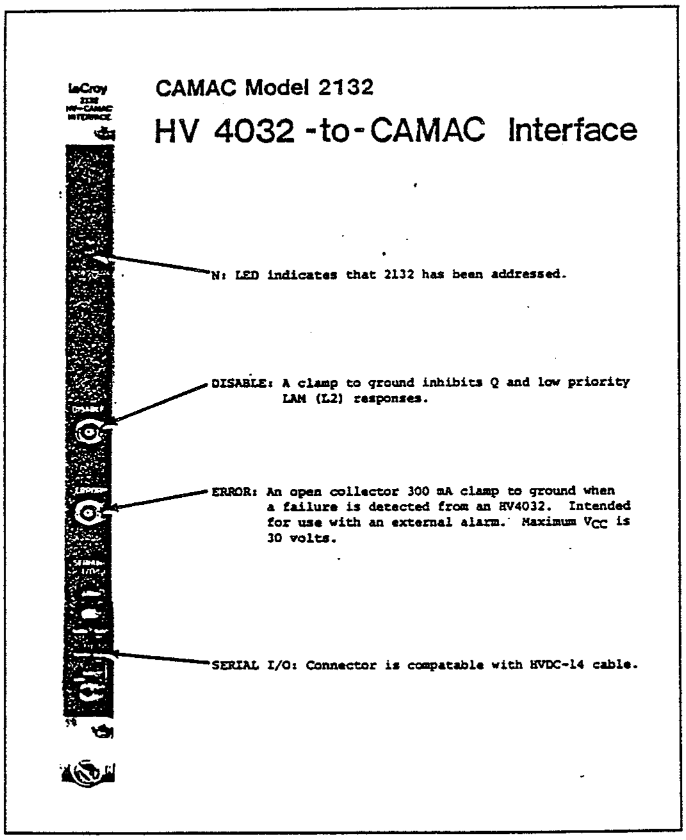


Fig. 2.12 Front panel of the HV4032 to CAMAC interface.

2.6.3 The Serial Bus Communication

All communcation from the 2132 to the HV4032 (output) will be called "commands", and all communication from the HV4032 to the 2132 (input) will be called "responses" (see figure 2.13). The commands are mostly followed by a response from the addressed HV4032 mainframe. Those which have a defined response are called "handshake commands" (see table 2.10). Any non-handshake command can optionally generate a "finished response" if this is programmed so. The responses will only be activated by a command, with the only exception of the response indicating a failed channel, which is spontaneously sent.

Figure 2.14 shows schematically how the model 2132 works. Because CAMAC is so much faster than the serial bus, 40 word memories (FIFO 1 and FIFO 2) are employed to buffer both the CAMAC R and W lines. That is also the link to the CAMAC dataway. In the other end towards the serial bus there is a Universal Asynchronous Receiver Transmitter (UART).

The information transferred at 2400 BAUD between the 2132 and the HV4032 mainframes is configured the way that figure 2.15 shows. The responses are 18 bit long and broken three 6-bit characters in special laches, after they have been serially received in the UART. The UART also adds one start bit, two stop bits, and a parity bit for each of the three ASCII caracters (bytes) , whereafter the control circuitry adds a flag bit allowing the system to perform checks. L1 and L2 are used to set the LAM status on (but not for resetting it), whereas the other 16 original response data bits (D1-D16) constitute the actual data word.

Data transmission from the 2132 to the HV4032 modules for commands is similar to the response receiving, except for that the L1 and L2 bits are not used, and that the transfer is considerably slower. The error checking (parity, framing, etc.) requires approximately 100 msec between characters, making the total transfer time approximately 250 msec/word (see figure 2.15). This timing is controlled from within the 2132.

Conversely the transfer back to the 2132 takes only 15 msec/word.

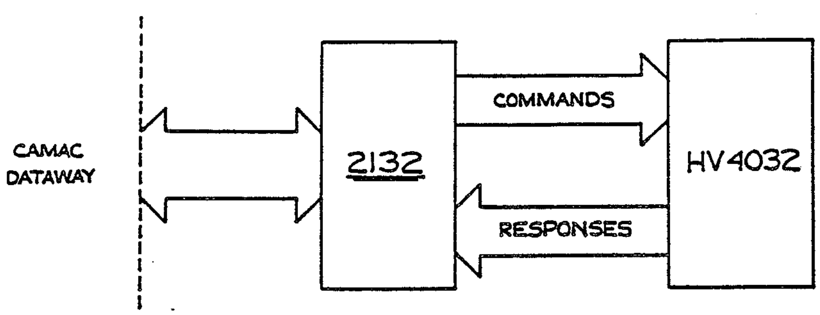


Fig. 2.13 How the 2132 interface communicates.

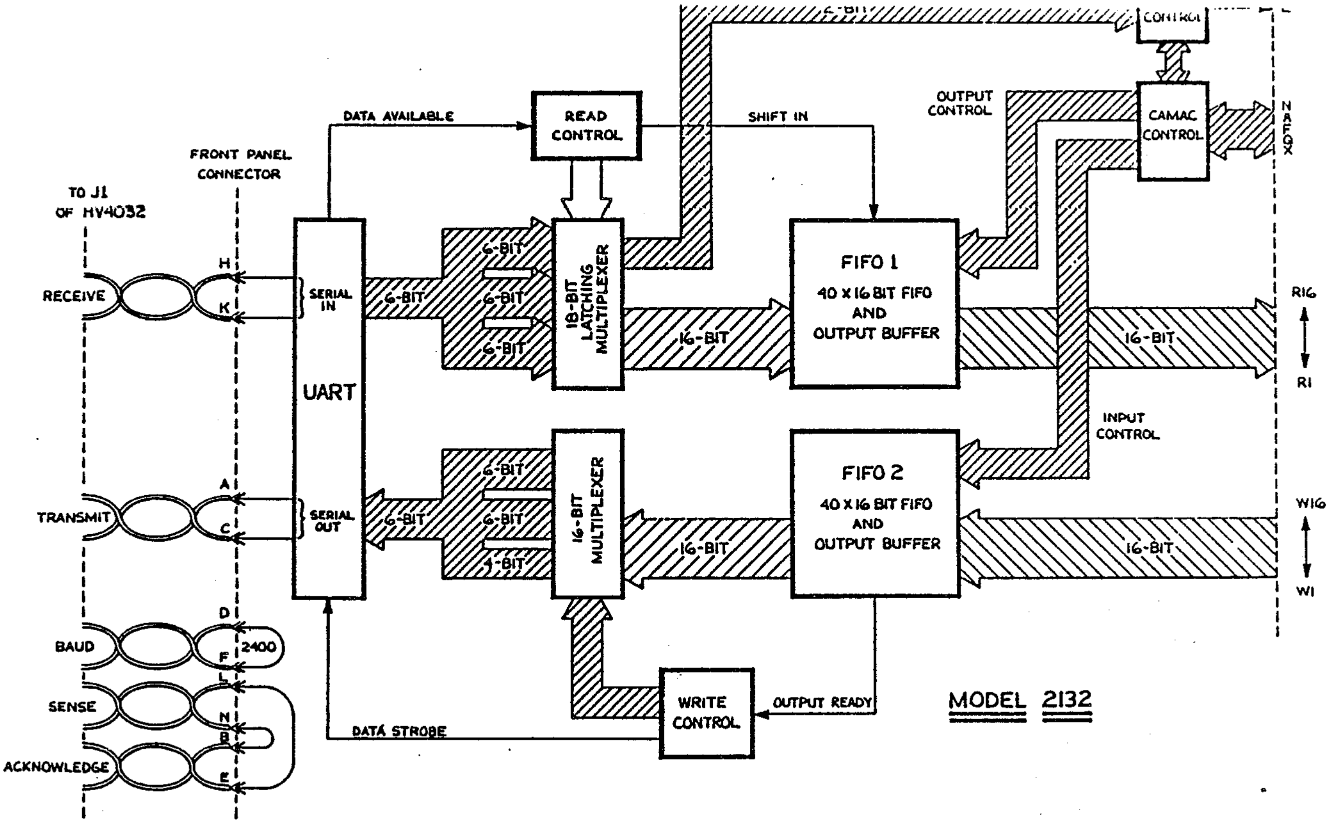


Fig. 2.14 How the model 2132 works.

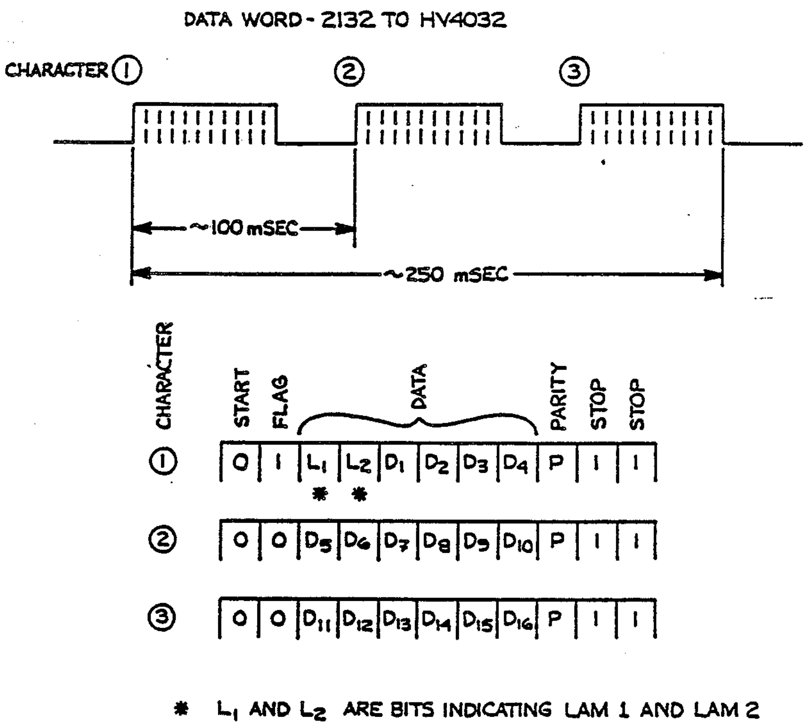


Fig. 2.15 Configuration of the serial data.

2.6.4 2132 Programming

The following syllables comprise the 16-bit data word:

* The mainframe address M. M=0 is used for addressing all units on the daisy chain (6-bits).

* The channel number C (6-bits).

* The voltage V (12-bits)

* The control tag T (4-bits).

* The software switch S (1-bit right justified in a 6-bit field).

The "commands" and "responses" are indicated in table 2.10. The commands are written into the 2132 via CAMAC F(16) commands (see table 2.9). If, as in my system, the 2132 is located in slot N=23 the following sequence sent on the IEEE bus does this:

F(16)•A(0)•N(23)•byte1•byte2•byte3

Byte1 and byte2 contain what is configured in figure 2.16, whereas byte3 is always zero. Note that the commands and the responses are fully specified by the value of the control tag. Each response is also followed by a LAM 2 (L2) setting. A finished response may be optionally used, but I have not done so in my system.

The read process is to first address via the F(16) command according to the sequence above (see also figure 2.16), selecting both mainframe MF and channel CH address[1] as well as the proper command control tag T (3,4,8,9 or 10 - see table 2.10), then to wait for LAM 2, whereafter the response value is stored in the input buffer. The input buffer is then read with:

F(2)•A(0)•N(23)

The input buffer can be cleared either by reading, using the F(2) command until the 8901 Q indicates zero, or by using F(9)•A(1), something that I prefere (see table 2.9).[2]

The command T=0 to address, followed by command T=1 with the value in volts (max 3000V), is an example of the write principle.

Status request to see what the ON/OFF S switch is, is done with command T=9 followed by response T=5, read according to the principle above. Command T=5 can set the ON/OFF status of the HV4032 crates, but it can take as much as 20 seconds before this is completed.

---

[1]This is actually the forth and fifth address levels (compare with section 2.4.2).

[2]The Q=1 marks that a 16 bit word (i.e. IEEE bytes byte1 and byte2 in the above format) have been accepted by the 2132 unit.

Caution should be taken so that the 40 word long FIFO buffers are not read or overwritten too fast, thus passing their limits and causing errors.

Table 2.9

SUMMARY OF CAMAC FUNCTIONS

F(0)•(A(0)+A(1))•N	Read LAM Register (R1=L1, R2=L2)
F(2)•(A(0)+A(1))•N	Read and advance input buffer (data valid if Q=1)
F(9)•A(0)•N	Clear buffers, L1 and L2
F(9)•A(1)•N	Clear input buffer
F(16)•A(0)•N	Write into output buffer (Q=1 if word is accepted at output buffer. Data transfer will then procede to the HV4032)
F(27)•A(0)•N	Test L1
F(27)•A(1)•N	Test L2
F(10)•A(0)•N	Clear L1
F(10)•A(1)•N	Clear L2
F(26)•A(0)•N	Enable L1
F(26)•A(1)•N	Enable L2
F(24)•A(0)•N	Disable L1
F(24)•A(1)•N	Disable L2
Z•S2	Clear L1, clear L2, clear buffers, enable L1, disable L2
C•S2	Clear L1, clear L2, clear buffers

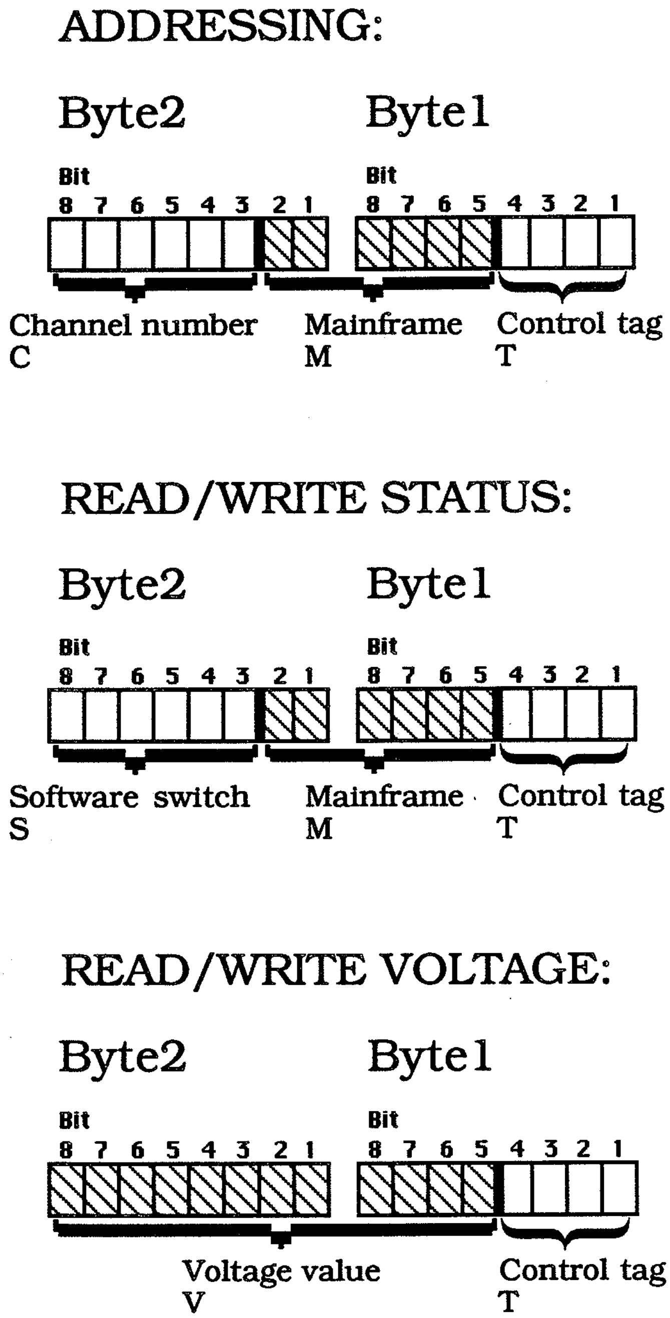


Fig. 2.16 The 2132 16-bit word configuration in IEEE bytes.

Table 2.10

TYPE ①	WORD ②	DESCRIPTION	RESULTING RESPONSE	INITIATING COMMAND
Command	C-M-0	Modify one channel voltage (Chan/Mainframe Select)	None (HV4032A waiting for a T = 1 command)	—
Command	V-1	Set Value (in volts). Must follow a T = 0, T = 2, or T = 7	T = 11 ⑧	—
Command	X-M-2	Modify all 32 channel voltages (Mainframe Select)	None (HV4032A waiting for 32 V-1 commands)	—
Command	C-M-3	Read true voltage of addressed channel	T = 3	—
Response	V-3	True voltage (actual rear panel output, in volts)	—	T = 3 or 4
Command	X-M-4	Read true voltage of all 32 channels	32 T = 3 words ④	—
Command	S-M-5 ⑨	HV ON/OFF Switch ⑨	T = 11 ⑧	—
Response	S-M-5 ⑨	HV ON/OFF Status ⑨	—	T = 9
Command	C-M-6	Restore addressed channel	T = 11 ⑧	—
Command	X-M-7 ③	Preset all 32 channels (Mainframe Select)	None (HV4032A waiting for a T = 1 command)	—
Command	C-M-8	Read set values of addressed channel	T = 8	—
Response	V-8	Set Value (in volts)	—	T = 8 or 10
Command	X-M-9	Status request	T = 9 and 5 ⑤	—
Response	C-M-9	Failed channel response	—	T = 9 ⑥
Command	X-M-10	Read set value of all 32 channels	32 T = 8 words ⑦	—
Command	S-M-11 ⑨	ENABLE/DISABLE *finished* (T = 11) response	None	—
Response	X-M-11	Finished Response	—	T = 1, 5, 6 or 14
Response	X-M-12	Parity error (HV4032A Receiver)	—	Any
Response	X-M-13	Overwrite error (HV4032A Receiver)	—	Any
Command	C-M-14	Zero addressed channel	T = 11 ⑧	—
Response	M-M-14 ⑩	Report of "Trip" condition (HV4032A only)	—	—
Response	X-X-15	Transmission Error (2132 Receiver)	—	Any

① *Command* is a word transfer from the 2132 to the HV4032A. *Response* is a word (or block of words) transferred from the HV4032A to the 2132 and is always accompanied by one L2 (attached to the last word if in a block).

② C = Channel address. CAMAC W16 to W11 for commands, R16 to R11 for responses (0 to 63 channels possible, 0-31 for HV4032A).
X = Don't care, CAMAC W16 to W11 for commands, R16 to R11 for responses.
S = Switch, W11 (where W16 to W12 should be 0) for commands, R11 (where R16 to R12 are don't cares) for responses. W11 or R11 equal to "1" is ON, equal to "0" is OFF.
M = Mainframe address, CAMAC W10 to W5 for commands, R10 to R5 for responses (0 to 63 addresses possible, but there should be no more than 16 HV4032A mainframes per 2132).
V = Voltage, CAMAC W16 to W5 for commands, R16 to R5 for responses. (0 to 4095 possible, HV4032A limits at 3000).
Eq. 5 = Tag (T), CAMAC W4 to W1 for commands, R4 to R1 for responses.

' must be OFF for command to be accepted.

④ Expect 32 *true voltages*, channel 0 first, etc. with L2 on last (chan 31) data only.

⑤ Expect one T = 9 word, no L2, for each failed channel in module, followed by T = 5 with an L2.

⑥ Also can be transferred spontaneously at time of failure with an L2 and the high priority L1.

⑦ Expect 32 *Set Values*, channel 0 first, etc., with L2 on last (chan 31) data only.

⑧ Can be enabled or disabled by T = 11 command,

⑨ S = 1 is ON or ENABLED, S = 0 is OFF or DISABLED.

⑩ Spontaneous transfer due to panic off or interlock being asserted with L2 and the high priority L1.

2.7 The High Voltage System Crates

The five LeCroy HV 4032A, 32 photomultiplier high voltage system crates, connected on a daisy chain, and interfaced to the CAMAC as described in the previous section, are used as power supplies for the ISOLDE-2 beam line elements. Up to 32 channels are programmable in each crate (or mainframe as it has also been called). The tension ranges, that may be used in each crate, are of four types: 0 to +3.3 kV, 0 to -3.3 kV, 0 to +7 kV, and 0 to -7 kV. The minimum voltage regulation step is 1 Volt from about 100 Volts to maximum output.

The internal of a HV4032 crate uses a microprocessor to handle different tasks, e.g. the storage of each of the 32 channel settings. It also checks bit patterns after power up, since a battery pack has ensured the memory to keep its values for up to 24 hours without AC power. Another microprocessor task is to use an A.D.C (Analogue/Digital Converter) to regulate the HV outputs and display these readings on the front panel display. The CPU also monitors the serial bus, when the crate is in remote operation.

The front panel of the HV4032A is shown in figure 2.17 and the different buttons and indicators are explained in table 2.11.

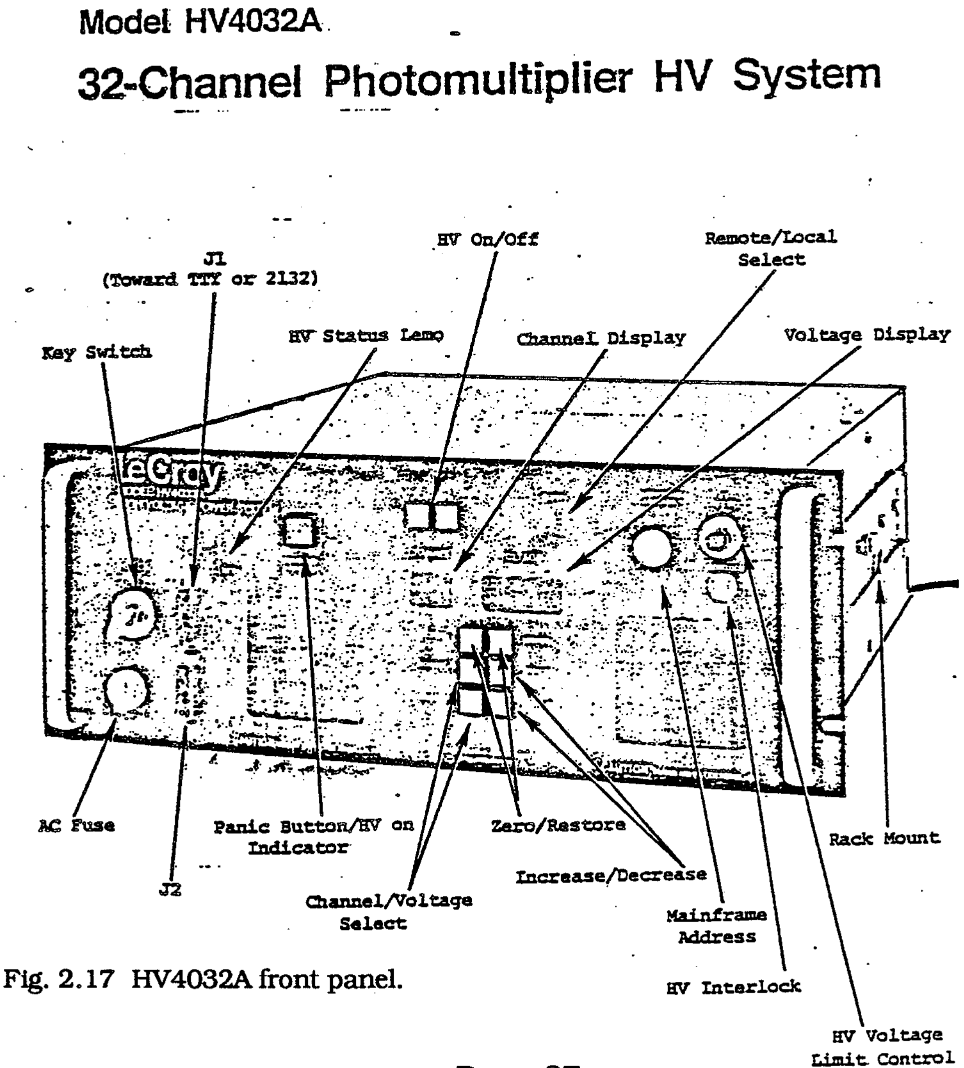


Fig. 2.17 HV4032A front panel.

Table 2.11

<u>Front Panel Controls</u>

POWER ON:	The POWER ON key switch controls the line power to the HV4032A.
ZERO/RESTORE:	The voltage setting of a channel can be zeroed using the ZERO button. The RESTORE button returns the voltage to its original value. This operation is not effective for channel 33. A RESTORE operation on a non zeroed channel is ignored.
CHANNEL/VOLTAGE AND INCREASE/DECREASE:	These buttons are used in conjunction with the Channel and Voltage displays. Increase and Decrease functions act upon whichever parameter button, CHANNEL or VOLTAGE, is depressed. Two buttons must be used together in order to achieve a change. For HV4032A7N and HV4032A7P pods, voltage programming is also used to set the current limit/trip. Release of the VOLTAGE button loads a digital buffer memory for the indicated channel with the displayed value. When the VOLTAGE button is depressed the demand for the indicated channel is displayed. Otherwise, the actual channel output is shown. If the supplies are turned on and the user modifies the setting of a channel, the output voltage will not change until the user releases the VOLTAGE button.
HV ON and HV OFF:	Used to turn the high voltage ON or OFF. The high voltage outputs will change at a rate of 1KV/sec for HV4032A1 pods or 2KV/sec for HV4032A7 pods. All channels change simultaneously. Unlike other front panel controls which may be ignored if the microprocessor is still responding to an earlier command, the HV ON and HV OFF button request is always responded to immediately or when the previous operation is completed.
PANIC OFF:	Front-panel momentary pushbutton unconditionally disables all channels. Outputs decrease $\leq$20 msec (1/e time for full load). Integral with HV ON and READY indicators. Trip reported to Model 2132 or TTY in remote operation.
INTERLOCK:	TTL compatible input with internal 4.7 kΩ pull-up resistor. Negative edge triggers a PANIC OFF. While the INTERLOCK is low, the READY indicator is extinguished and all commands will be ignored. Internal jumper option complements the disabling state of the INTERLOCK. Trip is reported in the REMOTE mode.
STATUS:	Front-panel Lemo connector. Clamp-to-ground when HV is present at rear connectors. Sinks $\geq$35 mA; up to 32 mainframes may be daisy-chained.
MAINFRAME ADDRESS:	The setting of this switch establishes the address of the mainframe. It is relevant only in the remote mode. Valid addresses are 1 to 16. If only one HV4032A system is employed, its address must be 16. For more than one, the last one in the daisy chain must be address 16. This is necessary for priority control of the serial data bus. The HV4032 employed address 63 for this purpose. Both models may be mixed if proper last module addressing is employed. Both 16 and 63 may not be present in the same daisy chain.
J1/J2 CONNECTORS: (Serial Port)	These connectors provide for daisy chaining up to 16 HV4032 and HV4032A units in a system. See the interconnection wiring diagram, Figure 3.1. The HVDC-14 data cable provides the necessary connections.
HV LIMIT SET:	A 10-turn potentiometer providing a hardware limit to the output of all channels. (The limit is adjusted by first setting a channel to the desired maximum, adjusting until the display just begins to indicate a decrease, and backing off slightly). Limit for 7 kV channels is approximately twice that of 3.3kV channels.
REMOTE/LOCAL:	This switch selects operation either from the front panel in LOCAL MODE or from the TTY or 2132 CAMAC Interface in REMOTE MODE.
FUSE INDICATOR:	Illuminated to indicate blown fuse. (Replace with 3AG SLO BLO type. 5A for 220 V operation 10A for 110 V.)
FANS:	Two high power fans provide airflow through the HV4032A and guarantee cool operation of the multi-channel pods. Serviceable air cleaners are provided to prevent an accumulation of dust inside the mainframe. These filters should be cleaned whenever they become visably dirty. The filter is easily pulled out of it's holder. Clean the filter by trapping it against a hard surface. Twice yearly, or when convenient, washing with mild soap and water is recommended. The filter should be dry before reinstallation.

2.8 The ISOLDE-2 Beam Line Elements

Each of the LeCroy HV4032A power crates use 16 channels 0 to +3.3 kV and 16 channels 0 to -3.3 kV. The allocations of the six different HV4032A's (mainframes) to the four different beam lines BL-1 to BL-4 are shown in figure 2.18 (compare with figures 1.1-1.3).

Figure 2.19 shows the three different beam line elements at ISOLDE-2: electrostatic quadrupoles (lenses), electrostatic deflectors (bends the beam), and electrostatic correction plates (adjusts the beam horisontally and vertically).

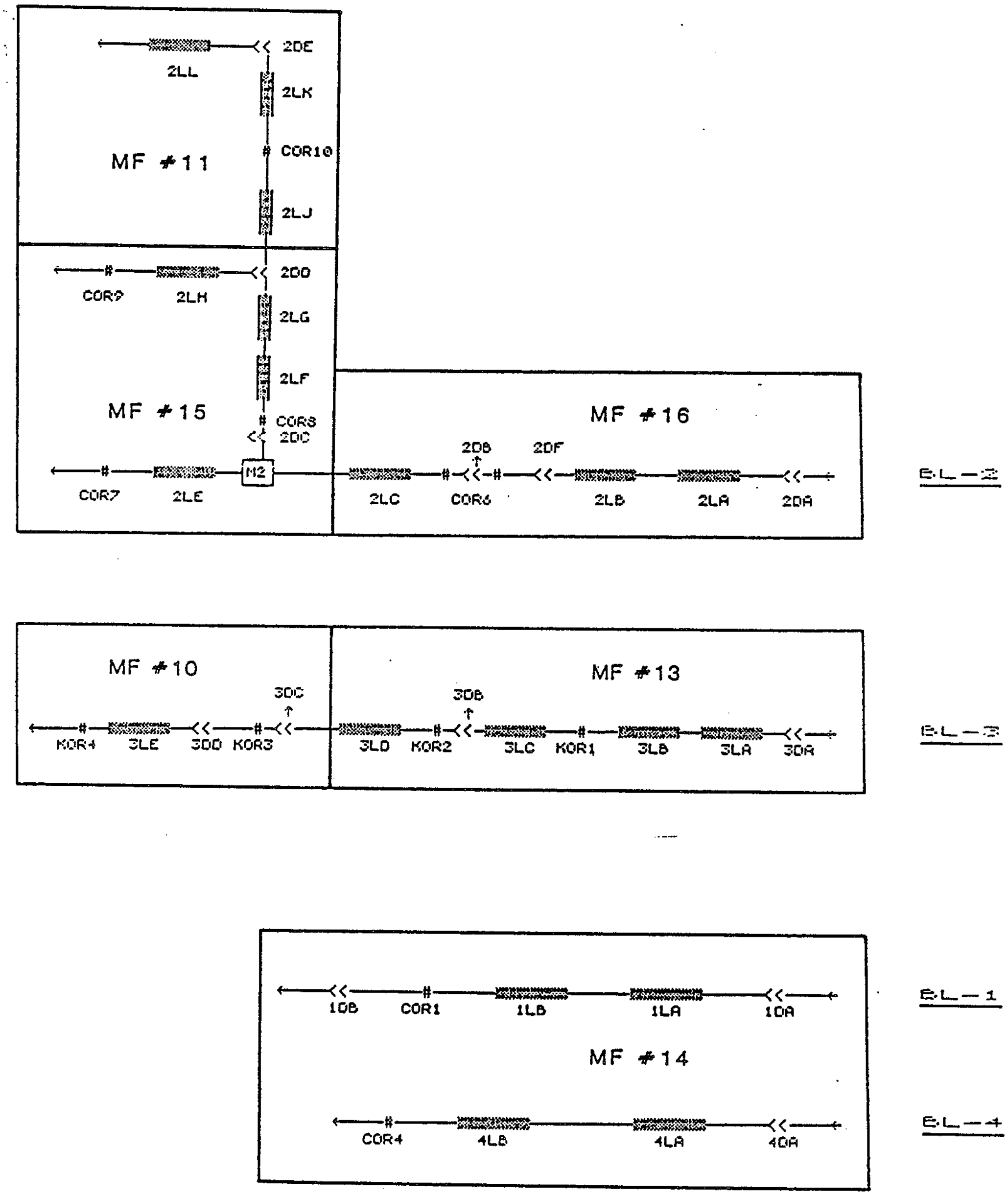


Fig. 2.18 LeCroy Mainframe distribution to ISOLDE-2 beam lines

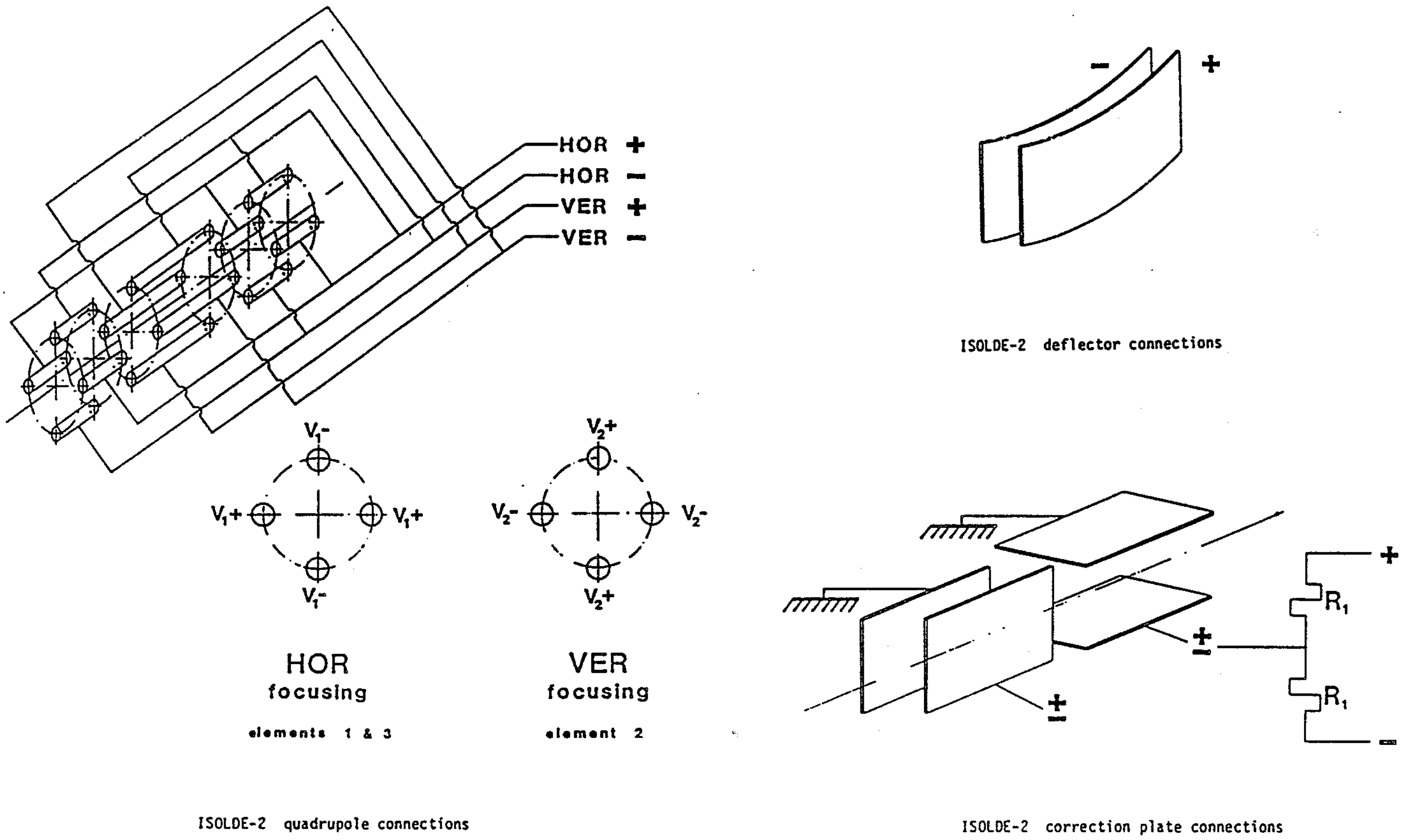


Fig. 2.19 ISOLDE-2 beam elements

A quadrupole uses 4 channels. There is one pair to adjust the horizontal focusing elements and one pair to adjust the vertical focusing elements. Each pair in turn uses 1 channel for the supply of a positive potential +V and 1 channel for the supply of a negative potential -V. Both the channels in a pair are set to the same value V.

A deflector uses 2 channels, set to the same value V. They are monitored by 1 positive and 1 negative power supply, delivering the tensions +V and -V respectively.

A correction element also uses 2 channels, that are monitored by 1 positive and 1 negative power supply. The outputs are connected to one of the correction plates over resistors of equal resistance (see figure 2.18). The other plate is grounded. The actual tension V_{Cor} between the plates as a function of the (positive) channel settings V_+ and V_- is then governed by the relation:

$$V_{Cor} = (V_+ - V_-)/2$$

A positive value will bend a positive ion beam up or to the right. V_- is kept at a chosen value whereas V_+ is varied, in my application.

The figures in Appendix A show both the beam element addresses, and the cable connections to each high voltage power supply.

2.9 The Electronics Controlling the Analysing Magnet

The CAMAC D.A.C. (section 2.5) gave an output in the range 0 to +10V. Now, the interface that sets the tension of the magnetic field in the analysing magnet needs 0 to +2V to cover all of the mass range. This means that the D.A.C. output voltage has to be reduced by a factor of 5. A box containing electronics to fulfil this task was constructed by my supervisor H. Haas. It has a trimming potentiometer, so that the conversion can be adjusted to let mass 250 be reached by the +10V of the D.A.C. output. The box also contains its own power supply, and it has a switch to select between local mode (a manual knob on the UR-9 control panel, until then the only existing one) and remote mode (this application).

2.10 The Electrometer

2.10.1 Connections and Front Panel

The Keithley 485 Electrometer is a digital amperemeter that measures the current of the separated beam. From the sensor after the analysing magnet a coaxial cable conducts the signal up to the input of the 485 in UR-9. About three measurements per second are excecuted. A digital display shows the mesured value (see figure 2.20).

The instrument can be set up in remote mode. A 4853 interface card has been installed inside the cover to connect it for IEEE communication. At the rear side of the meter there is a standard IEEE connector and also 6 switches, arranged as in figure 2.21, to select the primary address, beween 0 and 30 (see table 2.12) and the remote working mode. In my application I used the address 22 and the ordinary "ADDRESSABLE MODE", not the "TALKER ONLY MODE", since I wanted the instrument to be able to listen for commands too.

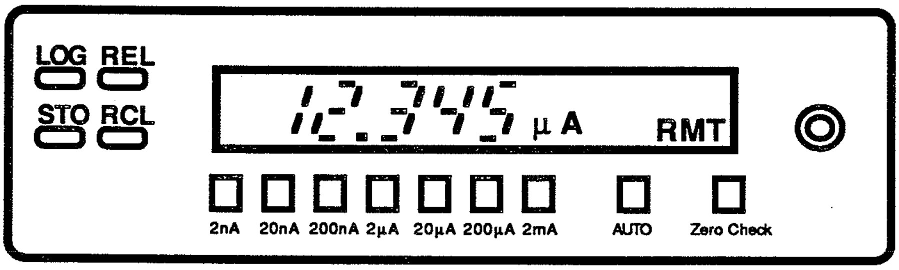


Fig 2.20 The front panel of the model 485 Electrometer.

Table 2.12

Primary Address Switch Positions

Primary Address	5	4	3	2	1
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	0	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	0	1	0	0	1
10	0	1	0	1	0
11	0	1	0	1	1
12	0	1	1	0	0
13	0	1	1	0	1
14	0	1	1	1	0
15	0	1	1	1	1
16	1	0	0	0	0
17	1	0	0	0	1
18	1	0	0	1	0
19	1	0	0	1	1
20	1	0	1	0	0
21	1	0	1	0	1
22	1	0	1	1	0
23	1	0	1	1	1
24	1	1	0	0	0
25	1	1	0	0	1
26	1	1	0	1	0
27	1	1	0	1	1
28	1	1	1	0	0
29	1	1	1	0	1
30	1	1	1	1	0

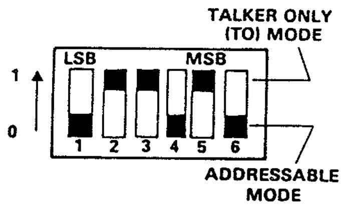


Fig. 2.21 TO/ADDRESSABLE and Primary Address Switches (Factory Set Address 22 Shown)

There are eight ranges used by the electrometer: 2nA, 20nA, 200nA, 2µA, 20µA, 200µA, 2mA, and AUTO. When "AUTO" is choosen the instrument adapts the proper range, between all the other ranges, depending on the measured value. One of the multi-choice buttons on the front panel is pressed to select a range (figure 2.20). The selection can also be done remotely as explained later.

Measured values are displayed logarithmically when the "LOG" button is pressed, or this option is choosen remotely. The "STO" and "RCL" buttons stores and recalls instrument readings. The "REL" button turns the meter into a mode where a stored baseline value (e.g. 100nA) is subtracted from the actual measured value (e.g. 1.2mA is shown as 1.1mA).

Once the "Zero Check" button is pressed the display can be zeroed to make values reliably measured with a zero offset.

2.10.2 Programming the Electrometer

2.10.2.1 Introduction - Table 2.13 shows all the possible device-dependent commands available to program the 485 electrometer. The commands are sent as bytes using the ASCII code, e.g. the string containing the three characters "COX" executes the command "Zero Check off".

Table 2.13

Device-Dependent Command Summary

Mode	Command	Description
Zero Check	C0 C1	Zero check off Zero check on
LOG	D0 D1	LOG off LOG on
Range	R0 R1 R2 R3 R4 R5 R6 R7	Auto 2nA 20nA 200nA 2 µA 20 µA 200 µA 2mA
Relative	Z0 Z1	REL off REL on
Trigger	T0 T1 T2 T3 T4 T5	Continuous on talk. One-shot on talk. Continuous on GET. One-shot on GET. Continuous on X. One-shot on X.
EOI	K0 K1	EOI enabled. EOI disabled.
Status Word	U0	Output status word.
SRQ Mode	Mnn	SRQ on error and/or data conditions
Data Format	G0 G1	Readings with prefix. Readings without prefix.
Digital Calibration	V±n.nnnnE±nn	n represents calibration value.
Store	L0	Store Calibration Constants.
Terminator	Y(ASCII) Y(LF) Y(CR) Y(DEL)	ASCII character. CR LF LF CR None
Execute	X	Execute other device-dependent commands.

2.10.2.2 Addressing - Before any device-dependent commands can be sent the ATN line has to be high, the REN signal line low, and the electrometer has to be addressed to listen (see sections 2.2.6.14, 2.2.4.2, and 2.2.4.4 respectively). The front panel display then indicates "RMT" when the electrometer is in remote mode (figure 2.20).

2.10.2.3 Execute (X) - All the command strings usually end with an 'X' to start the execution.[1] The execute character also controls instrument operation in the T4 and T5 trigger modes.

2.10.2.4 Zero Check (C) - The string "COX" turns zero check on, independent of the "Zero Check" button, if the instrument is in remote mode. Likewise "C1X" turns zero check off.

[1]The characters are in fact loaded into a command buffer so a string like "CO" can be loaded without execution until the string "X" is sent.

2.10.2.5 LOG (D) - The LOG function is set on with "D1X" and off with "D0X".

2.10.2.6 Range (R) - The range is also programmable over the IEEE bus. The alternatives are found in table 2.13.

2.10.2.7 Relative (Z) - The previously described relative mode is enabled with "Z1X" and disabled with "Z0X".

2.10.2.8 Triggering (T) - Triggering is provided to begin a reading conversion with the instrument. Triggering may be done in two basic ways: in a continuous mode, a single trigger command is used to start a continuous series of readings; in a one-shot mode, a separate trigger command is required for each reading. Table 2.13 gives the six possible commands, e.g. "T1X" triggers the 485 to send one measurement when it is addressed to talk.[1]

2.10.2.9 EOI (K) - "K0X" enables the EOI line to be set low when the 485 transfers the last data byte of any reading sequence. "K1X" disables EOI.

2.10.2.10 SRQ Mode (M) and Status Byte - The model 485 can be programmed to generate an SRQ under one or more of the following conditions:

* If a reading has been completed.

* If an owerflow condition occurs.

* If a busy condition occurs.

* If an Illegal Device-Dependent Command Option (IDDCO) is received.

* If an Illegal Device-Dependent Command (IDDC) is received.

* If the instrument is not in remote mode when a command is sent.

Table 2.14 lists how the letter 'M' followed by a decimal number makes up the command for which conditions should generate an SRQ. The number is converted to binary bits B0-B5, as the table 2.14 shows.

The status byte, obtained by the SPE command (explained in section 2.2.6.10), is configured like in figure 2.22 and table 2.15. An error sets B5. Bits B0-B2 are then set according to the cause of the error (right column in table 2.15). If "Overflow", "Reading done" or

---

[1]Of cource the electrometer always has to be addressed to talk in the usual standard way when it is about to send something on the IEEE bus. This is done after the triggering, except for the T0 and T1 modes, where the triggering and the talk addressing is one and the same ting.

Table 2.14

SRQ Mask Commands

Command	Status Bits Set	Conditions to Generate SRQ
M1	B0	Reading Overflow
M8	B3	Reading Done
M9	B3, B0	Reading Done or Reading Overflow
M16	B4	Busy
M17	B4, B0	Busy or Reading Overflow
M24	B4, B3	Busy or Reading Done
M25	B4, B3, B0	Busy, Reading Done or Reading Overflow
M33	B5, B0	IDDCO
M34	B5, B1	IDDC
M35	B5, B1, B0	IDDC or IDDCO
M36	B5, B2	Not in Remote
M37	B5, B2, B0	Not in Remote or IDDCO
M38	B5, B2, B1	Not in Remote or IDDC
M39	B5, B2, B1, B0	Not in Remote, IDDC or IDDCO

Table 2.15

Status Byte and Mask Interpretation

Bit	Bit 5=0 (Data Conditions)	Bit 5=1 (Error Conditions)
0 (LSB)	Overflow	IDDCO
1	N/A	IDDC
2	N/A	No remote
3	Reading done	N/A
4	Busy	N/A
5	Data	Error
6	SRQ	SRQ
7	N/A	N/A

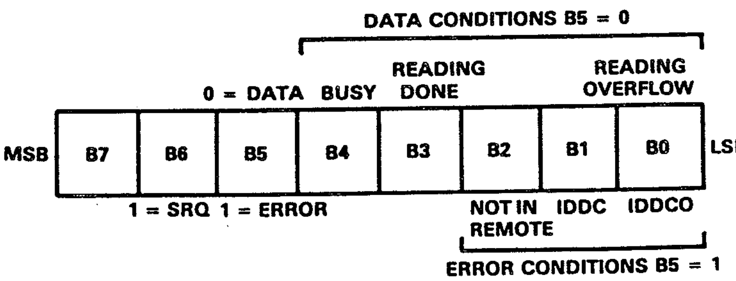


Fig. 2.22 **Status Byte Format**

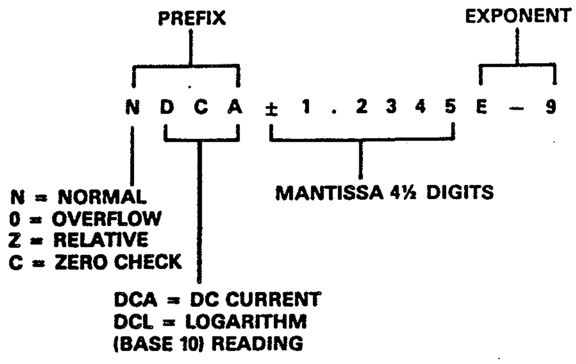


Fig. 2.23 **Data Format**

"Busy" occur their respective bits B0, B3 or B4 will be set (left column in table 2.15) but not the error bit B5. The SRQ bit B6 is set if any "MnnX" command string was specified, where the conditions to generate SRQ were any of those currently occuring.

2.10.2.11 Data Format - To read the electrometer, it first has to be trigged (see section 2.10.2.8), then addressed to talk. The format of the returned string of ASCII characters is shown in figure 2.23.

The reading is decimal with 4 1/2 digits mantissa and exponent. The first 4 characters constitute the prefix, and their meaning is explained in figure 2.23.

2.10.2.12 Prefix - The read data string either includes the prefix if the command "G0X" is sent, or excludes it if "G1X" is sent.

2.10.2.13 Programmable Terminator (Y) - In the end of a data string none, one or two characters, so called terminators are sent. The following terminators are programmable using the 'Y' letter followed by one character:

* CR LF is sent if Y(LF) (default) is specified.

* LF CR is sent if Y(CR) is specified.

* Any ASCII character, except capital letters, numbers, space, + - / , . or e is sent if Y(ASCII) is specified.

* "No terminator" is sent if Y(DEL) is specified.

2.10.2.14 Digital Calibration (V) and Storage (L) - The calibration command is sent as the string "V±n.nnnnE-nnX" where '±n.nnnn' represents the mantissa (41/2 digits), and 'E-nn' the exponent. A known calibration current is connected to the input, before this command is given. Then, when "L0X" is sent, the value is stored into a special RAM memory in the instrument.

2.10.2.11 Status Word (U) - "U0X" sends status on operating modes such as range, LOG, e.t.c. according to figure 2.24.[1]

2.10.2.12 Default Settings - The default settings when the instrument is powered up, or SDC or DCL commands are sent on the IEEE bus, are listed in table 2.16.

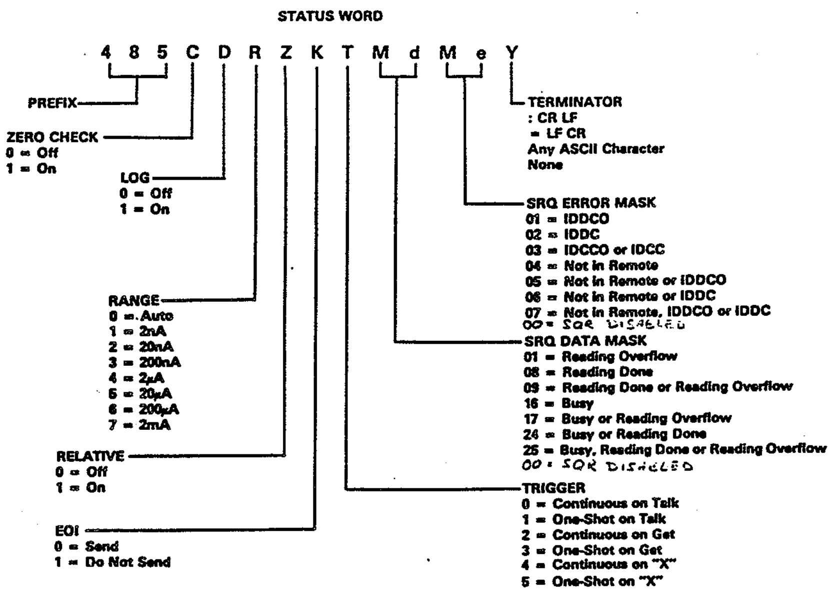


Fig. 2.24 General format U0 command.

Table 2.16

Default Conditions (Status Upon Power Up or After SDC or DCL)

Mode	Value	Status
Zero Check	—	Reflects front panel selection.
Range	—	Reflects front panel selection.
Relative	Z0	Off
EOI	K0	Send EOI.
Trigger	T0	Continuous on Talk.
SRQ	M00	Disabled
Data Format	G0	Send prefix.
LOG	D0	Off
Terminator	Y(CR LF)	CR LF
Data Store	—	Unchanged

[1]This is not to confuse with the status byte, which is sent after SPE polling.

3 THE SOFTWARE

3.1 Initiation

3.1.1 MS-DOS

The Olivetti PC uses MS-DOS (Micro Soft Disk Operating System) as its operating system.[1] Some of the commands in this system are actually files that are located in the system's directories.[2] The hierarchy of the directories on the hard disk of my PC looks like figure 3.1. The special DOS command files are found in the DOS directory which is a subdirectory under the ROOT directory. The most important file there is the "GWBASIC.EXE" file, since it contains the code for the computer language BASIC (Beginners All-purpose Symbolic Instruction Code) that I use for all my programs.

Table 3.1 shows most of the current files on the Olivetti hard disk. The most important ones are marked with dots. They will be explained later.

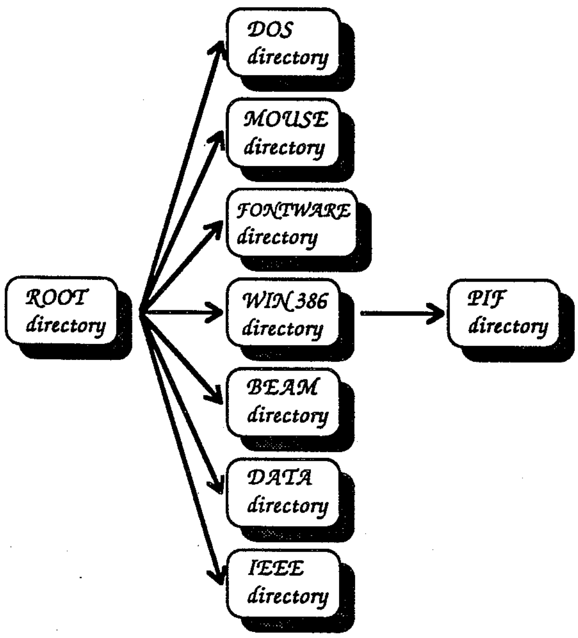


Fig 3.1 The hierarchy of the Olivetti directories.

[1]For a thorough explanation of the MS-DOS commands you are referred to the MS-DOS "User's Guide". See the literature list.

[2]A "file" is a memory area, associated with a name, where the computer code is stored. MS-DOS filenames are of the form "xxxxxxxx.eee" where x = 1 to 8 ASCII characters; e = 0 to 3 extension characters (EXE and COM are MS-DOS executable, BAS and BAC are GWBASIC executable). A "directory" is a "box" where files and other directories (subdirectories) are grouped together.

Table 3.1

Files on the Olivetti Hard Disk

Root directory on hard disk

• BEAM	<DIR>		
• DATA	<DIR>		
• DOS	<DIR>		
FONTWARE	<DIR>		
• IEEE	<DIR>		
• MOUSE	<DIR>		
• WIN386	<DIR>		
ADJUST.BAS	13347	8/22/89	3:47am
• AUTOEXEC.BAT	106	8/21/89	10:12pm
• BEAMDIR.TXT	223	8/21/89	6:24pm
• BIB.M	2223	10/01/84	12:00pm
COMMAND.COM	25292	2/12/88	9:00am
CONFIG.OLD	66	6/27/89	10:04am
• CONFIG.SYS	48	8/21/89	10:15pm
• DATADIR.TXT	116	8/22/89	3:47am
FASTRAMP.BAS	3032	8/06/89	12:37am
GOMOUSE.BAT	50	6/27/89	10:03am
• GPIB.COM	7815	8/03/89	4:52pm
• HVSET.BAS	13347	8/22/89	3:46am
• IBCONF.EXE	40810	10/01/84	12:00pm
• IBIC.EXE	30058	10/01/84	12:00pm
• NONE	19	8/21/89	2:29am
• PLASMA1.001	40961	8/19/89	12:01am
PT.LAS	33	7/25/89	1:47pm
Q.BAT	9	7/14/89	1:17pm
RUN_FW.BAT	65	7/31/89	1:43pm
• SCANNER.BAS	3230	8/21/89	12:02am
• START.BAS	958	8/21/89	1:39am
W.BAT	16	7/28/89	2:38pm
• XE.380	40961	8/21/89	12:01am
• XE.400	40961	8/21/89	12:19am
• XE380GR1.XLC	76555	8/22/89	2:16am
• XE380GR2.XLC	76555	8/22/89	2:00am
• XE400GR1.XLC	76555	8/22/89	3:02am
• XE400GR2.XLC	76555	8/22/89	3:25am
• XECAL.400	32266	8/22/89	3:34am
~SPL1F47.TMP	2741	8/22/89	10:29pm
~SPL3442.TMP	50	8/22/89	10:29pm

\Beam directory

• BL-1.B18	157	8/19/89	2:07am
• BL-2A.B2A	140	8/21/89	5:42pm
• BL-2M.B2M	121	8/21/89	5:39pm
• BL-3A.B3A	197	8/21/89	6:38pm
• BL-4.B48	140	8/19/89	2:06am
• UR10.B20	440	8/21/89	5:58pm
• URB.B28	251	8/21/89	5:49pm
• URB.B38	367	8/21/89	6:36pm
• UR9.B29	387	8/21/89	5:16pm

\Data directory

• BL-1.B18	316	8/19/89	5:47am
• BL-2A.B2A	296	8/21/89	6:02pm
• BL-2M.B2M	252	8/21/89	6:03pm
• BL-4.B48	284	8/19/89	6:04am
• CENTRAL.B2M	260	8/21/89	6:05pm
• UR9.B29	369	8/21/89	5:22pm

\IEEE directory

BBSAMP.BAS	4265	10/01/84	12:00pm
DBSAMP.BAS	2738	10/01/84	12:00pm
• DECL.BAS	4694	7/04/89	12:01pm
• DECLSHOR.BAS	606	8/19/89	12:10am
KEITH.BAS	3891	7/20/89	4:40pm
KLWORK.BAS	3361	7/18/89	5:44pm
T7.BAS	1191	8/19/89	12:17am
TESLA.BAS	2107	8/19/89	12:13am
• VOLTBIT.BAS	1501	8/19/89	12:14am
VOLTCHL.BAS	1933	8/19/89	12:15am
VOLTRAMP.BAS	1756	8/19/89	12:08am

\DOS directory

4201.CPI	17089	2/12/88	9:00am
5202.CPI	459	2/12/88	9:00am
ANSI.SYS	1647	2/12/88	9:00am
APPEND.EXE	5794	2/12/88	9:00am
ASSIGN.COM	1530	2/12/88	9:00am
ATTRIB.EXE	10656	2/12/88	9:00am
BACKUP.COM	29976	2/12/88	9:01am
BUS	0	7/28/89	6:10pm
BUSDATA	0	7/29/89	6:11pm
CACHER.SYS	8470	1/26/88	11:14am
CHKDSK.COM	9819	2/12/88	9:00am
COMMAND.COM	25292	2/12/88	9:00am
COMP.COM	4183	2/12/88	9:00am
COUNTRY.SYS	11254	2/12/88	9:00am
DEBUG.COM	15866	2/12/88	9:00am
DISKCOMP.COM	5848	2/12/88	9:00am
DISKCOPY.COM	6264	2/12/88	9:00am
DISPLAY.SYS	12207	2/23/88	9:00am
DRIVER.SYS	1165	2/12/88	9:00am
• EDIT.EXE	33095	3/07/88	12:00pm
EDLIN.COM	7495	2/12/88	9:00am
EGA.CPI	49065	2/12/88	9:02am
EXE2BIN.EXE	3050	2/12/88	9:00am
FASTOPEN.EXE	3888	2/12/88	9:00am
FC.EXE	15974	2/12/88	9:00am
FDISK.COM	51441	3/18/88	9:00am
FIND.EXE	6403	2/12/88	9:00am
FORMAT.COM	11681	2/12/88	9:00am
GRAFTABL.COM	21880	3/09/88	9:00am
GRAPHICS.COM	2752	2/12/88	9:00am
• GWBASIC.EXE	84208	2/26/88	2:20pm
HEXDUMP.COM	558	2/12/88	9:00am
JOIN.EXE	9612	2/12/88	9:00am
KEYB.COM	10031	3/09/88	9:00am
KEYBOARD.SYS	22151	2/12/88	9:00am
LABEL.COM	2346	2/12/88	9:00am
LCD.CPI	10752	2/12/88	9:00am
MODE.COM	15440	2/12/88	9:00am
MORE.COM	282	2/12/88	9:00am
NLSFUNC.EXE	3029	2/12/88	9:00am
PRINT.COM	8995	2/12/88	9:00am
PRINTER.SYS	13559	2/12/88	9:00am
PT.LAS	4	8/21/89	9:56pm
RECOVER.COM	4268	2/12/88	9:00am
REPLACE.EXE	13234	2/12/88	9:00am
RESTORE.COM	35664	2/12/88	9:01am
SELECT.COM	4132	2/12/88	9:00am
SHARE.EXE	8608	2/12/88	9:00am
SORT.EXE	1946	2/12/88	9:00am
SUBST.EXE	10552	2/12/88	9:00am
SYS.COM	5061	2/12/88	9:00am
TREE.COM	3540	2/12/88	9:00am
UNNAMED.A	0	8/06/89	12:38am
VDISK.SYS	3424	2/12/88	9:00am
XCOPY.EXE	11216	2/12/88	9:00am

\MOUSE directory

CLICK.EXE	4986	9/10/89	2:10pm
CLICK.SRC	2466	9/10/89	[illegible]
DOS.DEF	4221	7/24/87	2:26pm
DOS.MNU	3194	6/27/89	10:02am
EDLIN.DEF	1105	8/04/87	9:55am
EDLIN.MNU	366	6/27/89	10:02am
EXASSIGN.DEF	961	8/07/87	5:52pm
EXBEGIN.DEF	668	8/07/87	5:52pm
EXSEGIN.MNU	59	8/10/89	6:21pm
EXCHORDS.DEF	590	8/07/87	5:57pm
EXEXECUT.DEF	565	8/07/87	5:54pm
EXMATCH.DEF	1171	8/07/87	5:55pm
EXMATCH.MNU	563	8/10/89	6:12pm
EXMENU.DEF	661	8/07/87	5:56pm
EXMENU.MNU	291	8/10/89	5:58pm
EXPOPUP.DEF	844	8/07/87	5:58pm
EXPOPUP.MNU	185	8/10/89	6:21pm
EXTYPE.DEF	629	8/07/87	5:50pm
EXTYPE.MNU	52	8/10/89	6:21pm
FW.DEF	1834	8/04/87	9:55am
FW.MNU	154	6/27/89	10:02am
• GWBASIC.DEF	761	8/19/89	6:44am
• GWBASIC.MNU	59	8/19/89	6:45am
KEYBASE.DEF	1897	8/06/89	12:12am
KEYBASE.MNU	75	8/10/89	9:09pm
KNOBB.BAS	2612	8/06/89	8:30am
KNOBBB.BAS	2204	8/06/89	6:15am
M123.COM	8563	4/06/88	8:59am
M123.MEN	1161	4/21/88	3:36pm
• MENU.COM	7464	9/04/87	9:24am
MOD.DEF	1979	8/04/87	9:56am
MOD.MNU	864	6/27/89	10:02am
MOUSE.COM	14464	3/24/88	12:46pm
• MOUSE.SYS	14620	3/24/88	12:47pm
MUS.BAS	74	8/06/89	1:59am
NEWCLICK.EXE	17548	7/13/87	5:47pm
• NEWMENU.EXE	17362	7/17/87	10:44am
PROLOG.DEF	2354	8/03/87	5:06pm
PROLOG.MNU	859	6/27/89	10:02am
PT.EXE	109129	1/08/88	12:36pm
PT.HLP	67507	7/15/87	7:39am
PT.INI	1527	1/13/88	4:33pm
PT.LAS	69	6/30/89	6:54pm
PT.MSG	8000	11/05/86	9:53pm
PTCOLOR.INI	1527	1/13/88	4:33pm
PTEXPERT.INI	1729	11/25/87	6:08pm
PTMONO.INI	1456	1/13/88	4:33pm
PTMOTION.INI	1261	3/09/87	1:56pm
QMODEM.DEF	2585	8/04/87	9:56am
QMODEM.MNU	1123	6/27/89	10:02am
SC3.DEF	2074	8/04/87	9:56am
SC3.MNU	826	6/27/89	10:02am
SC4.DEF	1712	8/19/87	10:54am
SC4.MNU	512	6/27/89	10:02am
SYMPHONY.DEF	2335	8/03/87	5:03pm
SYMPHONY.MNU	1312	6/27/89	10:02am
TEST.TST	0	6/30/89	6:53pm
TURBO.DEF	2555	8/03/87	5:07pm
TURBO.MNU	1436	6/27/89	10:02am
VP.DEF	2056	8/03/87	5:08pm
VP.MNU	893	6/27/89	10:02am
WP.DEF	9537	9/08/87	4:22pm
WP.MNU	3423	6/27/89	10:02am

\WIN386 directory

PIF	<DIR>		
ABC.TXT	12	8/11/87	10:55am
CALC.EXE	28000	11/12/87	1:21pm
CALENDAR.EXE	38895	11/12/87	1:22pm
CARDFILE.EXE	39264	11/12/87	1:27pm
CGA40.FON	6544	11/15/87	9:01pm
CGA80.FON	4512	11/15/87	9:01pm
CLIPBRD.EXE	10800	11/12/87	1:26pm
CLOCK.EXE	8960	11/12/87	1:27pm
COMMANDE.PIF	369	9/10/87	11:12am
COMMANDF.PIF	369	9/10/87	11:12am
COMMANDW.PIF	369	9/10/87	11:13am
CONTROL.EXE	57216	11/12/87	1:23pm
COURB.FON	19088	12/15/87	5:53pm
COURC.FON	13040	12/15/87	5:53pm
COURD.FON	21328	12/15/87	5:54pm
COURE.FON	23808	12/15/87	5:54pm
CVTPAINT.EXE	5712	11/12/87	1:10pm
DOTHIS.TXT	493	8/14/87	10:56am
EGA40.FON	8576	11/15/87	9:01pm
EGA80.FON	5520	11/15/87	9:01pm
• EXCEL.EXE	597824	10/31/87	12:00pm
• EXCEL.INF	2796	10/31/87	12:00pm
• EXCELHLP.HLP	357552	10/31/87	12:00pm
HELVB.FON	50816	12/18/87	2:40pm
HELVC.FON	39008	12/18/87	2:41pm
HELVD.FON	58256	12/18/87	2:42pm
HELVE.FON	64688	12/18/87	2:44pm
IBMGRX.DRV	14512	11/12/87	1:28pm
MODERN.FON	7584	11/12/87	1:26pm
MSDOS.EXE	1	7/28/89	4:14pm
NOTEPAD.EXE	19072	11/12/87	1:28pm
PAINT.EXE	93280	11/12/87	1:23pm
PIFEDIT.EXE	26464	9/10/87	5:39pm
PRACTICE.WRI	2944	8/14/87	10:56am
READATOL.TXT	4022	3/15/88	12:14pm
README.TXT	21841	10/31/87	12:00pm
READMEEM.TXT	20811	10/31/87	12:00pm
REVERSI.EXE	15552	11/12/87	1:28pm
ROMAN.FON	11120	11/12/87	1:27pm
SCRIPT.FON	10304	11/12/87	1:27pm
SETUP.EXE	40136	10/31/87	12:00pm
SETUP2.EXE	4566	10/31/87	12:00pm
SPOOLER.EXE	14640	11/12/87	1:27pm
TERMINAL.EXE	48640	11/12/87	1:28pm
TMSRB.FON	45872	12/18/87	2:35pm
TMSRC.FON	37872	12/18/87	2:36pm
TMSRD.FON	57360	12/18/87	2:37pm
TMSRE.FON	58224	12/18/87	2:39pm
TRANS.EXE	106384	10/31/87	12:00pm
TRANS123.XLM	18500	10/31/87	12:00pm
WIN.INI	4024	8/22/89	2:31am
WIN.OLD	3699	7/28/89	4:04pm
WIN200.BIN	229008	7/28/89	4:11pm
WIN200.OVL	273760	7/28/89	4:11pm
WIN386.386	139568	12/07/87	10:57am
WIN386.EXE	34460	12/07/87	10:48am
WIN86.COM	6877	7/28/89	4:11pm
WINOLDAP.GRB	2589	7/28/89	4:14pm
WINOLDAP.MOD	31120	12/07/87	11:01am
WRITE.EXE	198336	11/12/87	1:20pm

DOS is interactive and is in succesion asking for commands by showing a promt like:

C>

You just type the command after the prompt and press "ENTER". "C" means the hard disk, whereas "A" and "B" mean the disk drives. The command to change to another drive is:

A: B: or C:

The command to change directory is:

cd \	(to ROOT directory)	or
cd \DOS	(to DOS directory)	or
cd \WIN386\PIF	(to PIF directory)	
e.t.c.		

DOS command lines may be grouped together and stored in a so called batchfile. The "AUTOEXEC.BAT" file in the ROOT directory is an example of such a file.

Figure 3.2 shows the contents of "AUTOEXEC.BAT", and also that of the file "CONFIG.SYS"[1], because these two are startup files. They are automatically executed after power up or system reset. Using the texteditor "EDIT.EXE" (in the DOS directory) I changed them to be like figure 3.2 shows. Dots mark the most important commands.

One of the commands that I changed was "PATH". This command tells how the computer should search for files in the directories. If the path is not given for a file it can only be reached from its own directory, or the path must be put as a prefix before the filename.

CONFIG.SYS

```
•DEVICE = C:\MOUSE\MOUSE.SYS
•DEVICE = GPIB.COM
```

AUTOEXEC.BAT

```
•PATH C:\:C:\DOS:\IEEE;\MOUSE:\WIN386:
 KEYB US 437 C:\DOS\KEYBOARD.SYS
 ECHO OFF
•MENU GWBASIC
 CLS
 VER
```

Fig. 3.2 The start up files.

[1]"CONFIG.SYS" is not a batchfile, only a start up file where drivers are loaded. It is executed before "AUTOEXEC.BAT".

```
:-------------------------------------------------------------------------:
:                                                                         :
:                              GWBASIC                                    :
:                                                                         :
:-------------------------------------------------------------------------:

BEGIN   leftb, midb, rightb, leftm, rightm, upm, downm, 4,8

:
: The three buttons
:
leftb:   TYPE 3
midb:    TYPE ENTER
rightb:  TYPE ESC

:
: The movements
:
leftm:   TYPE 0,75        : Left arrow (extended 75)

rightm:  TYPE 0,77        : Right arrow (extended 77)

upm:     TYPE 0,72        : Up arrow (extended 72)

downm:   TYPE 0,80        : Down arrow (extended 80)
```

Fig. 3.3 The mouse definition file "GWBASIC.DEF".

3.1.2 The Logitech Mouse

The statement "DEVICE = C: \MOUSE\MOUSE.SYS" in "CONFIG.SYS" loads the mouse driver software for the three button Logitech mouse by calling the file "MOUSE.SYS" in the MOUSE directory. Then when the command "MENU GWBASIC" in the "AUTOEXEC.BAT" file is accounted for, the MOUSE directory file "MENU.COM" will be called, which in turn calls the file "GWBASIC.MNU". This sets up the actual usage of the mouse buttons and mouse movements.

The manual "Getting The Most From Your Mouse" (see the literature list) explains how to program the mouse. I used this to make the MOUSE directory file "GWBASIC.DEF". The print-out is shown in figure 3.3. The statement "leftb TYPE 3" means that when the left mouse button is pressed an "ASCII 3" (keyboard control-C) is sent to the terminal. In the same way the mid button sends "ENTER", and the right "ESC". Mouse movements will send keyboard arrows, and much faster than possible by pressing them (and keep pressed) on the keyboard.

The "NEWMENU.EXE" file was used to compile the source file "GWBASIC.DEF" into "GWBASIC.MNU". The command syntax is:

NEWMENU GWBASIC

3.1.3 The IBM Software

The statement "DEVICE = GPIB.COM" in "CONFIG.SYS" loads the IBM IEEE bus driver software, compatible with both GWBASIC and the Elite card hardware, by calling the file "GPIB.COM" in the ROOT directory.

The IEEE driver software was supported by an IBM master diskette. The relevant files on this diskette are shown in table 3.2. With the diskette in drive "A" the following command installed the files:

GBIBSET C: 216 BASICA

It was then necessary to use the "IBCONF.EXE" file to set special IEEE communication parameters, such as primary address, timeout (the time to wait before the communication is aborted), and data on termination with EOS (End Of String) and EOI (End Or Identify). The current settings are listed in table 3.3, which shows IBCONF screen print-outs. "GPIB.COM" serves as a storing place for the IBCONF parameters, read in upon start up, as said in the beginning of this section.

The controller, i.e. the Olivetti PC, is called "GPIB0" adapter.[1] It has primary address 02 Hex (remember that the Elite card address is 02E1 Hex, where the first two hexadecimal digits are 02 Hex). The so called DMA is a control unit inside the computer. The factory set Elite card address for the DMA is 1.

"DEV0" is the device name for the CAMAC 8901 interface. "DEV1" is the device name for the electrometer. Their respective addresses are 5 and 22 (16 Hex), as mentioned in sections 2.4 and 2.10.

Table 3.2

GPIBSET.BAT	Installation program
GPIB216.COM	Device Driver (supports up to 2 GPIB Adapters and up to 16 devices)
GPIB448.COM	Device Driver (supports up to 4 GPIB Adapters and up to 48 devices)
CONFIG.SYS	Sample configuration file
IBCONF.EXE	Interactive configuration program
IBIC.EXE	Interface Bus Interactive Control Program
BIB.M	Interface to BASICA
DECL.BAS	Declarations for BASICA

[1]Actually the IBM software configures the Olivetti to have two adapters with up to 16 bus devices , or four adapters with up to 48 bus devices, if 448 was used in stead of 216 in the GBIBSET command. However in my system there was only need for one adapter.

Table 3.3

IBCONF.EXE

```
Adapter Name: GPIB0            ADAPTER PARAMETERS                     Number: A

                DESCRIPTION        NEW VALUE     VALID RANGE

          Primary GPIB Address?      02H           [0H to 1EH]
        Secondary GPIB Address?      00H           [60H to 7EH: 0H disables]
               Timeout setting?      T1s         ± [T10us to T1000s:TNONE disables]

                     EOS Byte?      0AH           [0H to FFH or '<character>]
        Terminate Read on EOS?      No          ± [Yes or No]
        Send EOI with EOS byte?     No          ± [Yes or No]
      Use 8-bit Compare on EOS?     No          ± [Yes or No]

  Is Adapter System Controller?     Yes         ± [Yes or No]
  Send EOI w/last Byte of Write?    No          ± [Yes or No]
                 Adapter Number?    0             [0 to 7]
        Adapter Interrupt Level?    6             [2 to 7]
                   DMA Channel?     1           ± [1 to 3: NONE disables]
              Tri-State Timing?     Yes         ± [Yes or No]
   Local lockout on all devices?    No          ± [Yes or No]

          F1 = Help     F2 = Previous Values     F6 = Exit
```

```
Device Name: DEV0...            DEVICE PARAMETERS                    Number: D 0

                DESCRIPTION        NEW VALUE     VALID RANGE

           Access Adapter Name?     GPIB0       ± [GPIBx]
          Primary GPIB Address?     05H           [0H to 1EH]
        Secondary GPIB Address?     00H           [60H to 7EH; 0H disables]
               Timeout setting?     T3s         ± [T10us to T1000s:TNONE disables

                     EOS Byte?     00H           [0H to FFH or '<character>]
         Terminate Read on EOS?     No          ± [Yes or No]
        Send EOI with EOS byte?     No          ± [Yes or No]
      Use 8-bit Compare on EOS?     No          ± [Yes or No]

  Send EOI w/last Byte of Write?    No          ± [Yes or No]

          F1 = Help     F2 = Previous Values     F6 = Exit
```

```
Device Name: DEV1...            DEVICE PARAMETERS                    Number: D 1

                DESCRIPTION        NEW VALUE     VALID RANGE

           Access Adapter Name?     GPIB0       ± [GPIBx]
          Primary GPIB Address?     16H           [0H to 1EH]
        Secondary GPIB Address?     00H           [60H to 7EH; 0H disables]
               Timeout setting?     T1s         ± [T10us to T1000s:TNONE disables]

                     EOS Byte?     0AH           [0H to FFH or '<character>]
         Terminate Read on EOS?     No          ± [Yes or No]
        Send EOI with EOS byte?     No          ± [Yes or No]
      Use 8-bit Compare on EOS?     No          ± [Yes or No]

  Send EOI w/last Byte of Write?    No          ± [Yes or No]

          F1 = Help     F2 = Previous Values     F6 = Exit
```

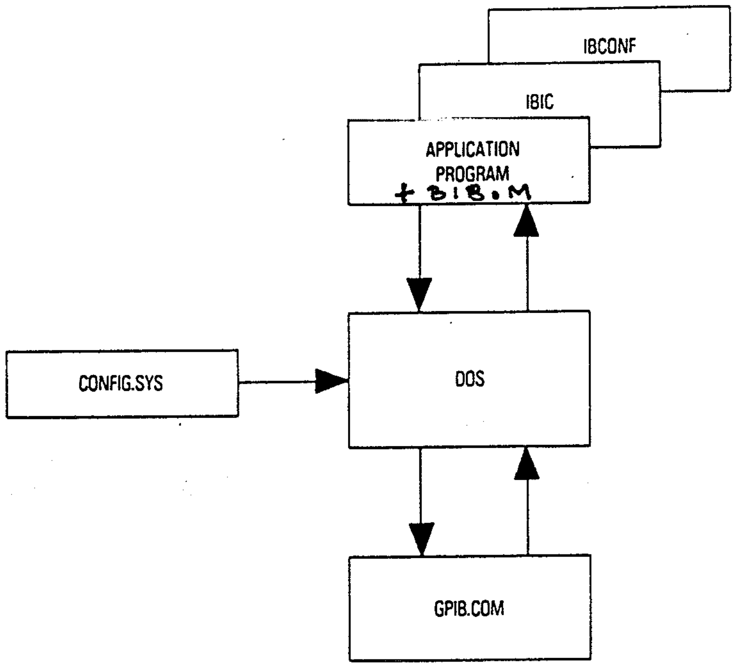


Fig. 3.4 How the application program functions with DOS and "GPIB.COM".

Figure 3.4 shows how the application programs (GWBASIC with files) are related with DOS and "GPIB.COM".

IBIC is a test program to interactively try IEEE commands. "BIB.M" is a file containing assembler code, to be loaded into BASIC programs to accomplish IEEE commands in the GWBASIC environment. These commands, or "CALL" routines, are listed and described in appendix B.

More details about the IBM IEEE software are found in the manual "IBM Personal Computer General Purpose Interface Bus Adapter Programming Support" (see the literature list).

3.2 The Program VOLTBIT

To test the IEEE communication with GWBASIC I wrote the simple program VOLTBIT (listed in appendix C).[1]

VOLTBIT sets the 12 input bits of channel 1 of the D.A.C. installed in CAMAC slot position 20. One bit at a time is set to 1, while the others are set to 0. The D.A.C. output is connected to a voltmeter.

[1] I also made other BASIC test files. The still existing ones are listed in table 3.1with file extension BAS. I put most of them in the IEEE directory.

The following steps make the program run:

* From DOS at the C> prompt type GWBASIC and press ENTER to load the BASIC.

* From BASIC type CHDIR"\IEEE" and press ENTER to let the IEEE directory be the current directory.

* Type LOAD"VOLTBIT" and press ENTER to load the "VOLTBIT.BAS" file.

* The command RUN will begin the execution.[1]

On the Olivetti screen the digital D.A.C. input value will appear both in hexadecimal and decimal notation (VALUE). The expected D.A.C. output voltage (VOLT) is also shown. It is calculated in the program using the formula:

VOLT=10*VALUE/4095

The screen now displays "READ VOLTAGE IN CH1 AND PRESS ANY KEY TO CONTINUE!". Thus the output voltage of the voltmeter may now be checked and any key pressed for a new bit of the 12 D.A.C. input bits to be tested.

The BASIC statement lines 60-150 in the beginning of VOLTBIT are also common to the other programs SCANNER and HVSET described in the following sections (except for line numeration). They are copied from the IEEE directory file "DECLSHOR.BAS", which is my modification (unnecessary lines deleted) of the supported "DECL.BAS" file. The purpose of these lines is to load the previously mentioned IBM assembler code file "BIB.M", and also to initiate some variables. I had to calculate the assembler start address 57611 by the following subtraction:

BASIC free memory	59834
"BIB.M" memory size	2223
Difference	57611

One example of an important IBM BASIC routine in VOLTBIT is:

```
WRT$ = CHR$(16) + CHR$(0) + CHR$(20) + LB$ + MB$
CALL IBWRT(DEV0%,WRT$)
```

It accommodates the CAMAC write command (see section 2.5):

F(16)•A(0)•N(20)•byte1•byte2

---

[1] A discription of the GWBASIC is given in the manual "MS GW-BASIC INTERPRETER under MS-DOS" (see the literature list).

3.3 The Program START

The BASIC file "START.BAS" is a program that displays a menu from where the other two important programs HVSET and SCANNER can be selected and started.[1] START is loaded in a similar way to VOLTBIT:

* From DOS at the C> prompt type GWBASIC and press ENTER to load the BASIC.

* From BASIC type CHDIR"\" and press ENTER to let the ROOT directory be the current directory.

* Type LOAD"VOLTBIT" and press ENTER to load the "VOLTBIT.BAS" file.

* The command RUN will begin the execution.

Figure 3.5 shows the screen layout at this stage. The mouse can now be used to move a cursor up and down in the framed menu. If the middle mouse button is "clicked" (or keyboard ENTER is pressed) the following actions will take place, depending on the marked text:

MASS SCAN	loads "SCANNER.BAS" and starts execution.
BEAM LINE CONTROL	loads "HVSET.BAS" and starts execution.
EXIT	finishes START and returns to BASIC.

```
WELCOME TO THE SYSTEM FOR MAGNET CONTROL OF ISOLDE 2



                    *********************
                    * MASS SCAN         *
                    * BEAM LINE CONTROL *
                    * EXIT              *
                    *********************
```

Fig. 3.5 The START menu.

[1] In the future even more programs may be loaded from START.

3.4 The Program HVSET

The mouse controlled program HVSET (listed in appendix C) constitutes the software for the beam adjustment system. It sets the values of the different beam elements of ISOLDE-2 by setting values of tension in the HV4032A high tension units via the IEEE bus and a CAMAC controller (see the hardware description in chapter 2).

When the program starts a menu pops up (figure 3.6) asking for which beam line and set up ("EXPERIMENT") should be used. Just move the mouse up or down (or use the cursor keys) and click the middle button (or press keyboard ENTER) to select any beam line or action. If the list of beam-line set-ups contains more than 10 cases (the number shown at once) you can move in the list by clicking:

--TOP	to go to the top of the list.
--PREVIOUS	to scroll 10 names up.
--NEXT	to scroll 10 names down.
--BOTTOM	to go to the bottom of the list.

There is also the option:

CANCEL	leaves HVSET and reloads START.

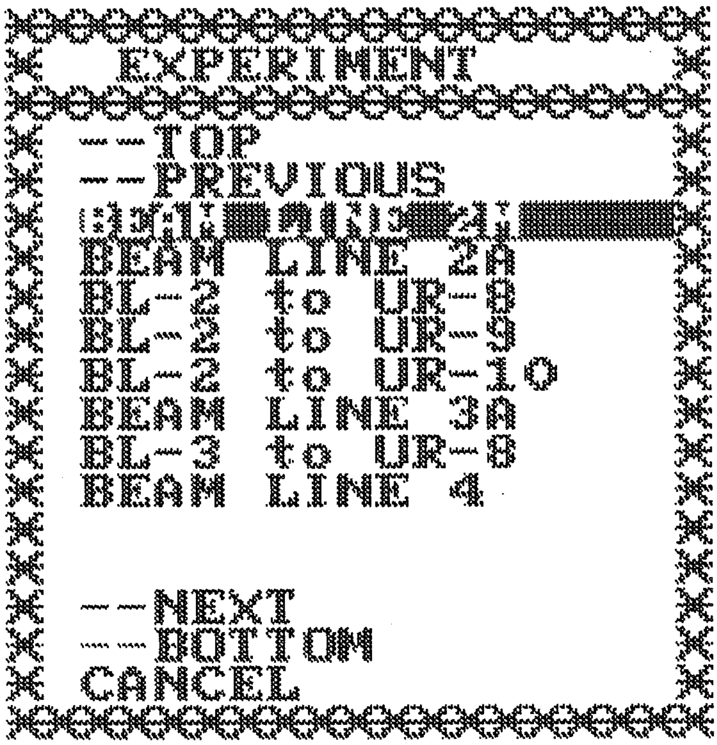


Fig. 3.6 The BEAM menu, which is also the first displayed menu in HVSET.

Assume that "BEAM LINE 2M" is choosen. Then HVSET searches for this name in the left column of the ROOT directory file "BEAMDIR.TXT" and finds it on the second line (see table 3.4). The right column shows the filenames of BEAM directory files, where the beam element addresses (such as mainframe, channel e.t.c.) are stored. In our example the file is "BL-2M.B2M". Thus the program reads this file from the BEAM directory.

On the first line of "BL-2M.B2M" the text "BL-2M" means the keyname that is also found in data storage files.[1] The integer 6 is the number of affected beam elements. Their names, (e.g. "2LB ver"), mainframe addresses (e.g. 16), positive channel addresses (e.g. 5), and negative channel addresses (e.g. 26) are extracted from the following lines of "BL-2M.B2M".

At this stage a text and graphic layout appears on the Olivetti screen. Figure 3.7 shows a screen-copy.

The first time and every time a new beam line is choosen the values of the beam elements and high tension status are read in from the power crates. High tension can be "ON", "OFF" or "BOTH" (if some used mainframes are switched on and others off). The upper right screen corner shows the power status. If high tension is on then the "true" output voltage values are read, otherwise the internally "set" voltage values are read. "VALUE" shows the reading, for every beam element.

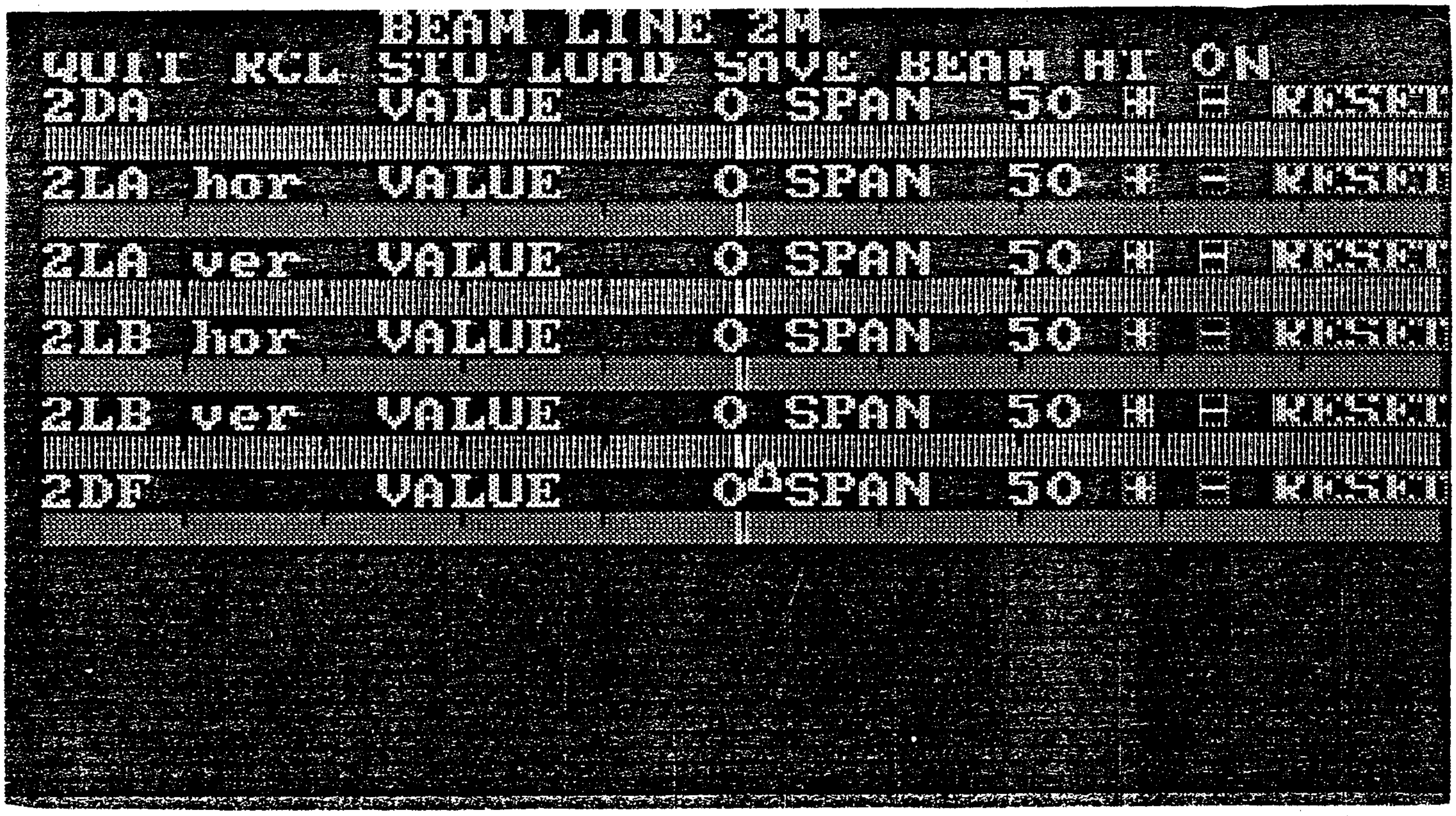


Fig. 3.7 The main HVSET layout with bars and action boxes.

---

[1]These DATA directory files are explained later in the text.

Table 3.4

Data files used by HVSET[1,2]

The file where beam line element addresses are stored

File stored with SAVE

BEAMDIR .TXT

```
BEAM LINE 1.    BL-1.B18
BEAM LINE 2M    BL-2M.B2M
BEAM LINE 2A,   BL-2A.B2A
BL-2 to UR-8,   UR8.B28
BL-2 to UR-9,   UR9.B29
BL-2 to UR-10,  UR10.B20
BEAM LINE 3A,   BL-3A.B3A
BL-3 to UR-8,   UR8.B38
BEAM LINE 4.    BL-4.B48
```

Beam line name shown by the HVSET BEAM-menu

BL-2M. B2M

```
BL-2M         6
2DA,      16,0,31
2LA hor,16,2,29
2LA ver,16,3,28
2LB hor,16,4,27
2LB ver,16,5,26
2DF,      16,1,30
```

Number of beam line elements

Beam line element name

Mainframe

Channel for negetive output

Channel for positive output

Beam line keyname

CENTRAL . B2M

```
BL-2M
FIRST TRY
TRANSMISSION 50%
ALL WORKS

2DA.        1988     100
2LA hor.    677      100
2LA ver.    653      100
2LB hor.    855      100
2LB ver.    778      100
2DF.        0        100
```

4 lines of comments

Beam line element name

Voltage values

Span values

BL-1.B18

```
BL-1.       8
1DA.        14.8,23
1LA hor.    14,0,31
1LA ver.    14.1.30
1LB hor.    14,2.29
1LB ver.    14,3.28
COR1.       14,10.21
1DB too.    14.11.20
1DB bot.    14.14.17
```

BL-2A.B2A

```
BL-2A.      7
2DA,      16,0,31
2LA hor,16,2,29
2LA ver,16,3,28
2LB hor,16,4,27
2LB ver.16,5,26
COR6 ver,16,9,22
2DB.      16,10,21
```

UR8, B28

```
BL-2 to UR-8,    13
2DA.      16,0,31
2LA hor,16,2,29
2LA ver,16,3,28
2LB hor,16,4,27
2LB ver,16,5,26
COR6 hor,16.8.23
COR6 ver,16.9.22
2LC hor,16,6.25
2LC ver.16,7,24
2LE hor,15,0.31
2LE ver.15,1,30
COR7 hor,15,2.29
COR7 ver.15,3.28
```

UR9 . B29

```
BL-2 to UR-9.    20
2DA.      16,0,31
2LA hor,16,2.29
2LA ver,16,3,28
2LB hor,16,4,27
2LB ver,16.5,26
COR6 hor.16.8.23
COR6 ver.16.9.22
2LC hor.16.6.25
2LC ver.16.7.24
2DC.      15.4.27
COR8.     15.5.26
2LF hor.15,6.25
2LF ver.15.7,24
2LG hor,15.8.23
2LG ver,15.9.22
2DD,      15.14.17
2LH hor.15.10.21
2LH ver.15.11.20
COR9 hor.15.12.19
COR9 ver.15.13.18
```

UR10.B20

```
BL-2 to UR-10.    24
2DA.      16.0.31
2LA hor.16.2.29
2LA ver.16.3.28
2LB hor.16.4.27
2LB ver.16.5.26
COR6 hor.16.8.23
COR6 ver.16.9.22
2LC hor.16.6.25
2LC ver.16.7.24
2DC.      15.4.27
COR8.     15.5.26
2LF hor.15.6.25
2LF ver.15.7.24
2LG hor.15.8.23
2LG ver.15.9.22
2LJ hor.11.0.31
2LJ ver.11.1.30
COR10 hor.11.2.29
COR10 ver.11.3.28
2LK hor.11.4.27
2LK ver.11.5.26
2DE.      11.8.23
2LL hor.11.6.25
2LL ver.11.7.24
```

BL-3A.B3A

```
BL-3A.      10
3DA.        13,0,31
3LA hor.    13,2,29
3LA ver,    13.3,28
3LB hor.    13,4,27
3LB ver.    13.5,26
KOR1 hor,13,10.21
KOR1 ver,13,11,20
3LC hor,    13,6,25
3LC ver.    13.7,24
3DB.        13,1,30
```

UR8.B38

```
BL-3 to UR-8,  19
3DA.        13.0.31
3LA hor.    13,2,29
3LA ver.    13,3,28
3LB hor,    13,4,27
3LB ver,    13,5,26
KOR1 hor,13,10,21
KOR1 ver,13,11,20
3LC hor,    13,6,25
3LC ver,    13,7,24
KOR2 hor,13,12,19
KOR2 ver,13,13,18
3LD hor,    13,8,23
3LD ver,    13,9,22
KOR3 ver,10,6,25
3DD,        10,1,30
3LE hor,    10,2,29
3LE ver.    10,3,28
KOR4 hor.10.4,27
KOR4 ver.10.5,26
```

BL-4.B48

```
BL-4.       7
4DA.        14.9.22
4LA hor.    14.4.27
4LA ver.    14,5.26
4LB hor,    14.6,25
4LB ver,    14,7,24
COR4hor,    14.12.19
COR4ver,    14,13,18
```

[1] To create a new beam line configuration just use a texteditor to make a file like "BL-2M.B2M" and put it into the BEAM directory. Then put another line into the "BEAM.TXT" file, with the name of the new beam line, and the name of the created file.

[2] Also used files: "DATADIR.TXT" containing the current LOAD-filename list (it is contionously changed), and "NONE" used to dump system messages that are unwanted on the screen.

Associated with every beam element is a bar (looking like a ruler - see figure 3.7). The middle position of a bar is where the voltage is as told by "VALUE" after any new beam line set up is loaded, or you have choosen "LOAD" or "RESET". "SPAN" tells you how many voltage steps all of the bar makes.

The following can be done with the mouse:

* Movements (the same as the cursor keys) move the triangle cursor across the screen.

* The middle button (the same as the ENTER key) clicks a bar or an action box (see below).

* The left button scrolls the bars and beam element information upwards. This is used if there are more elements than there is room for on the screen.

* The right button scrolls the bars and beam element information downwards.

If you click on a bar, the barmarker moves to the position where you clicked, and a new voltage value is calculated and stored into the corresponding HT unit for the choosen beam element. The "action boxes" and their actions, when clicked at, are listed in the following (see also figure 3.7):

* QUIT (or Q key) stops the program and reloads the START menu.

* RCL recalls a previously stored (with STO) set of tension values.

* STO temporally stores values later to be recalled by RCL.

* LOAD presents a menu, similar to the one in the beginning of the program (figure 3.6), with the names of DATA directory files containing previously saved values (with SAVE). Move the mouse and click. Then four comment lines about file contents will be shown. Before loading the values, the used HT units are all set to 0 (zero). When the loading is finished obey the text "PRESS MIDDLE BUTTON ON MOUSE OR RETURN".

* SAVE presents a menu asking for a filename (without file extension, which is automatically put after the name) and then four comment lines to describe the file. Type in these things and do not forget ENTER after each line (only ENTER makes an empty comment line). The current values in the middle of the bars (note this!) are now saved, with the specified filename, into the DATA directory, the way table table 3.4 shows.

* BEAM presents the same menu as in the beginning of the HVSET execution. Move the mouse and click for a new beam line.

* HT sets high tension ON or OFF alternatingly (when BOTH were set then all the used HT crates are turned OFF).

* + or - increases or decreases the span of the corresponding beam element. Minimum is 20 and maximum is 2000.

* RESET places the barmarker of the corresponding beam element in the middle of the bar, without affecting the VALUE. This is used before SAVE, or to move the bar span further up or down the voltage scale (place the marker at the bar's end and click RESET, then place it there again e.t.c.).

The pair STO-RCL is used to test a beam line setting. New values overwrite the old ones, and everything disappears when HVSET is left. The pair SAVE-LOAD on the other hand is used when ready adjustments are made, and there is a wish to keep them constantly in a file.

3.5 The Program SCANNER

The program SCANNER implements the mass spectrometer (or scanner) on the Olivetti. It uses 4096 digital channels (from 0 to 4095) that are sent by the IEEE bus to a Digital/Analogue Converter, which through some electronics stears the separating magnet. The response that the program gets is the beam current, which is a measure of particle intensity (see also the hardware description in chapter 2). The following steps occur in SCANNER:

* The program asks for the first and last channels to be checked and the lenght of stepping between channels (figure 3.8a). You can even step backwards if you choose the first channel to be greater than the last. The stepping lenght should always be positive though. Note that 0 is minimum and 4095 is maximum channel numbers.

* The program now asks if the units are switched on (figure 3.8b) . If not, do so. And do not forget to turn the unit between the magnet and the D.A.C. into remote mode. Then press any key.

* SCANNER displays the channel, time between succesive readings, read beam current, and the converted integer (which is in tenths of a nanoampere), for every reading until all the specified channels have been read (figure 3.8c).

* When the readings are finished the program asks for a filename to store the values of the converted integers or RETURN, i.e. ENTER

(figure 3.8d). The filename should be given with its extension.[1] The file format is one integer (maximum value 32767) on each line, right adjusted in a 8 character field. This format is accepted by graphics software such as INNOVAT and MS-EXCEL (see chapter 4).

* START is reloaded when the mass spectrum file has been stored.

The median time between succesive readings of the electrometer is 0.32 seconds. This was as fast as I managed to reach. To do this I used so called "asynchronous" readings, which gave the disadvantage that a lag of one reading occured between readings. But I have taken this into account in my program. The system clock has also been turned off, so that it cannot interfere with the BASIC interpreter and cause unwanted time fluctuations. Be ware of this. Time and date are for instance unreliable, after SCANNER has been executed.

The total time for SCANNER is about 25 minutes when 4096 readings are done. Code listing for SCANNER appears in appendix C.

```
FIRST CHANNEL? 0
LAST CHANNEL ? 100
STEP LENGTH  ? 1

1LIST   2RUN  3LOAD"  4SAVE"  5CONT 6,"LPT1 7TRON 8TROFF9KEY    0SCREEN
```

a

```
MAKE SURE THAT THE AMPEREMETER AND
THE CAMAC CONTROLLER ARE SWITCHED ON!
THEN PRESS ANY KEY!

1LIST   2RUN  3LOAD"  4SAVE"  5CONT 6,"LPT1 7TRON 8TROFF9KEY    0SCREEN
```

b.

```
START OF SCANNING!
 0          .32          3.116351E-06  32767
 1          .32          3.116351E-06  32767
 2          .32          3.116351E-06  32767
 3          .32          3.116351E-06  32767
 4          .32          3.116351E-06  32767
 5          .32          3.116351E-06  32767
 6          .32          3.116351E-06  32767
 7          .32          3.116351E-06  32767
 8          .32          3.116351E-06  32767
 9          .32          3.116351E-06  32767
 10         .32          3.116351E-06  32767
 11         .32          3.116...

1LIST   2RUN  3LOAD"  4SAVE"  5CONT 6,"LPT1 7TRON 8TROFF9KEY    0SCREEN
```

c.

```
 80         .32          3.116352E-09  31
 81         .32          3.116352E-09  31
 82         .32          3.116352E-09  31
 83         .32          3.116352E-09  31
 84         .32          3.116352E-09  31
 85         .32          3.116352E-09  31
 86         .32          3.116352E-09  31
 87         .32          3.116352E-09  31
 88         .32          3.116352E-09  31
 89         .32          3.116352E-09  31
 90         .32          3.116352E-09  31
 91         .32          3.116352E-09  31
 92         .32          3.116352E-09  31
 93         .32          3.116352E-09  31
 94         .32          3.116352E-09  31
 95         .32          3.116352E-09  31
 96         .32          3.116352E-09  31
 97         .32          3.116352E-09  31
 98         .32          3.116352E-09  31
 99         .32          3.116352E-09  31
 100        .32          3.116352E-09  31
LAST        9.999999E-02               3.116352E-09  31
END OF SCANNING!
SAVE DATA AS (RETURN=SKIP)?
1LIST   2RUN  3LOAD"  4SAVE"  5CONT 6,"LPT1 7TRON 8TROFF9KEY    0SCREEN
```

d.

Fig. 3.8 a) First SCANNER screen. b) Second SCANNER screen. c) The start of the mass scan. d) The end of the mass scan and file saving.

---

[1]"PLASMA1.001", "XE.380" and "XE.400" are such files.

4 RESULTS

4.1 Test of the Beam Adjustment System

When a new beam line set-up is to be tested, it seems natural to measure the ion current after it has passed a number of beam elements. The simple mouse control of HVSET makes it possible to adjust those elements and get an immediate response on an oscilloscope screen where the ion current is displayed. The transmission (fraction of the intensity that gets through) can be optimized in this way. The beam may also be positioned in the center of the beam tube.

In the file "CENTRAL.B2M", stored with SAVE and working with the "BEAM LINE 2M" set up (see table 3.4), voltage values are kept of the first try to optimize a beam line using HVSET. Here the transmission was 50%.

4.2 Mass Spectra

4.2.1 The INNOVAT Spectrum Analyser

The files from SCANNER containing current measurements in a specified number of channels are accepted by the spectrum analyser program INNOVAT. At present this program is not located in the Olivetti that I used, so I had to copy the files onto a diskette and then put this diskette into the disk station of the INNOVAT equiped computor.

I did so with the file "PLASMA1.001". Figure 4.1 shows the resulting print-out. Since the screens were just directly copied onto the paper, i.e. there is no special routine in INNOVAT to make nice printer curves, and since INNOVAT makes dots - no lines, it is difficult to tell what the print-out represents. To turn from INNOVAT to EXCEL, as suggested by Helge Ravn, improved the graphics a great deal.

4.2.2 The MS-EXCEL Program

The Micro Soft EXCEL exists in a version for the Olivetti PC. The files "EXCEL.EXE", "EXCEL.INF" and "EXCELHLP.HLP" are stored in the WIN386 directory (see table 3.1) on the Olivetti I used.

EXCEL works with so called charts (data or text matrices) and graphs. The files created with SCANNER can be directly loaded and converted into a long column of a chart. This column represents the entries for the y-axis of a graph. Actually the maximum number of entries are 2000, so I made two graphs for each SCANNER file.

The graph's x-axis could be the channel number, but why not the mass number, since a mass calibration is easy? Figures 4.2 and 4.3 show the graph print-outs corresponding to the files "XE.380" and "XE.400". This is the way the calibration was made:

* The values recorded from the SCANNER file are moved into the second column from the left (column B), to leave room for mass numbers in the first column (column A).[1]

* An investigation shows that there are peaks corresponding to rows 360 and 1387 in the EXCEL chart. These are known to have masses 2 (H_2^+) and 28 (N_2^+ or CO^+). Now since the SCANNER channel, from which the EXCEL row (row) is a linear function, is also about linearly dependent on the square root of the mass of the separated particles (m), the following is the calibration formula:

$\sqrt{m} = C \bullet \text{row} + \text{offset}$

The two value pairs above resolves the constants to be:

C = 0.00377533
offset = 0.05509217

So what is actually typed into the EXCEL chart, column A; row 1 to 2000, to calculate mass number, is:

=ROUND((0.00377535 * ROW() + 0.05509217)^2, 1)

* Rows 1 to 2000 in column A and B are then marked and converted into a graph, just like figures 4.2a and 4.3a.

* $^{132}Xe^{++}$ and $^{132}Xe^{+}$, have mass numbers 66 and 132.[2] Their peaks are found at the EXCEL rows 2140 and 3054, so the calibration constants and the EXCEL statement to type into rows 2001 to 4000 of column A are:[3]

C = 0.00368171
offset = 0.24518295

=ROUND((0.00368171 * ROW() + 0.24518295)^2, 1)

* Rows 2001 to 4000 in column A and B are marked and converted into a graph with the same procedure as above. Figures 4.2b and 4.3b show two graphs made this way.

---

[1]The EXCEL commands are found in "Microsoft EXCEL Reference Guide" (see the literature list).
[2]The charge to mass ratio q/m determining the separation, makes doubly charged particles becomes separated as had they had half their mass.
[3]The rows 4001-4096 were also supported by the SCANNER file, but they were not cared for.

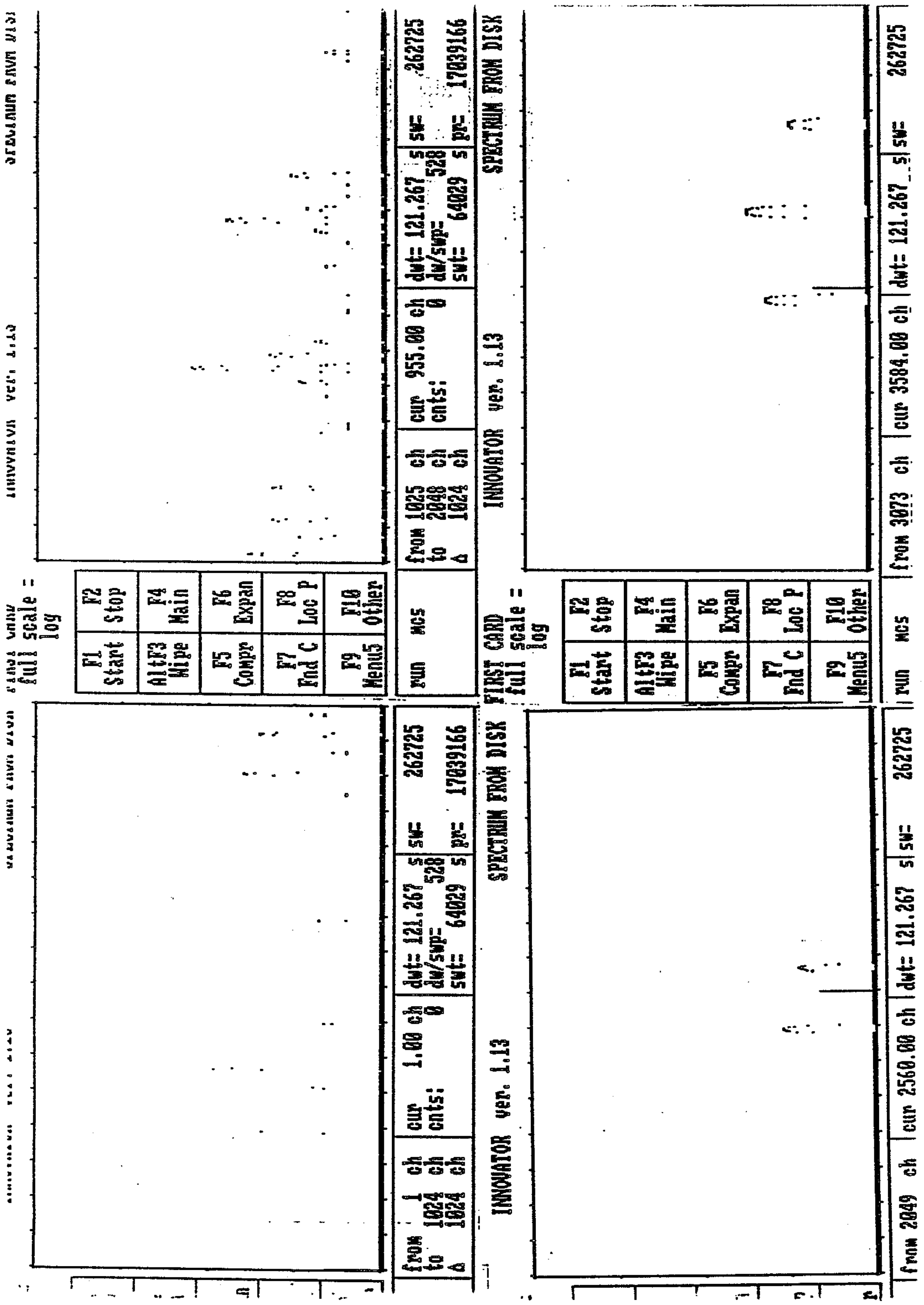


Fig. 4.1 The INNOVAT print-out of the mass spectrum, created with SCANNER and stored in the file "PLASMA1.001".

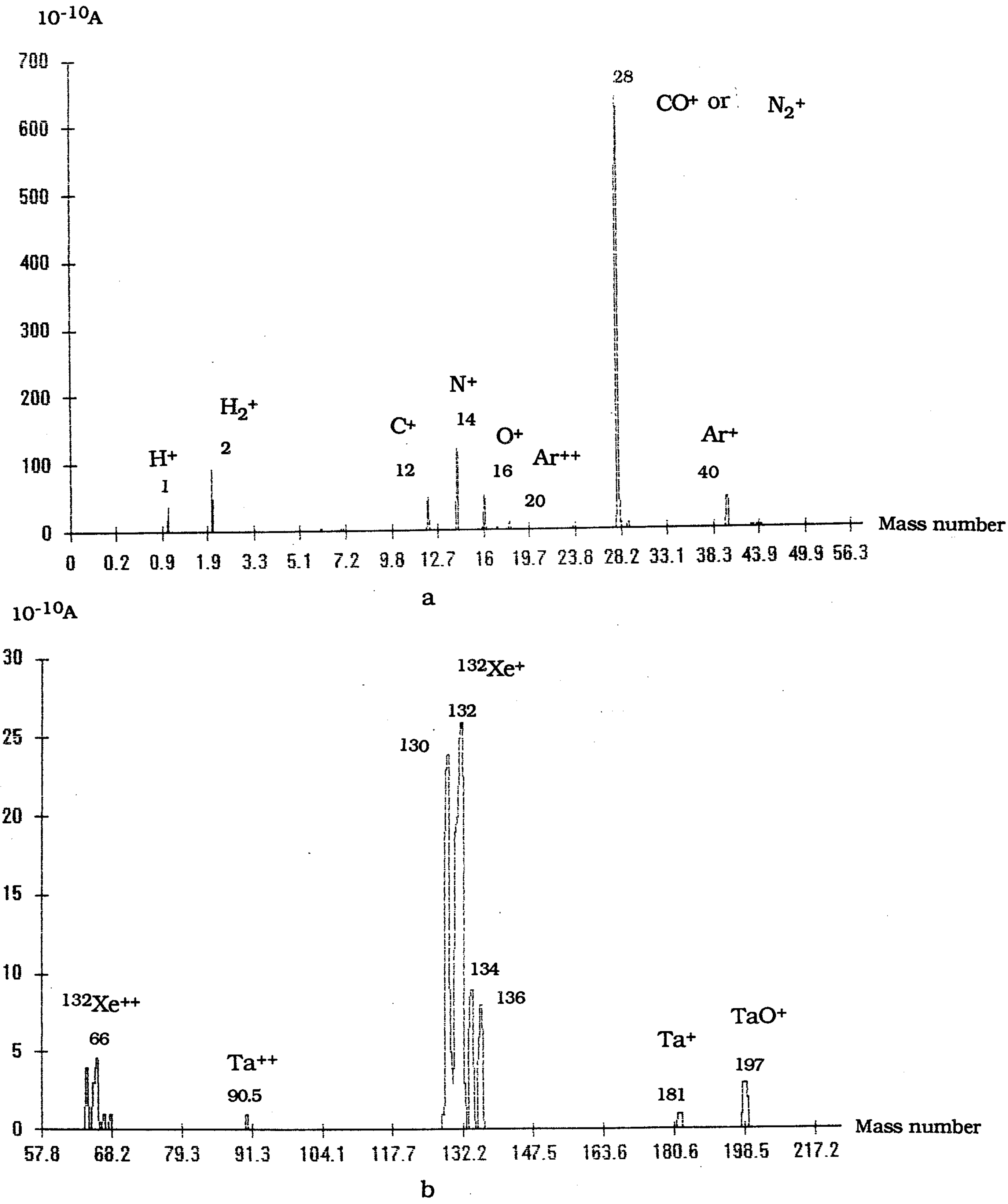


Fig. 4.2 Xenon ion source filament heated with 380A current. a) shows masses ≤ 57. b) shows masses > 57.

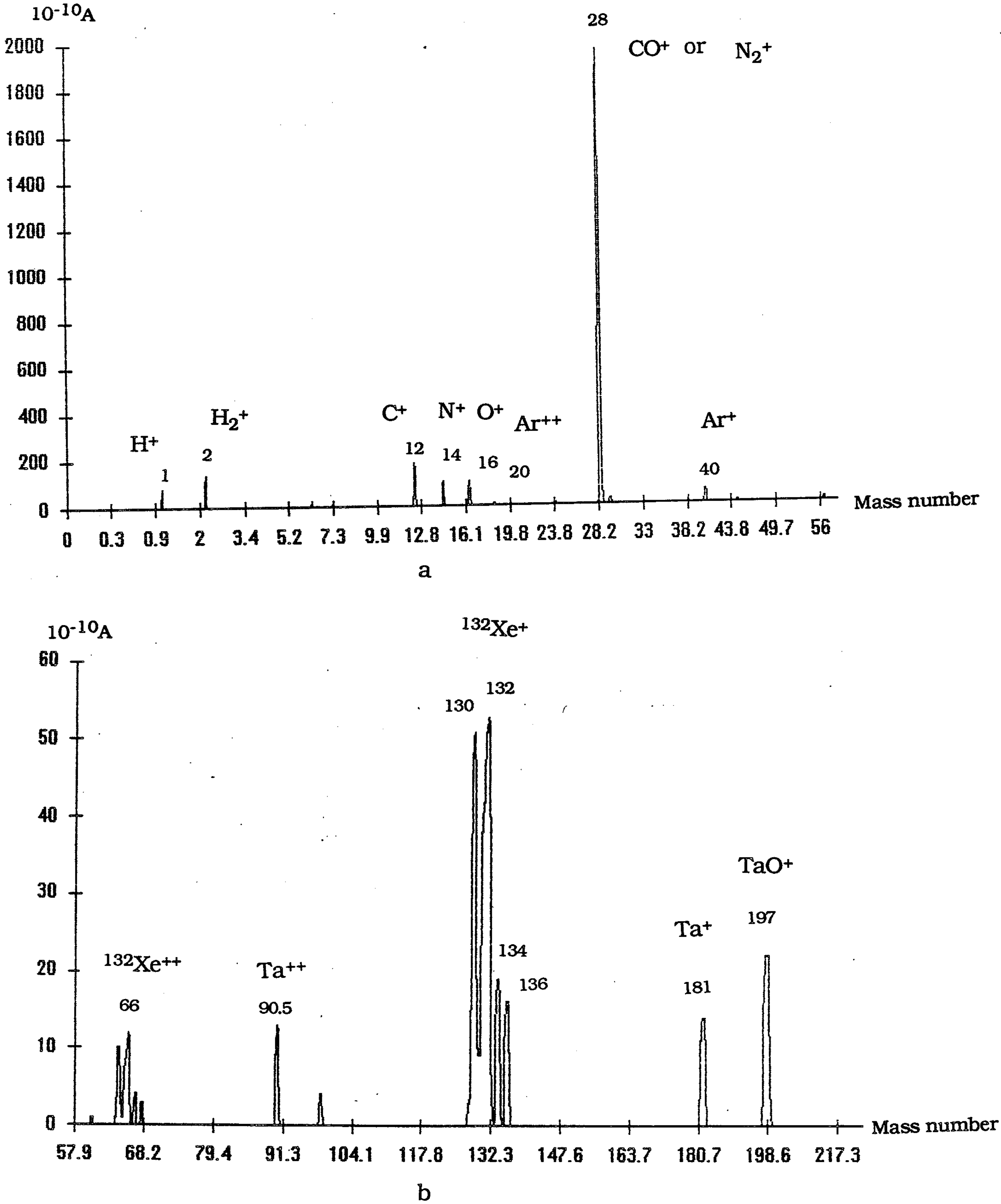


Fig. 4.3 Xenon ion source filament heated with 400A current. a) shows masses ≤ 57. b) shows masses > 57.

4.2.3 The Xenon Mass spectra

My mass spectrometer system is supposed to be used for off line tests of stable ions; mostly "dirt", before the target is put into operation. In the first try a xenon (Xe) filament was used as plasma-discharge ion source, and the target not yet used was lanthanum (La).[1]

An ion source was resistor heated with a current of 380A in the first test (figure 4.2) and 400A in the second test (figure 4.3). The increase in "heating current" showed that the dirt, i.e. C^+, N^+, O^+ and CO^+ became relatively higher, so there is some relation with the heating, but the real cause of the dirt is (was?) not understood. The dirt normally disappears after outgasing.

A waterleak was detected during some experiments at ISOLDE-2 in july 1989. This reduced the efficiency from normally 30% down to only 1%. The cause was found to be a scratch. The stable H_2^+ ion in both Xe spectra may very well come from the water.

The old ion source was argon (Ar). This was still detectable as Ar^+ and Ar^{++}.

The target container and the transfer line to the ion source both are made of tanthalum (Ta). The dirt found due to this is Ta^+, Ta^{++} and TaO^+.

[1]Radioactive Xe was wanted in the on-line run.

Part 3

STEPPING MOTOR CONTROL

1 HOW TO CONTROL THE STEPPING MOTOR

1.1 The Purpose of the System

The stepping motor control system is designed to let a motor run 665 steps at about 1.8 seconds.[1] This sequence of steps is repeatedly executed every fifth second (figure 1.1). The motor is driving a band equiped with thin foils, which will accordingly be exchanged every fifth second (figure 1.2). The foils are supposed to collect the incoming ion beam; each foil in turn.

An acceleration ramp in the step frequency is put on the motor to drive it from rest up to maximum velocity. A retardation; from maximum velocity down to rest, is also made. This way the operation becomes much smoother than would be the case if an abrupt transition from zero up to maximum step velocity was given (figure 1.1 again).

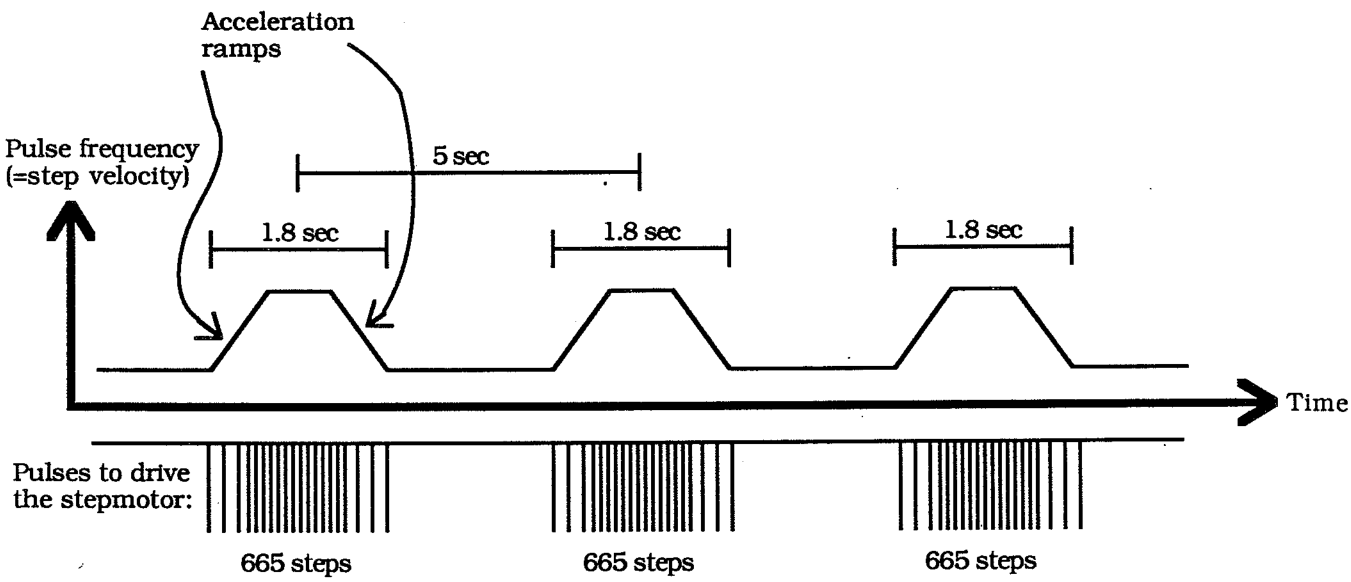


Fig. 1.1 How the pulses that stear the stepmotor are distributed in time. The operation is made smoother by letting the pulse frequency (and thereby the motor run velocity) accelerate in the beginning and retardate in the end of each 665-step-sequence.

[1] A "step " is a small mechanical rotation that is executed every time an electrical pulse is sent to the motor.

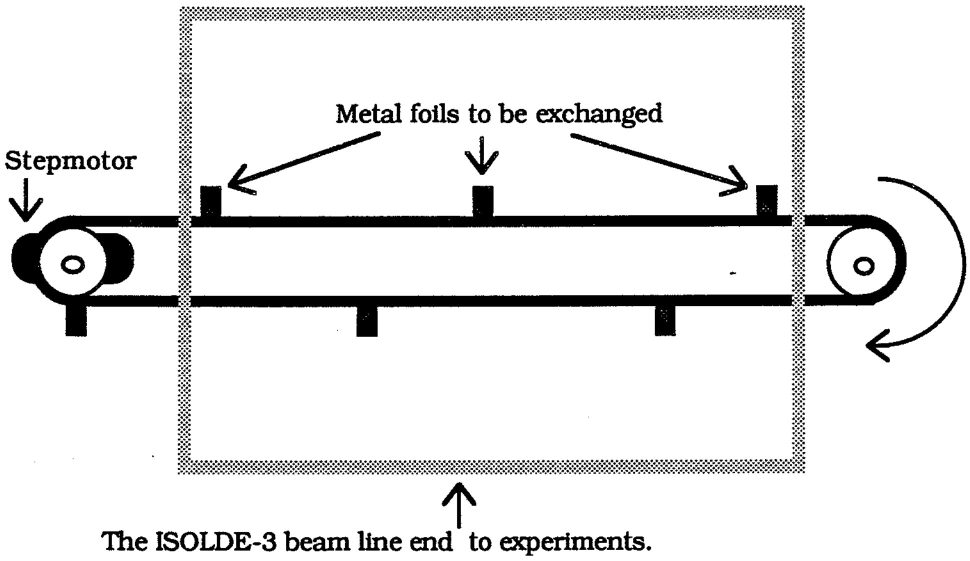


Fig. 1.2 The stepmotor driving a band of foils, who are exchanged with 5 second intervals.

The experiment that uses the stepping motor is located in the Proton Hall, and the beams come from ISOLDE-3.

1.2 The Parts

1.2.1 Equipment

The stepping motor control system consists of:

* The Pericom Terminal.

* The HSMC 20A stepping motor controller with a 5V net unit.

* A box with 8 switches.

* An electronics rack with counters, a pulse generator and 24V supplies.

* The motor driving unit.

* The stepmotor.

The system is shown in figure 1.3. The different functions of the system are illustrated by giving the cables in the figure different grey scale patterns.

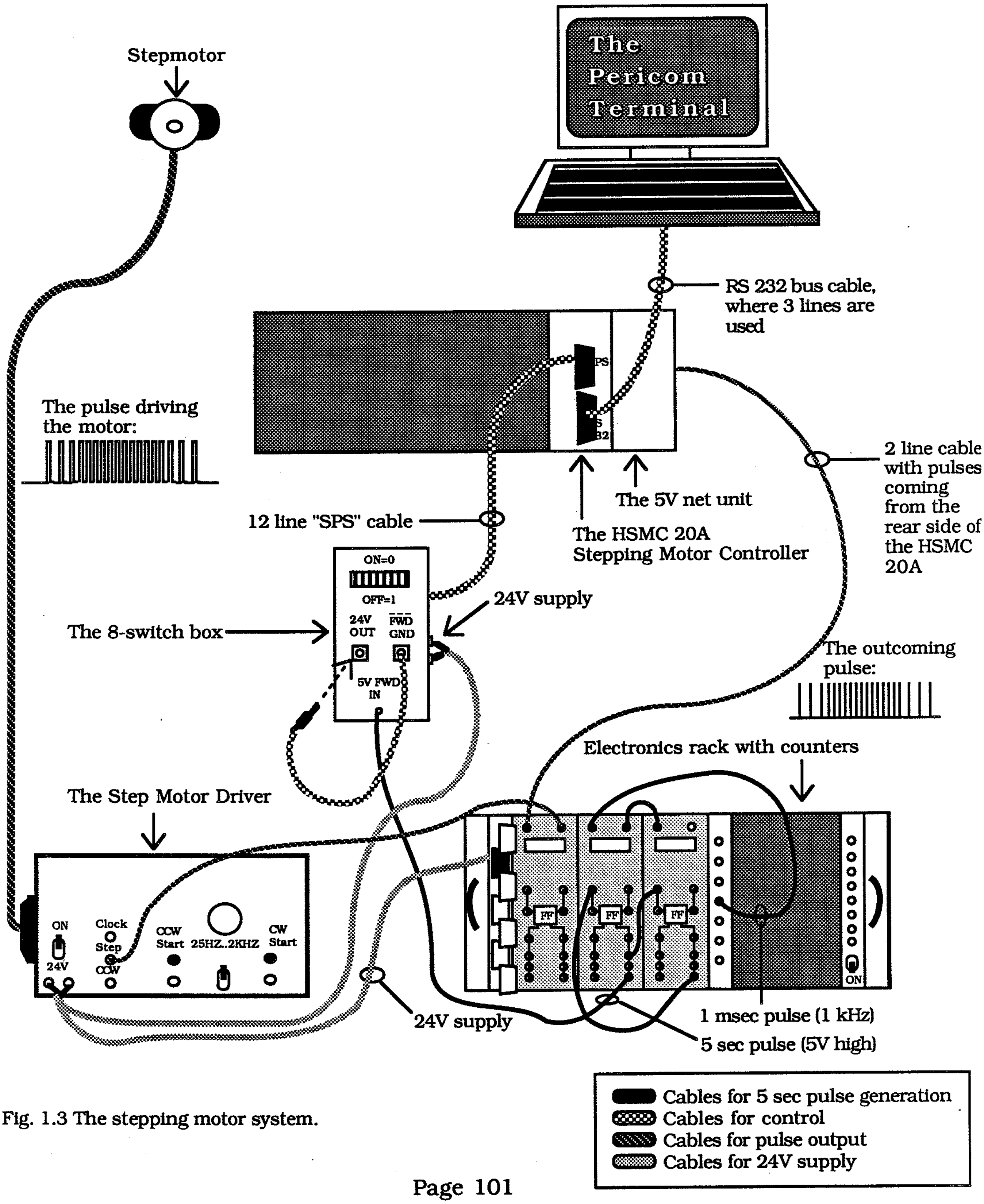


Fig. 1.3 The stepping motor system.

1.2.2 5 Seconds Pulse Generation[1]

First 1 msec pulses are generated in one of the modules of the electronics rack. These pulses are trigging a counter (the middle one of the three in the rack) which is set to give a signal after 5000 counts. The rightmost counter serves as a 10 msec delay, after which the middle counter is reset to zero, for a new counting round up to 5000.

The result is 5V high and 10 msec long pulses delivered every 5:th second to the FWD (forward) input of the 8-switch-box.

1.2.3 The Control Part[2]

The 8-switch-box has two functions: it can set the HSMC 20A mode of operation, and it amplifies the maximum amplitude of the incoming signal (with a periodicity of 5 seconds) from 5 volts up to 24 volts, which is accepted by the HSMC 20A FWD input for trigging a new round of step pulses.

The Pericom Terminal is also to set the mode of operation, and to trigg pulse sequencies. It is connected via the serial RS 232 bus to the HSMC 20A. The special feature with the terminal is that it can modify different step parameters, such as step acceleration, number of steps in a sequence e.t.c. These may then be stored in an EEPROM memory on the HSMC 20A card.

Chapter 2 will deal with terminal operations.

1.2.4 The Output Part[3]

The HSMC 20A delivers the pulse trains as small peaks, with far too little power to stear the stepmotor directly, but led through an input and output of a counter, and then through the stepmotor driver the signal is accepted by the motor.

1.2.5 The 24 Volt Power Supply[4]

Both the 8-switch-box and the motor driving unit need 24 volts for their functioning. This tension is delivered by a power module in the electronics rack.

1.2.6 Further Details

Chapter 3 will give a more detailes about the different instruments in the system for stepping motor control.

---

[1]The units connected with black pattern cables in figure 1.3.
[2]The units connected with big square pattern cables in figure 1.3.
[3]The units connected with diagonally striped cables in figure 1.3.
[4]The units connected with small square pattern cables in figure 1.3.

1.3 To Start

The following will make the stepmotor run in the previously dercribed way:

* The switches on the 8-switch-box should be like:

```
      ON=0
0000
      1111
12345678
      OFF=1
```

* The "24V OUT" and "GND" on the 8-switch-box should be disconnected. (The banana contact not plugged in.)

* The electronics rack and the motor driving unit may now be powered on, and the net cable ot the HSMC 20A plugged in. (The terminal need not to be on because the 8-switch-box manages the control.)

* If everything is working the motor will step forward the predefined number of steps (665) every fifth second. (It may be necessary to push the start buttons on the electronic counters to initiate the operation.)

1.4 To Enter Terminal Mode

To enter terminal mode do the following:

* The "24V OUT" and "GND" on the 8-switch-box should now be connected. (The banana contact plugged in - see figure 1.3.) Otherwise there should be a forward signal disenabling the transfer to terminal mode.

* Set the switches of 8-switch-box to:

```
      ON=0
0
 1111111
12345678
      OFF=1
```

* Turn on the terminal.

* If the command line "E, M, P, L, S, H, ?" does not show up "press the reset button", i.e. switch 8 on the 8-switch-box is first switched to "ON=0" then back to "OFF=1".

* Terminal handling is described in chapter 2.

To reenter from terminal mode do the following:

* Make sure that the command line "E, M, P, L, S, H,   ?" is on the screen. (Send for instance a reset signal as described above.)

* Press the RETURN key so the symbol '#' appears on the screen.

* On the 8-switch box disconnect "24V OUT" and "GND" and make the settings:

```
_____ON=0
0000
_____1111
12345678
     OFF=1
```

2 TERMINAL OPERATIONS

2.1 Dialog Operation

After power up or entering terminal mode the following command line appears on the Pericom terminal:

E, M, P, L, S, H, ?

Where:

E = Enter positions
M = Modify parameters
P = Positioning procedure
L = Read parameters
S = Store parameters
H = Manual (Hand) operation

The desired function is selected by pressing the desired key. This will send the corresponding ASCII character to the HSMC 20A.

2.2 Entering Positions (E)

The E command can set up to 24 different positions and store them into the internal EEPROM of the HSMC 20A (24 bits/position). The maximum value is decimal 16,777,215.

The '+' key will move up and the '-' key down in the position table. How values are entered is shown in figure 2.1.

Position	Value	Input sequence
ØØ	Ø/	- (minus)
23	20000/	- (minus)
22 a)	10000/15000	15000+
23	20000	+
ØØ	Ø/	+
Ø1 b)	100/50:200	50:200+
Ø2	400/	+
Ø3	800/	+
etc.		

a) A new value can be entered directly after the slash character.

b) A wrong entry is cleared with a colon (:), followed by the desired entry.

Fig. 2.1 How position values are entered under the E command.

The RETURN key terminates the E function. The modified values will then be transferred into the EEPROM.

Causion should be taken so that the E command is not used too often, because the EEPROM can only handle a maximum of 10,000 write cycles during the entire life of the HSMC 20A.

Note that the positions entered with E are only executable in manual mode.[1]

2.3 Modify Parameters (M)

The parameters ACCELERATION (AR), MAXIMUM STEPPING RATE (FM) and START/STOP FREQUENCY (FC) are modified using the M command. If you agree with a shown value you just type RETURN, otherwise do the following:

* Press the colon key ':'.

* Enter the new value in decimal notation.

* Press RETURN.

This sequence can be repeated as desired.

The number of steps in the acceleration ramp (SR) and the time they will take (TR) are calculated and displayed automatically at the end of the M function.

Figure 2.2 shows an example of the M command and table 2.1 give valid ranges for the parameters.

```
AR 220/:100          Entering of acceleration in (kHz/s)
FM 12000/:20000      Entering maximum frequency in (Hz)
FC 500/:350          Entering start/stop frequency in (Hz)

SR 2000              Output of maximum ramp steps
TR 200               Output of maximum ramp time in (ms)
```

Fig. 2.2 How parameter values are entered under the M command.

Table 2.1

Valid range for parameters

	allowed values:	initial setting:
AR:	1 to 999 (kHz)	100 (kHz/s)
FM:	33 to 20'000 (Hz)	20'000 (Hz)
FC:	33 to 20'000 (Hz)	500 (Hz)

[1]The manual mode, using the 8-switch-box,will be explained in section 3.3.

2.4 Positioning (P)

Definitions:

SP = Starting (current) Position
+D = relative difference forward
-D = relative difference reverse
NP = New (target) Position

The SP value will be displayed after pressing the P key. Default value is 8,388,608. This is to enable motions in both directions.

The SP value can be re-defined by pressing the ':' key, followed by a decimal value and RETURN.

After the SP is entered there are three possible keys to press:

* + key for relative position forward (cw).

* - key for relative position reverse (ccw).

* RETURN to enter the absolut target position (NP).

The desired value is then typed in followed by RETURN. The HSMC 20A now either calculates the new absolute position (in the case of + or -) or the relative difference (in the case of RETURN). Figure 2.3 gives examples.

At this stage the following command line will appear:

A, B, X, LF, CR ?

Where:

A = Ramping positioning (with the acceleration ramp)
B = Start/stop positioning (without the acceleration ramp)
X = Test run in acceleration mode
LF = Line feed (Control-J key) to terminate the P function
CR = RETURN to skip execution

The desired function is started by pressing the respective key. Characters A, B and X are echoed on the Pericom screen, and when the positioning is finished a '*' character is sent by the HSMC 20A, like figure 2.4a shows. If the execution is aborted the string "**" is sent instead (figure 2.4b).[1]

A test run (X) causes continuous motion of the stepping motor forth and back until aborted.[2]

Whenever the entering of a key is possible the line feed (the Control-J key) can terminate the P command.

[1]This will happen if at least one of the HSMC 20A signal lines LS1 or LS2 is set low.
[2]The STOP signal can for instance abort the X run. STOP is sent with switch 7 on the 8-switch-box.

```
SP 8'388'608/:Ø         new starting position set zero
+D/1'500                difference of + 1'500 steps entered
NP 1'500/               new position (target) automatically
                        determined

SP 12'700/              SP not changed (carriage return only)
NP/10'300               new position (new target) entered
-D 2'400                Difference of -2'400 steps
                        calculated by HSMC 20A
```

Fig. 2.3 Examples of how to enter values under the positioning command P.

```
A, B, X, LF, CR  ?   B*
A, B, X, LF, CR  ?   A*          a


A, B, X, LF, CR  ?   A**         b
```

Fig. 2.4 Examples illustrating what is echoed on the Pericom screen a) after a succesful execution, and b) after an aborted execution.

2.5 Read Parameters (L)

The parameters ACCELERATION (AR), MAXIMUM STEPPING RATE (FM) and START/STOP FREQUENCY (FC) are read from an internal non-volatile memory with this command. The L command is also automatically executed after a power-on reset.

2.6 Store Parameters (S)

The parameters ACCELERATION (AR), MAXIMUM STEPPING RATE (FM) and START/STOP FREQUENCY (FC) are stored in an internal non-volatile memory with this command. The L command is used to constantly save modifications made with the M command.

2.5 Manual Operation (H)

The H command selects the manual functions of the HSMC 20A, provided that the signal line S6 is low. (This line is monitored by switch 4 on the 8-switch -box.)

When manual operation is entered the HSMC 20A first transmits the message "HAND".

The manual operation is terminated when S6 goes high and the HSMC 20A is not busy.

It is also possible to select the manual operation by pressing RETURN so the sign '#' appears on the screen. In this case a reset is needed before the reentering of the terminal operation mode.

3 THE INSTRUMENTS

3.1 Introduction

The units in the stepping motor system are here further revealed and some complements are given, to that which was said in connection to figure 1.3. Appendix D will then give the technical details.

3.2 The Pericom Terminal

The SET UP key was used to set the following communication parameters:

On Line
Full Duplex
8 Dtat, Non, 1 Stop
Parity Not Checked
Transmit Speed = 4800
Receive Speed = 4800

"On Line" was choosen in the SET UP Directory menu, whereas the other parameters were choosen in the SET UP Communication menu.

On the backside of the terminal there is a 25 pin plug for the RS 232 bus. This was connected to the HSMC 20A controller. Only three of the lines were used: TX (serial transmission), RX (serial receiving), ang GND (signal ground).

3.3 The HSMC 20A and the 5V Net Unit

The system uses a HSMC 20A from Rubran AG Zürich to make pulse trains that will finally after amplification steer the motor. The unit needs 5 volts to function. This is supplied by the VERO TRIVOLT net supply, where the 5V output was coupled to the +5V input of the HSMC 20A. Unfortunately there is no power on switch, so the net cable connector (Swiss standard) simply have to be plugged in before, and plugged out after usage.

Fig. 3.1 Me holding the 8-switch-box, pictured in the Proton Hall.

3.4 The 8 Switch Box

Figure 3.1 shows me holding the 8-switch-box. With the box it is possible to choose between terminal mode and manual mode. The terminal mode (using the Pericom) has already been described in chapter 2. However the manual mode will be shown here.

The manual operation is simply different configurations of the switches as table 3.1 explains.[1]

The registers referred to in the table are those loaded into EEPROM by the terminal E command. I loaded 665 into register 3, and this is also the one used in the final operation of the stepping motor system. It is however possible to use other positionings (number of steps) from other registers or from another E command settings.

The FORWARD line has been connected to the output of a transistor, so it should not be used, because it will shortcut the collector to +24V. Instead the transistor serves as an amplifier to let 5 volt input signals be converted to 24 volt FORWARD trigg pulses.

[1]Switch configurations not shown are not used. There are also some other features about the HSMC 20A that have not been used. If you are interested see appendix D.

Table 3.1

The switch settings on the 8-switch-box and their meanings:

Switch 8	7	6	5	4	3	2	1	Function
1	1	1	1	0	0	0	0	Positions as told by register 3
1	1	1	1	0	0	0	1	Positions as told by register 7
1	1	1	1	0	0	1	0	Positions as told by register 11
1	1	1	1	0	0	1	1	Positions as told by register 15
1	1	1	1	0	1	0	0	Positions as told by register 19
1	1	1	1	0	1	0	1	Positions as told by register 23
1	1	1	1	1	1	1	0	Terminal operation mode
1	1	1	‾_‾	X	X	X	X	FORWARD trigging (not to be used!)
1	1	‾_‾	1	X	X	X	X	REVERSE trigging
1	‾_‾	1	1	X	X	X	X	STOP signal
‾_‾	1	1	1	X	X	X	X	RESET signal

The HSMC 20A uses negative logic:

0 = the switch is ON (high)
1 = the switch is OFF (low)
X = don't care
‾_‾= switch from 1 to 0, then back to1

FORWARD is disabled if the 24V output, after the single-transistor amplifier, is connected to ground (with a banana contact cable).

REVERSE may be trigged but the motor driving unit is only installed to drive the motor in one direction.

3.5 The Electronics Rack

The rack contains 24V power supply, a frequency generator and three counters (digital delays). The middle counter is set to 5000 and the rightmost one to only 10. With 1 kHz oscillations the counters will generate a nice signal (10 msec peaks with 5 seconds in between). The IN and OUT of the leftmost counter amplifies the HSMC 20A pulses for the motor driving unit.

3.6 The Motor Driving Unit

The stepping motor driver unit from HMI-Berlin was modified to be used in the present system: The IC circuits 7404, 7402 and 74132 have been pulled out and replaced by sockets (scetches are drawn in appendix D). Removable connections to the "Step" and "CCW" positions in the circuit have been made. This has the disadvantage that the Power On LED indicator is not illuminated when the unit is working, but instead it has been possible to skip the disturbing 30 msec delay for incoming pulses.

LITERATURE LIST

Part 1

E. Hagebø, P. Hoff, O.C. Jonsson, E. Kugler, H.L. Ravn, S. Sundell and B. Vosicki
"PRODUCTION OF SHORT-LIVED ISOTOPES AT THE ISOLDE ON-LINE MASS-SEPARATOR"
Presented at the "2nd International Conference on Nuclear and Radiochemistry", Brighton
July 1988

H.J. Kluge (Ed.)
"ISOLDE USER'S GUIDE"
CERN, Geneva
July 1986

H.L. Ravn
"EXPERIMENTS WITH INTENSE SECONDARY BEAMS OF RADIOACTIVE IONS"
Physics Reports, volume 54, number 3
North-Holland Publishing Company, Amsterdam
August 1979

H.L. Ravn and B.W Allardyce
"ON-LINE MASS SEPARATORS"
CERN, Geneva
May 1987
To be published in:
"Treatise on Heavy-Ion Science" (Ed. D.A Bromley)
Volume 8: "Nuclei far from stability"
Plenum Press, New York

Part 2

Brain boxes
"Elite IEEEE 488 Interface"

IBM
"IBM Personal Computer General Purpose Interface Bus Adapter Programming Support"
IBM Corp.
1984

O.C. Jonsson
"ISOLDE-2 BEAM ADJUSTMENT SYSTEM user's guide"
CERN, Geneva
1984

Keithley
"Model 4853 Instruction Manual"
Cleveland, Ohio
1984

Lecroy
"Model HV 4032A High Voltage System"
Spring Valley, N.Y.
1980

Lecroy
"Model 8901 CAMAC to GPIB Interface"
Palo Alto, CA
1981

Logitech
"Getting The Most From Your Mouse"

Microsoft
"Excel Reference Guide"

Nuclear Enterprises LTD
"Multidac Type 9091"

Olivetti Operating Systems and Language Library:
"MS-DOS User's Guide"
"MS-WINDOWS User's Guide"
"MS GW-BASIC INTERPRETER under MS-DOS User's Guide"

Part 3

Rubran AG
"Stepping Motor Controller HSMC 20A"
Zürich

Appendix A

ALLOCATIONS FOR THE HV4032A CRATES

ADDRESSES AND CABLE CONNECTIONS

The six LeCroy HV 4032A, 32 photomultiplier high voltage system crates are arranged according to the beam lines as figure 1 shows. The addresses for the different beam elements, both for the positive and for the negative HV supplies, are shown in the figures 2 to 7, where the mainframes come in the order:

MF#14
MF#16
MF#15
MF#11
MF#13
MF#10

Cable connections are also shown in the figures 2 to 7 as well as in figure 8.

The figures originate from O. C. JONSSON's "ISOLDE-2 BEAM ADJUSTMENT SYSTEM users guide" (see the literature list).

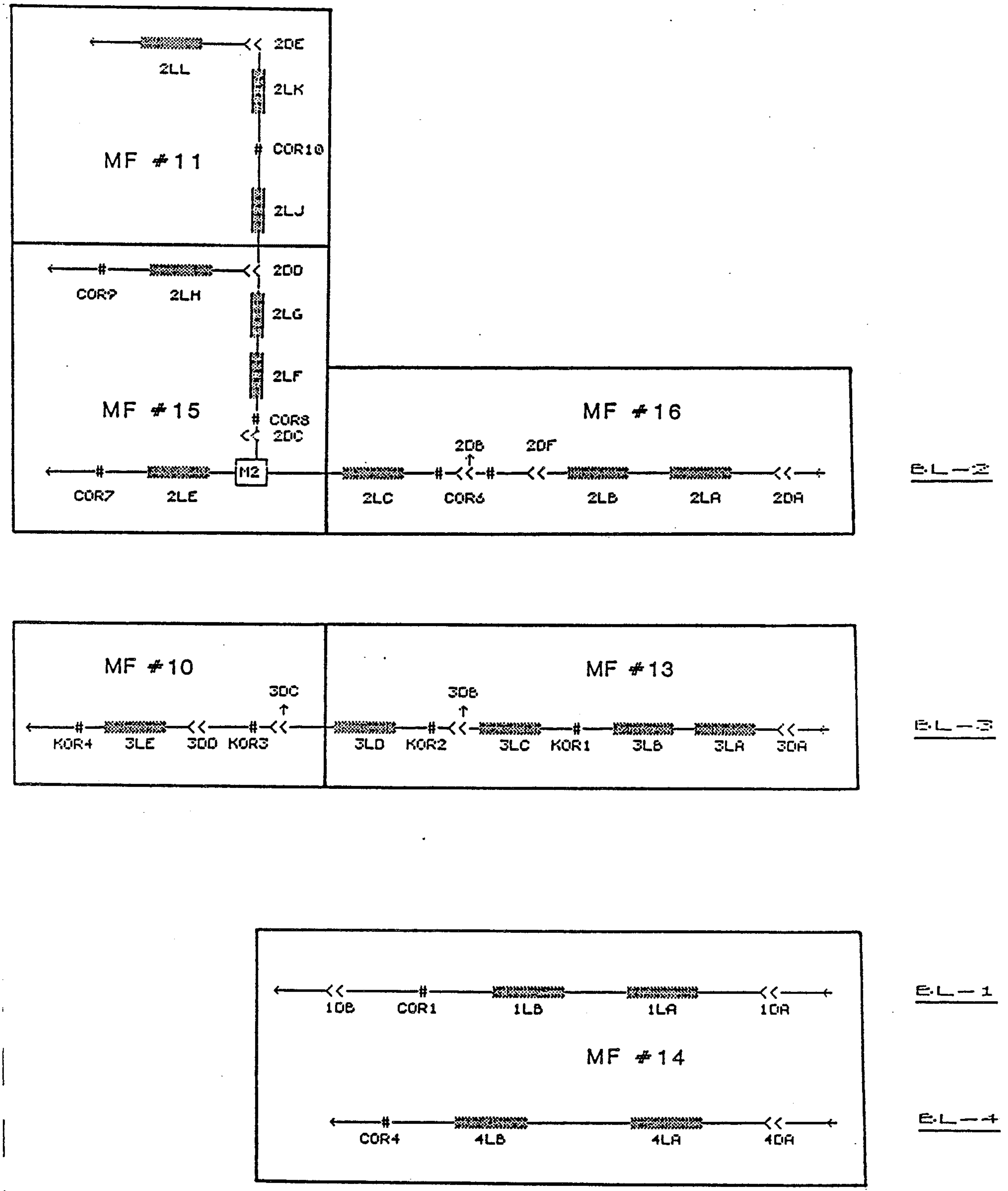
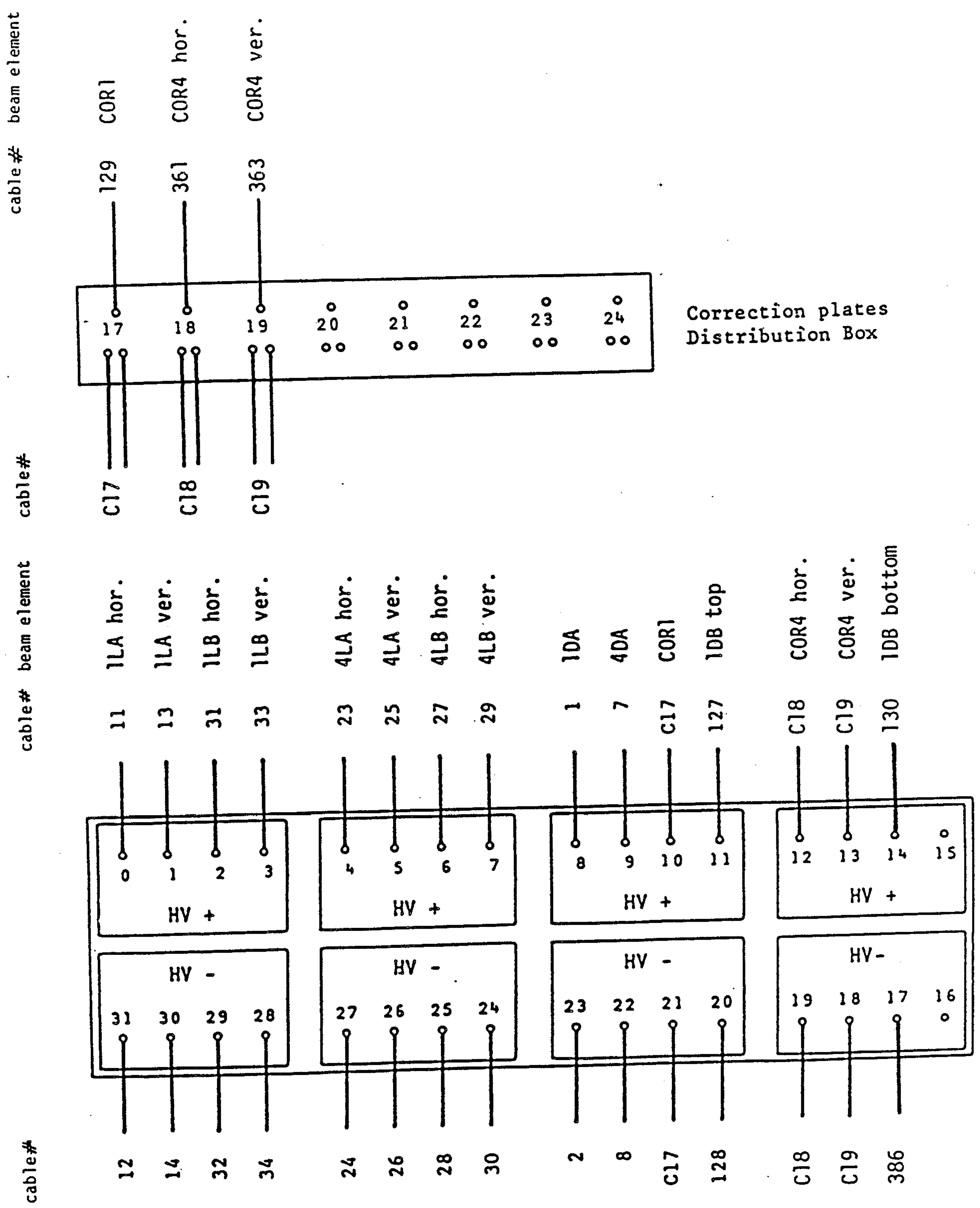
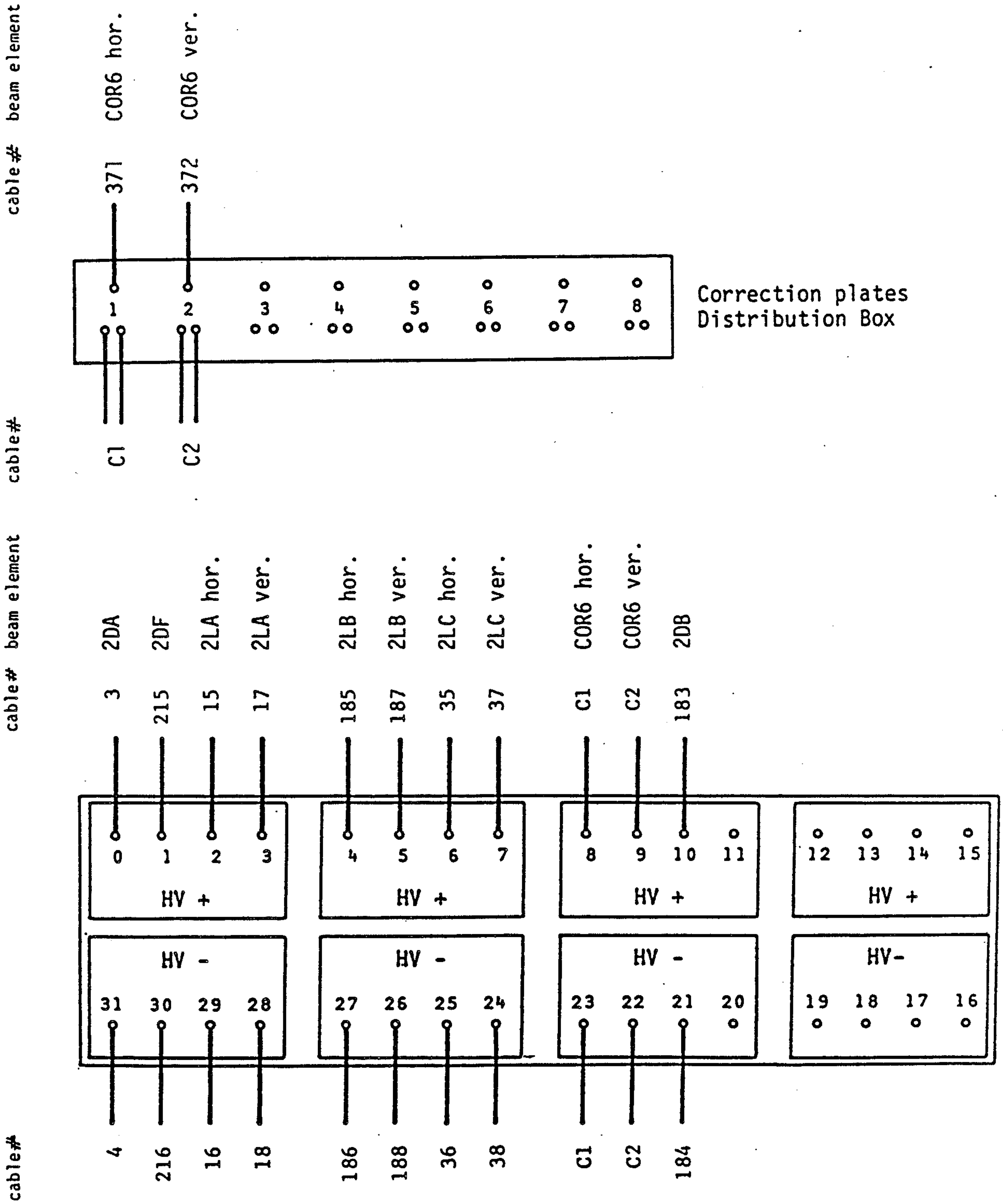


Fig. 1 LeCroy Mainframe distribution to ISOLDE-2 beam lines



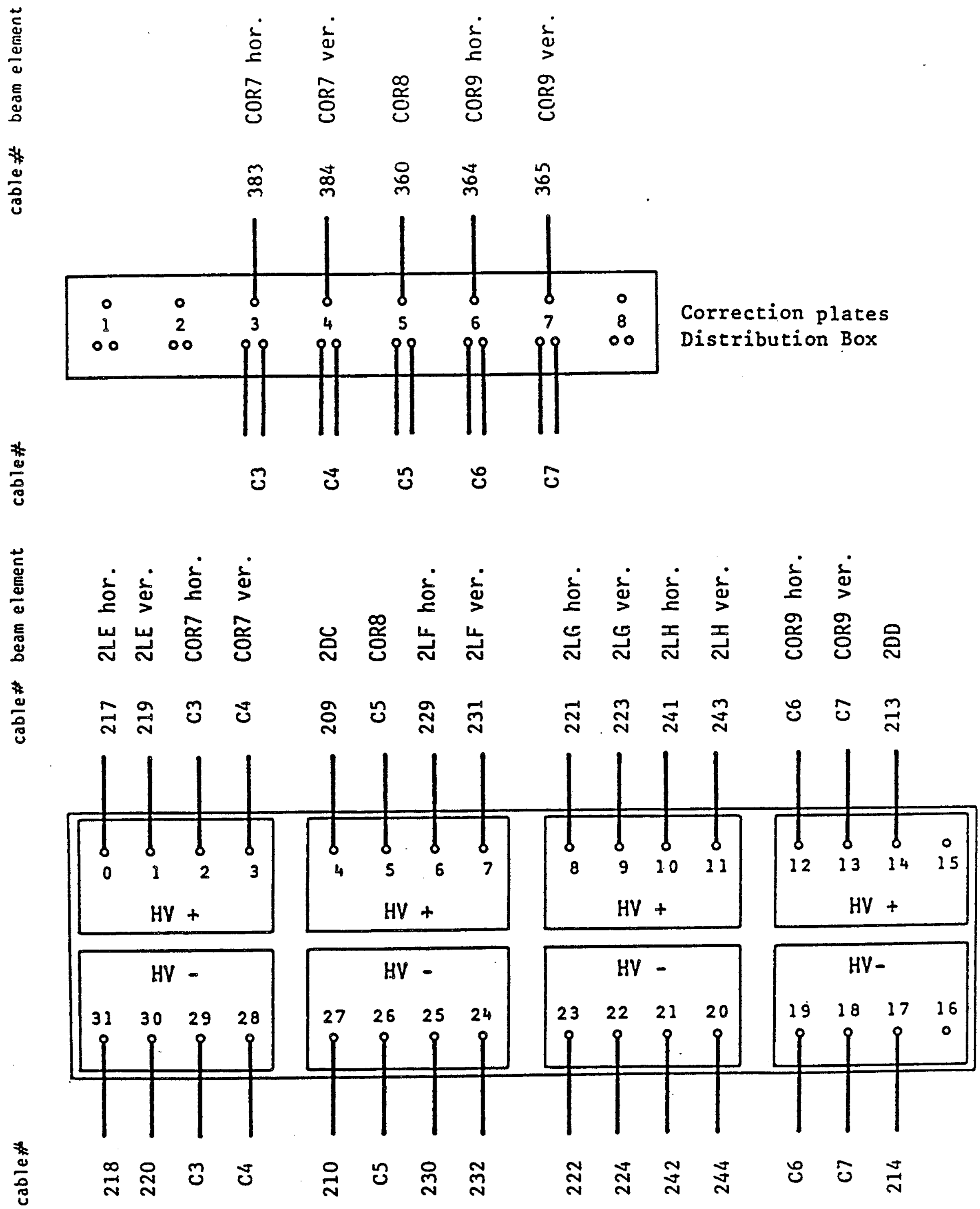
MF #14

Fig. 2 Cable connections to LeCroy 4032A HV-supply



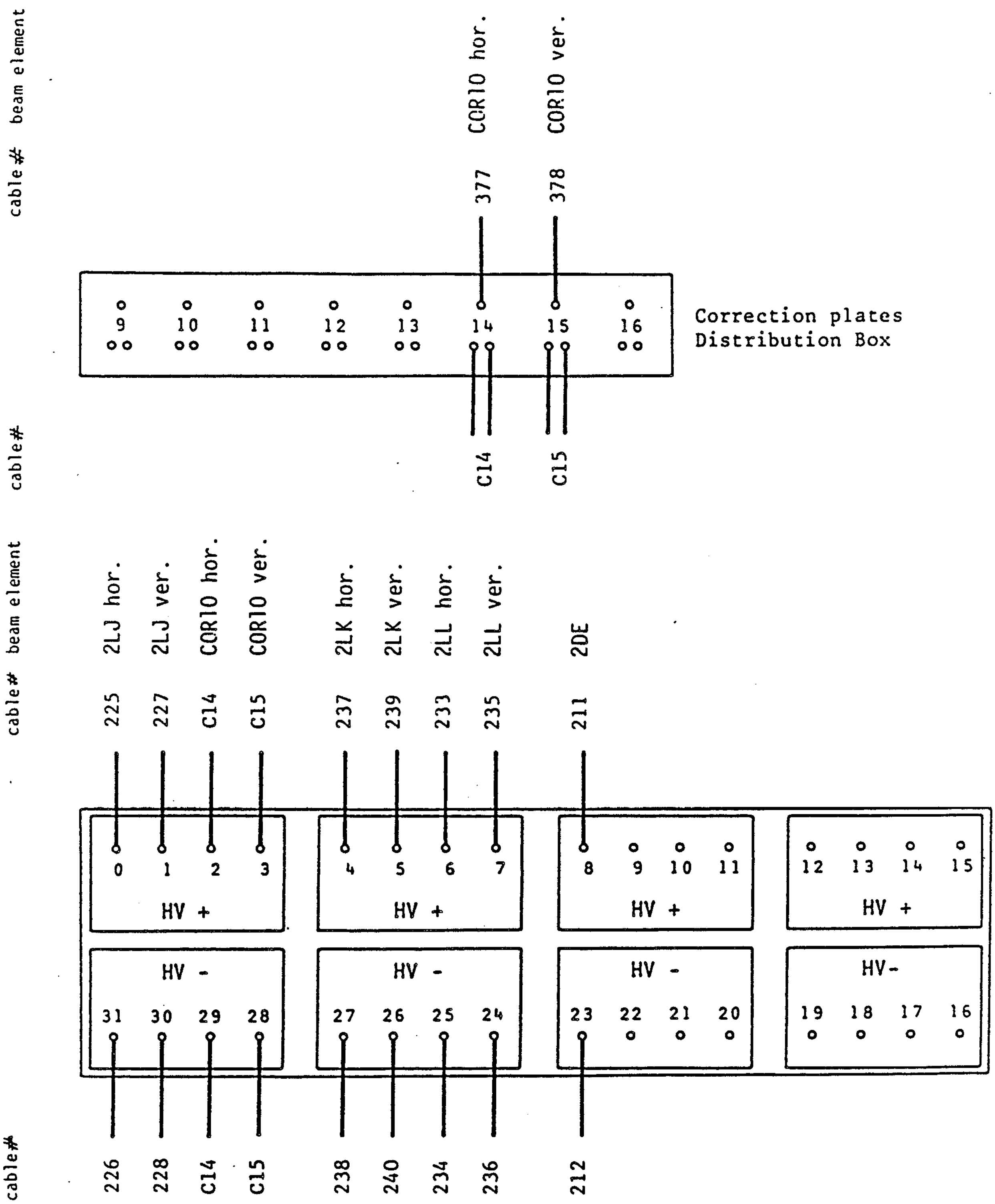
MF #16

Fig. 3: Cable connections to LeCroy 4032A HV-supply



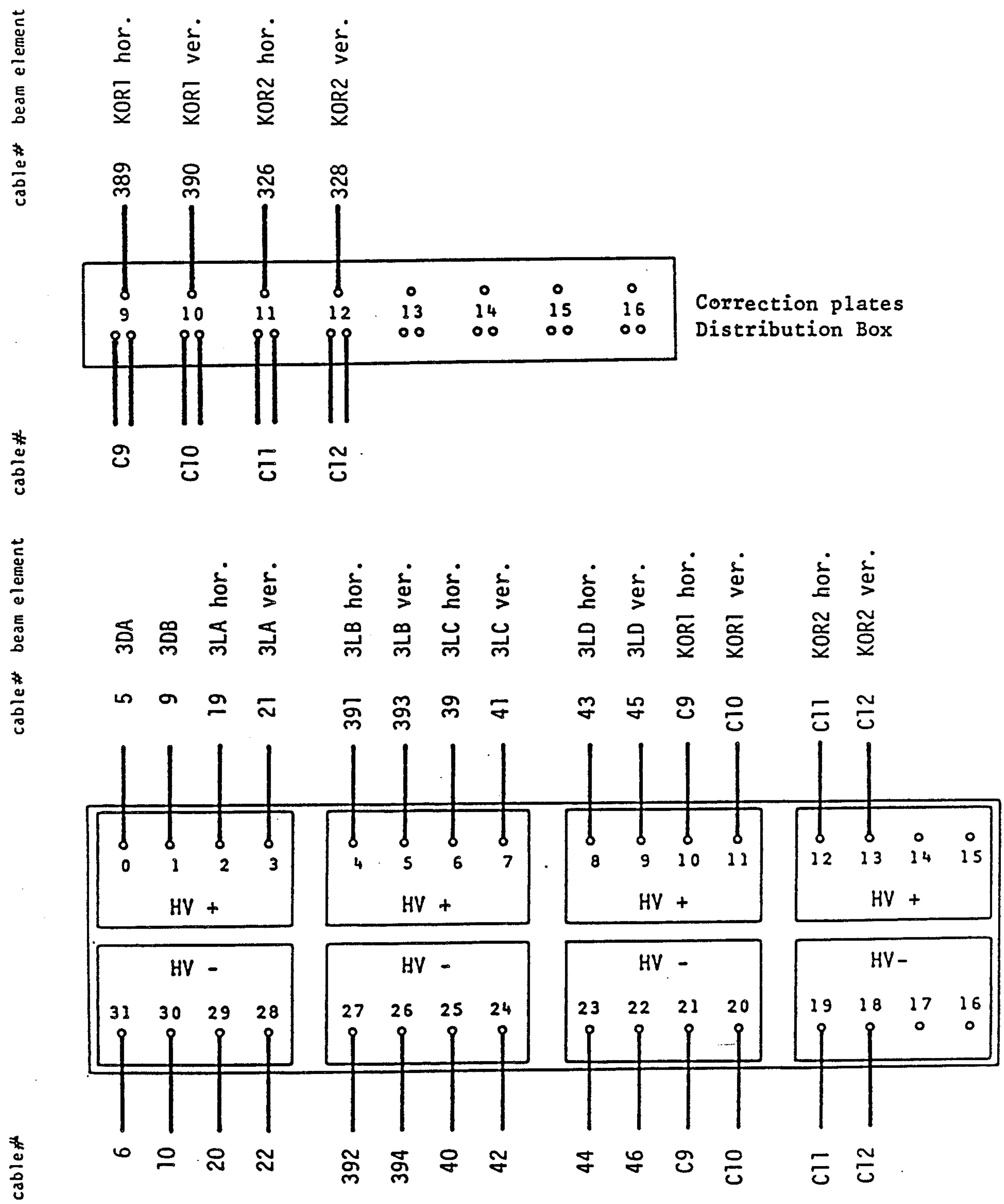
MF #15

Fig. 4 Cable connections to LeCroy 4032A HV-supply



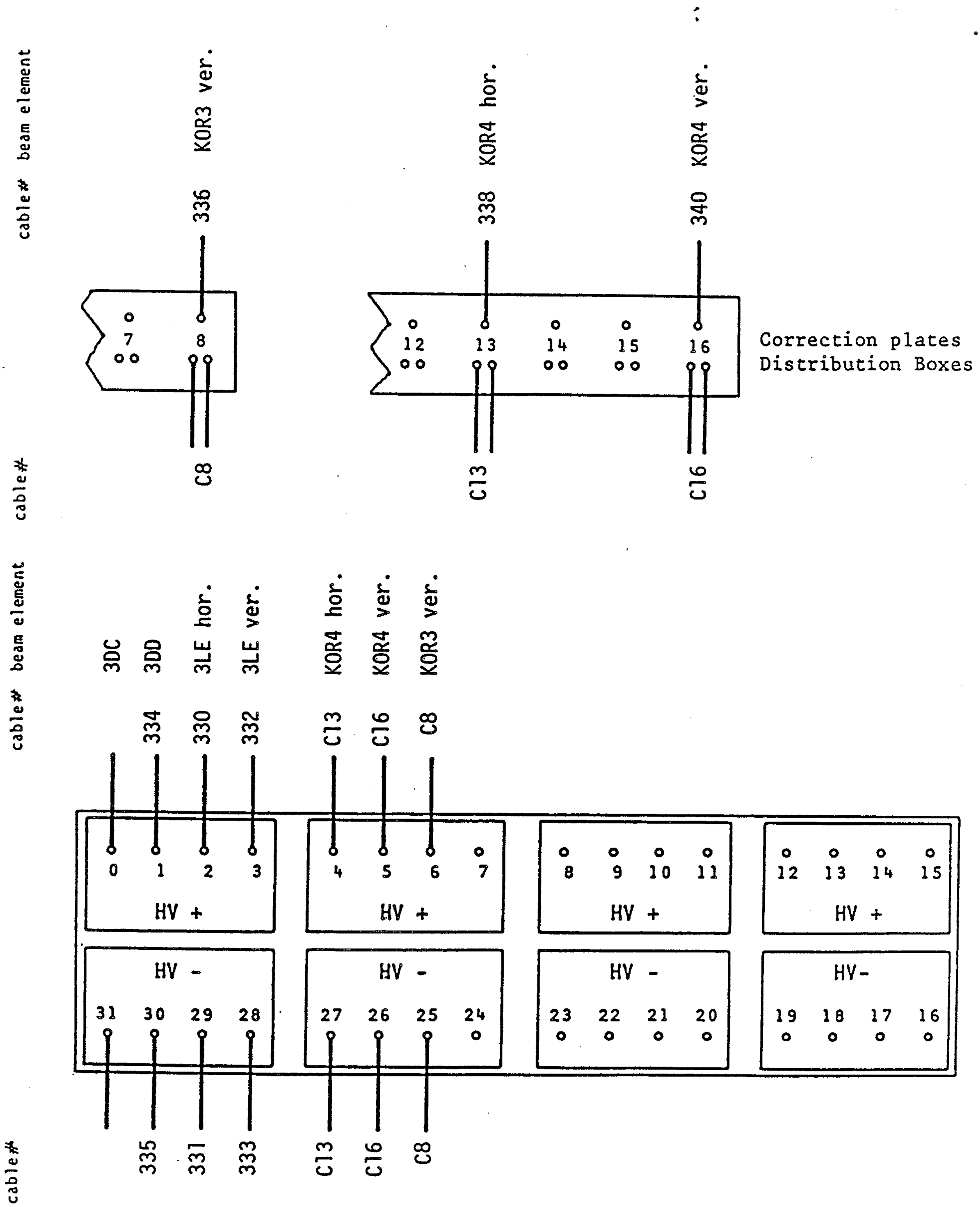
MF #11

Fig. 5 Cable connections to LeCroy 4032A HV-supply



MF #13

Fig. 6 Cable connections to LeCroy 4032A HV-supply



MF #10

Fig. 7 Cable connections to LeCroy 4032A HV-supply

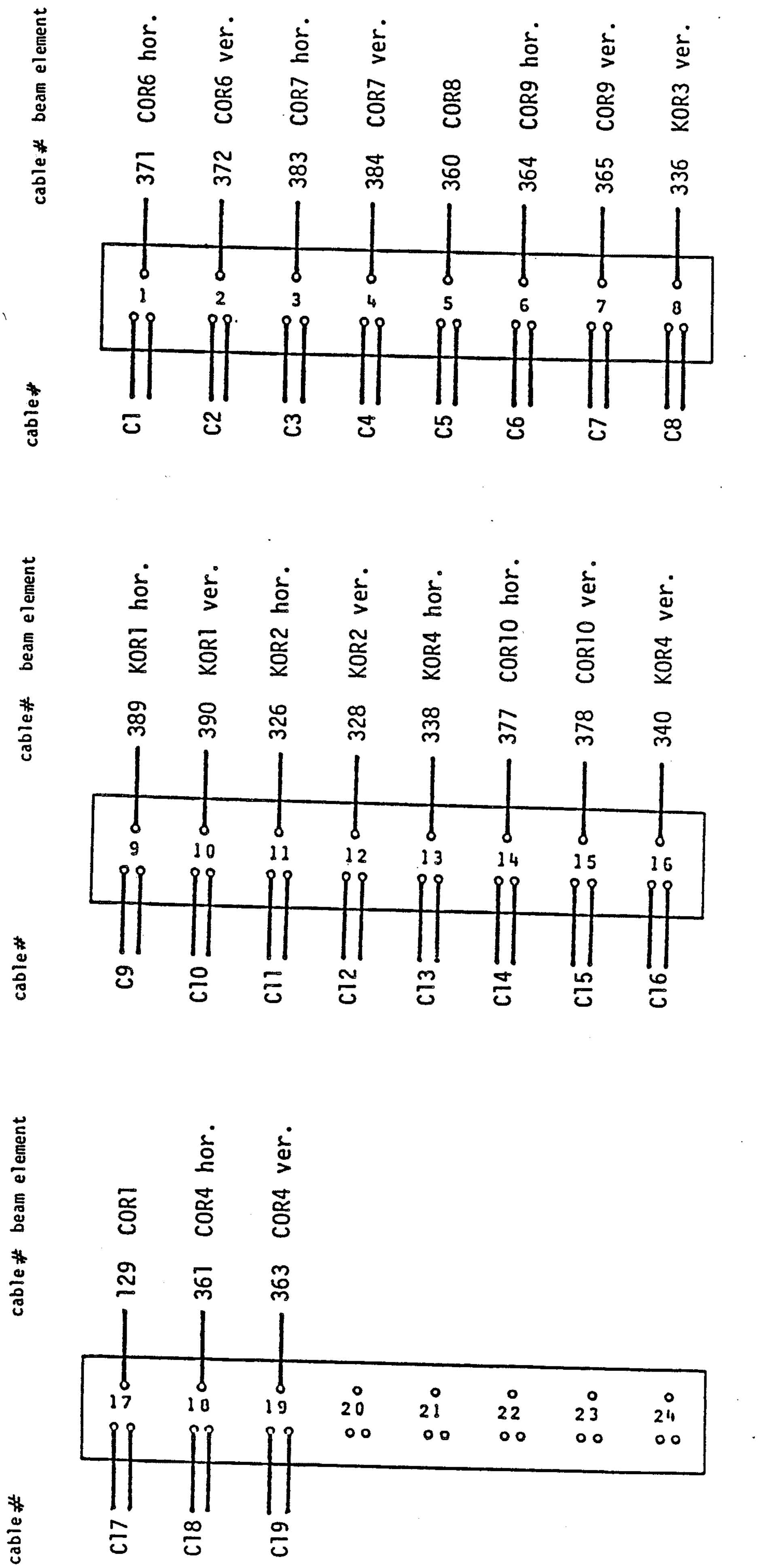


Fig. 8 Cable connections to distribution boxes for correction plates

Appendix B

BASIC FOR THE IEEE BUS

THE IBM SOFTWARE FOR INTERPRETATIVE BASIC

The file "BIB.M" supports the assembler for IEEE routines. In the GWBASIC these routines are executed using the "CALL" command. One example is:

```
CALL IBWRT(ad%,wrt$)
```

The CALL arguments have to be variables, not constants (numbers or text within quotation marks).

On the next pages follow first a BASIC function list of all the CALL commands that operate with the IEEE bus, and then comes special descriptions of the commands that I used in my BASIC programs (see e.g. appendix C). They are in order of apperance:

IBSIC
IBCLR
IBCMD
IBCMDA
IBFIND
IBRD
IBRDA
IBWRT

The lists here originate from the manual "IBM Personal Computer General Purpose Interface Bus Adapter Programming Support" (see the literature list).

BASIC Function Syntax

Function Syntax	Description
IBBNA (*ad*%, *adname$*)	Change access adapter of device
IBCAC (*ad*%, *v*%)	Become Active Controller
IBCLR (*ad*%)	Clear specified device
IBCMD (*ad*%, *cmd$*)	Send commands
IBCMDA (*ad*%, *cmd$*)	Send commands asynchronously
IBDMA (*ad*%, *v*%)	Enable/disable DMA
IBEOS (*ad*%, *v*%)	Change/disable EOS method
IBEOT (*ad*%, *v*%)	Enable/disable END message
IBFIND (*adname$*, *ad*%)	Return unit descriptor
IBGTS (*ad*%, *v*%)	Active Controller Go to Standby
IBIST (*ad*%, *v*%)	Set/clear individual status bit
IBLOC (*ad*%)	Go to Local
IBONL (*ad*%, *v*%)	Place adapter or device online or offline
IBPAD (*ad*%, *v*%)	Change Primary Address
IBPCT (*ad*%)	Pass Control
IBPPC (*ad*%, *v*%)	Parallel Poll Configure
IBRD (*ad*%, *rd$*)	Read data
IBRDA (*ad*%, *rd$*)	Read data asynchronously
IBRDF (*ad*%, *file$*)	Read data into file
IBRPP (*ad*%, *ppr*%)	Conduct a Parallel Poll
IBRSC (*ad*%, *v*%)	Request/release System Control
IBRSP (*ad*%, *spr*%)	Request Serial Poll
IBRSV (*ad*%, *v*%)	Request Service
IBSAD (*ad*%, *v*%)	Change Secondary Address
IBSIC (*ad*%)	Send Interface Clear
IBSRE (*ad*%, *v*%)	Set/clear 'remote enable' line
IBSTOP (*ad*%)	Stop asynchronous operation
IBTRG (*ad*%)	Trigger selected device
IBTMO (*ad*%, *v*%)	Change/disable timeout limit
IBWAIT (*ad*%, *mask*%)	Wait for selected event
IBWRT (*ad*%, *wrt$*)	Write data
IBWRTA (*ad*%, *wrt$*)	Write data asynchronously
IBWRTF (*ad*%, *file$*)	Write data from file

BASIC

Identifying the Adapter or Device

The first argument of all function calls except IBFIND is an integer variable, here denoted by *ad*%, which specifies the GPIB Adapter interface being used or the GPIB device being accessed. This integer variable is assigned a value using the IBFIND function. The IBFIND returned value (here assigned to *ad*%) is also referred to as the "unit descriptor" for the device or adapter.

IBSIC
Send Interface Clear

Adapter Input/Output

BASIC

Purpose: Sends interface clear.

Format: CALL IBSIC (*ad*%)

ad% specifies an interface adapter.

Example: To initialize the GPIB and become Controller-In-Charge at the beginning of a program:

```
100 CALL IBSIC (GPIB0%)
```

Remarks: The IBSIC function causes the GPIB Adapter to assert the IFC signal if, the GPIB Adapter has System Controller capability. This action initializes the GPIB and makes the GPIB Adapter the Controller-In-Charge.

The IFC signal is supposed to reset only the GPIB interface functions of bus devices and not to reset internal device functions. Device functions are reset with the Device Clear (DCL) and Selected Device Clear (SDC) commands. To determine the effect of these messages, consult the individual device documentation.

The ESAC error occurs if the GPIB Adapter does not have System Controller capability.

For more information, see IBRSC.

IBCLR
Clear Device

Device Management

BASIC

Purpose: Clears a device.

Format: CALL IBCLR (*ad*%)

ad% specifies a device.

Example: To clear device VMTR%:

```
100 CALL IBCLR (VMTR%)
```

Remarks: The IBCLR function clears the internal or device functions of a specified device. On exit, all devices are unaddressed.

IBCLR calls the adapter level IBCMD function to send the following commands using the designated access adapter:

- Listen address of the device
- Secondary address of the device if applicable
- Selected Device Clear (SDC)
- Untalk and Unlisten.

See IBCMD for additional information.

IBCMD
Send Commands

Adapter Input/Output

Purpose: Sends GPIB commands.

Format: CALL IBCMD (*ad%, cmd$*)

ad% specifies an interface adapter.

cmd$ contains the commands to be sent over the GPIB.

Examples: In these examples, GPIB commands and addresses are coded as printable ASCII characters. When the hex values to be sent over the GPIB correspond to printable ASCII characters, this is the simplest means of specifying the values. See the tables in Appendix C for conversions of hex values to ASCII characters.

To unaddress all Listeners and address a Talker at &H46 (ASCII F character) and a Listener at &H31 (ASCII 1 character):

```
110 CMD$ = ''?F1'' ' UNL TAD1 LAD2
120 CALL IBCMD (GPIB0%, CMD$)
```

Same as first example, except the Listener has a secondary address of &H6E (ASCII n character):

```
110 CMD$ = ''?F1n'' ' UNL TAD1 LAD2 SAD2
120 CALL IBCMD (GPIB0%, CMD$)
```


To clear all GPIB devices (reset internal functions) with the Device Clear command:

```
110 CMD$ = CHR$ (&H14) ' DCL
120 CALL IBCMD (GPIB0%, CMD$)
```

To clear two devices with Listener addresses of &H21 and &H28 with the Selected Device Clear command:

```
100 CMD$=''!(''+CHR$(&H04) ' LAD1 LAD2 SDC
110 CALL IBCMD (GPIB0%, CMD$)
```

To trigger any devices previously addressed to listen with the Group Execute Trigger command:

```
110 CMD$ = CHR$(&H08) ' GET
120 CALL IBCMD (GPIB0%, CMD$)
```

To unaddress all devices and serially poll a device at Talk address &H52 using the Serial Poll Enable and Serial Poll Disable commands (the GPIB Adapter listen address is &H20 or ASCII blank):

```
100 CMD$=''?R '' + CHR$(&H18) ' UNL TAD MLA SPE
110 CALL IBCMD (GPIB0%, CMD$)
120 RD$ = '' '' ' declare rd buffer of length 1
130 CALL IBRD (GPIB0%, RD$)
140 REM After checking the status byte in RD$,
150 REM disable this
160 REM device before polling the next one.
170 CMD$ = CHR$(&H19) + ''_'' ' SPD UNT
180 CALL IBCMD (GPIB0%, CMD$)
```

Remarks: The IBCMD function transmits interface messages (commands) over the GPIB. These commands, which are listed in Appendix C, include device talk and listen addresses, secondary addresses, serial and parallel poll configuration messages, and device clear and trigger instructions.

The IBCMD function also passes GPIB control to another device. This function is *not* used to transmit programming instructions to devices. Programming instructions and other device dependent information are transmitted with the write function.

The IBCMD operation terminates on any of the following events:

- All commands are successfully transferred.
- Error is detected.
- Timeout limit is exceeded.
- Take Control (TCT) command is sent.
- Interface Clear (IFC) message is received from the System Controller (*not* the GPIB Adapter).

After termination, the IBCNT% variable contains the number of commands sent. A short count can occur on any event but the first.

An ECIC error results if the GPIB Adapter is not Controller-In-Charge. If it is not Active Controller, it takes control and asserts ATN prior to sending the command bytes. It remains Active Controller afterward. The ENOL error does *not* occur if there are no Listeners.

IBCMDA
Send Commands Asynchronously

Adapter Input/Output

Purpose: Sends commands asynchronously.

Format: CALL IBCMDA (*ad%, cmd$*)

ad% specifies an interface adapter.

cmd$ contains the commands to be sent over the GPIB.

Example: To address several devices for a broadcast message to follow while testing for a high priority event to occur:

```
100 REM GPIB0% at talk address
105 REM &H40 (ASCII @)
110 REM addresses nine Listeners at addresses
120 REM &H31-&H39 (ASCII 1-9) to receive the
130 REM broadcast message
150 CMD$=''?@123456789'' 'UNL MTA LAD1...LAD9
160 CALL IBCMDA (GPIB0%, CMD$)
170 CALL EVENTTST 'Unspecified routine to test
180 REM     and process a high priority
190 REM     event
200 MASK% = &H4100 ' Wait for timeout or I/O
210 REM     completion.
220 CALL IBWAIT (GPIB0%, MASK%)
230 REM Loop until complete
240 IF (IBSTA% AND &H100) = 0 GOTO 170
```

IBCMDA
Send Commands Asynchronously

Remarks: The IBCMDA function transmits interface commands over the GPIB. These commands, which are listed in Appendix C, include device talk and listen addresses, secondary addresses, serial and parallel poll configuration messages, and device clear and trigger instructions.

The IBCMDA function also passes GPIB control to another device. This function is *not* used to transmit programming instructions to devices; programming instructions and other device dependent information are transmitted with the write function.

IBCMDA is used in place of IBCMD when the application program must perform other functions while processing the GPIB I/O operation. IBCMDA returns after starting the I/O operation. If the number of bytes to send is small and they are accepted quickly by the GPIB device(s), the operation can complete on the initial call.

If this is the case, the IBSTA% status bit is set. If it is not set, one or more calls to IBWAIT are needed to advance the I/O to completion (until CMPL is set to IBSTA%). When the wait returns, indicating that the I/O is complete, the number of commands sent is reported in the IBCNT% variable.

The asynchronous I/O started by IBCMDA terminates for the same reasons IBCMD terminates.

Between the issuing of the IBCMDA call and the corresponding wait for complete, other GPIB function calls to this adapter return the error ESYN, with the following exceptions:

IBSTOP To cancel the asynchronous I/O

IBWAIT To monitor other GPIB conditions

IBONL To cancel the I/O and reset the interface.

An ECIC error results if the GPIB Adapter is not Controller-In-Charge. If it is not Active Controller, the GPIB Adapter takes control and asserts ATN prior to sending the command bytes. It remains Active Controller afterwards. The ENOL error does *not* occur if there are no Listeners.

IBFIND
Return Unit Descriptor

Remarks: IBFIND returns a number that is used in each function call to identify the particular adapter or device that is to be used or is the object of that function. Calling IBFIND is required to associate a variable name in the application program with a particular device named at configuration time.

The name used in the adname$ argument must match exactly the device name entered at configuration time. The number returned must be assigned to a variable that is used in all references to that adapter or device in GPIB function calls.

The variable name selected should suggest the actual name of the device and the symbolic name given at configuration time in order to ease programming effort. The unit descriptor is valid to use in function calls until the offline function (IBONL with a parameter of 0) is called.

If the IBFIND call fails, a negative number is returned in place of the unit descriptor. The most probable reason for a failure is that the string argument passed to IBFIND does not match the device or adapter name assigned at configuration time. Device and adapter names can consist of letters and digits. The first character must be a letter. As with other DOS device names, a colon can optionally follow the name.

DOS treats uppercase and lowercase letters as identical. The string "PLOTTER" is treated as though it is identical to the string "plotter."

IBFIND
Return Unit Descriptor

Device Input/Output

Adapter Input/Output

Purpose: Returns the unit descriptor associated with the given name.

Format: CALL IBFIND (*adname$, ad%*)

adname$ is a string containing a device or adapter name entered at configuration time.

ad% is the variable into which the unit descriptor is stored by IBFIND.

Examples: To assign the unit descriptor associated with the device name 'DVM'(Digital Voltmeter) to the variable DVM%:

```
100 ADNAME$ = ''DVM''
110 CALL IBFIND (ADNAME$, DVM%)
120 IF DVM% < 0 GOTO 1000 ' ERROR ROUTINE
```

To assign the unit descriptor associated with adapter 'GPIB0' to the variable GPIB0%:

```
100 ADNAME$ = ''GPIB0''
110 CALL IBFIND (ADNAME$, GPIB0%)
120 IF GPIB0% < 0 GOTO 1000 ' ERROR ROUTINE
```

Device Input/Output

Adapter Input/Output

Purpose: Reads data.

Format: CALL IBRD (*ad*%, *rd$*)

ad% specifies a device or an interface adapter.

rd$ identifies the storage buffer for data bytes that are read from the GPIB.

Examples: To read 56 bytes from a device called TAPE%:

```
110 REM Perform device level read
120 RD$ = SPACE$(56)
130 CALL IBRD (TAPE%, RD$)
140 REM Check IBSTA% to see how the
145 REM read terminated: on CMPL,
150 REM END, TIMO, or ERR.
155 REM Data is stored in RD$.
160 REM All unaddressing has been done.
```

To read 56 bytes of data from a device at talk address &H4C (ASCII L) and then unaddress it (the GPIB Adapter listen address is &H20 or ASCII blank):


```
100 CMD$ = ''? L'' ' UNL MLA TAD
110 CALL IBCMD (GPIB0%, CMD$)
120 RD$ = SPACE$(56)
130 CALL IBRD (GPIB0%, RD$)
140 REM check IBSTA% to see how the
145 REM read terminated: on CMPL,
150 REM END, TIMO, or ERR.
160 REM Data is stored in RD$.
170 REM Unaddress the Talker.
180 CMD$ = ''_'' ' UNT
190 CALL IBCMD (GPIB0%, CMD$)
```

Remarks: The IBRD function reads from 1 to 255 bytes of data from a GPIB device.

To terminate the read on an End-of-String character, see IBEOS examples.

Device IBRD Function:

When *ad*%specifies a device, the following adapter level steps are performed automatically to read from the device:

1. The IBCMD function is called to address the device to talk and the access adapter to listen.

2. The adapter level IBRD function is called to read the data from the device, as explained below.

3. The IBCMD function is called to unaddress the access adapter and all devices using the Unlisten and Untalk commands.

When the device level IBRD function returns:

- IBSTA% holds the latest device status
- IBCNT% is the actual number of data bytes read from the device
- IBERR% is the first error detected, if the ERR bit in IBSTA% is set.

Adapter IBRD Function:

When *ad*% specifies an interface adapter, the IBRD function attempts to read from a GPIB device that is assumed to be already properly initialized and addressed.

If the access adapter is Controller-In-Charge, the IBCMD function must be called prior to IBRD to address a device to talk and the adapter to listen. Otherwise, the device on the GPIB that is the Controller-In-Charge must perform the addressing.

If the access adapter is Active Controller, the adapter is first placed in Standby Controller state with ATN off and remains there after the read operation is completed. Otherwise, the read operation starts immediately.

An EADR error results if the adapter is Controller-In-Charge but has not been addressed to listen with the IBCMD function. An EABO error results if the adapter is not the Controller-In-Charge and is not addressed to listen within the timeout limit. An EABO error also results if the device that is to talk is not addressed and/or the operation does not complete for whatever reason within the timeout limit.

IBRD
Read Data

The adapter level IBRD operation terminates on any of the following events:

- Allocated buffer becomes full
- Error is detected
- Timeout limit is exceeded
- END message is detected
- EOS character is detected (if this option is enabled)
- Device Clear (DCL) or Selected Device Clear (SDC) command is received from another device which is the Controller-In-Charge.

After termination, IBCNT% contains the number of bytes read. A shorter count than expected can occur on any even but the first.

IBRDA
Read Data Asynchronously

Advanced Device Input/Output

Adapter Input/Output

Purpose: Reads data asynchronously.

Format: CALL IBRDA (*ad%*, *rd$*)

ad% specifies a device or an interface adapter.

rd$ identifies the storage buffer for data bytes that are read from the GPIB.

Examples: To read 62 bytes of data from a device called TAPE%:

```
110 REM Perform device level read
120 RD$ = SPACE$(62)
130 CALL IBRDA (TAPE%, RD$)
135 REM Perform other processing here
136 REM then wait for I/O completion
137 REM or a timeout
140 MASK% = &H4100 ' TIMO CMPL
145 CALL IBWAIT (TAPE%, MASK%)
150 REM Check IBSTA% to see how the
160 REM read terminated: on CMPL,
170 REM END, TIMO, or ERR.
180 REM Data is stored in RD$.
190 REM All unaddressing
200 REM has been done.
```

To read 43 bytes of data from a device at talk address &H4E (ASCII N) and then unaddress it (the GPIB Adapter listen address is &H20 (ASCII blank)):

IBRDA
Read Data Asynchronously

```
80  REM Perform addressing in preparation
90  REM for adapter level read
100 CMD$ = ''? N'' ' UNL MLA TAD
110 CALL IBCMD (GPIB0%, CMD$)
115 REM Perform adapter level read
120 RD$ = SPACE$(43)
130 CALL IBRDA (GPIB0%, RD$)
135 REM Perform other processing here
136 REM then wait for I/O completion
137 REM or a timeout
140 MASK% = &H4100 ' TIMO CMPL
150 CALL IBWAIT (GPIB0%, MASK%)
160 REM check IBSTA% to see how the
170 REM read terminated: on CMPL,
180 REM END, TIMO, or ERR.
190 REM Data is stored in RD$.
195 REM Unaddress Talker.
200 CMD$ = ''_'' ' UNT
210 CALL IBCMD (GPIB0%, CMD$)
```

To terminate the read on an End-of-String character, see IBEOS examples.

Remarks: The IBRDA function reads from 1 to 255 bytes of data from a GPIB device. Device level and adapter level operations of the function are described below.

IBRDA is used in place of IBRD when the application program must perform other functions while processing the GPIB I/O operation. IBRDA returns after starting the I/O operation. If the number of bytes to read is small and they are transmitted quickly by the GPIB device, the operation can complete on the initial call. If this is the case, the IBSTA% status bit CMPL is set. If it is not set, one or more calls to IBWAIT are needed to advance the I/O to completion, that is, until CMPL is set in IBSTA%. When the wait returns, indicating that the I/O is complete, the number of commands read is reported in the IBCNT% variable.

IBRDA
Read Data Asynchronously

The asynchronous I/O started by IBRDA completes for the same reasons IBRD terminates.

Device IBRDA Function:

When *ad%* specifies a device, the following adapter level steps are performed automatically to read from the device:

1. The IBCMD function is called to address the device to talk and the access adapter to listen.
2. The adapter level IBRDA function is called to read the data from the device, as explained below.
3. When the device level IBRDA function returns, IBSTA% holds the latest device status; IBERR% is the first error detected, if the ERR bit in IBSTA% is set.

Adapter IBRDA Function:

When *ad%* specifies an interface adapter, the IBRDA function attempts to read from a GPIB device that is assumed to be already properly initialized and addressed.

If the access adapter is Controller-In-Charge, the IBCMD function must be called prior to IBRDA to address the device to talk and the adapter to listen. Otherwise, the device on the GPIB that is the Controller-In-Charge must perform the addressing.

If the access adapter is Active Controller, the adapter is first placed in Standby Controller state with ATN off and remains there after the read operation is completed. Otherwise, the read operation commences immediately.

IBRDA

Read Data Asynchronously

IBRDA returns immediately even when no error condition exists. A subsequent wait for CMPL must be done to conclude any asynchronous I/O operation in progress. When the wait returns, indicating that the I/O is complete, IBCNT% indicates the number of bytes received, and unaddressing is performed.

Between the issuing of the IBRDA call and the corresponding wait for complete, other GPIB function calls to this device or to any other device with the same access adapter, or any adapter level calls to the access adapter itself, returns the error ESYN, with the following exceptions:

IBSTOP To cancel the asynchronous I/O

IBWAIT To monitor other GPIB conditions

IBONL To cancel the I/O and reset the interface.

An EADR error results if the interface adapter is Controller-In-Charge but has not addressed itself as a Listener with the IBCMD function.

IBWRT

Write Data

Device Input/Output

Adapter Input/Output

Purpose: Writes data.

Format: CALL IBWRT (*ad%*, *wrt$*)

ad% specifies an interface adapter or a device.

wrt$ contains the data to be sent over the GPIB.

Examples: To write 10 instruction bytes to a device called DVM%:

```
120 WRT$ = ''F3R1X5P2G0''
130 CALL IBWRT (DVM%, WRT$)
```

To write 10 instruction bytes to a device at listen address &H2F (ASCII /) and then unaddress it (the GPIB Adapter talk address is &H40 or ASCII @):

```
95  REM Perform addressing
100 CMD$ = ''?@/'' ' UNL MTA LAD
110 CALL IBCMD (GPIB0%, CMD$)
115 REM Perform adapter level write
120 WRT$ = ''F3R1X5P2G0''
130 CALL IBWRT (GPIB0%, WRT$)
135 REM Unaddress all listeners
140 CMD$ = ''?'' ' UNL
150 CALL IBCMD (GPIB0%, CMD$)
```

Remarks: The IBWRT function writes from 1 to 255 bytes of data to a GPIB device.

IBWRT

Write Data

Device IBWRT Function:

When *ad%* specifies a device, the following adapter level steps are performed automatically to write to the device:

1. The IBCMD function is called to address the device to listen and the access adapter to talk.
2. The adapter level IBWRT function is called to write the data to the device, as explained below.
3. The IBCMD function is called to unaddress the access adapter using the Untalk command and all devices using and Unlisten command.

When the device level IBWRT function returns, IBSTA% holds the latest device status; IBCNT% is the actual number of data bytes written to the device; and IBERR% is the first error detected if the ERR bit in IBSTA% is set.

Adapter IBWRT Function:

When *ad%* specifies an interface adapter, the IBWRT function attempts to write to a GPIB device that is assumed to be already properly initialized and addressed.

If the access adapter is Controller-In-Charge, the IBCMD function must be called prior to IBWRT to address the device to listen and the adapter to talk. Otherwise, the device on the GPIB that is the Controller-In-Charge must perform the addressing.

IBWRT

Write Data

If the access adapter is Active Controller, the adapter is first placed in Standby Controller state with ATN off and remains there after the write operation is completed. Otherwise, the write operation commences immediately.

An EADR error results if the adapter is Controller-In-Charge but has not been addressed to talk with the IBCMD function. An EABO error results if the adapter is not the Controller-In-Charge and is not addressed to talk within the timeout limit. An EABO error also results if the operation does not complete for whatever reason within the timeout limit.

The adapter level IBWRT operation terminates on any of the following events:

- All bytes are transferred.
- Error is detected.
- Timeout limit is exceeded.
- Device Clear (DCL) or Selected Device Clear (SDC) command is received from another device which is the Controller-In-Charge.

After termination, IBCNT% contains the number of bytes written. A shorter count than expected can occur on any event but the first.

Appendix C

CODE LISTINGS

THE PROGRAMS WRITTEN IN GWBASIC ON THE OLIVETTI

The following pages cover the code for the programs written in GWBASIC on the Olivetti. They are in order of appearance:

VOLTBIT
START
HVSET
SCANNER

```
 REM
20 REM        V O L T B I T
30 REM        by Petter Broman Summer Student 1989
40 REM
50 COLOR 2.1.1:CLS
60        CLEAR   ,57611!
70        IBINIT1 = 57611!
80        IBINIT2 = IBINIT1 + 3
90        BLOAD "\BIB.M",IBINIT1
100        CALL IBINIT1(IBFIND,IBSTOP.IBTRG,IBCLR.IBPCT,IBSIC.IBLOC.IBPPC.IBBNA,I
BONL,IBRSC,IBSRE,IBRSV,IBPAD,IBSAD,IBIST,IBDMA,IBEOS,IBTMO.IBEOT)
110        CALL IBINIT2(IBGTS,IBCAC,IBWAIT,IBPOKE,IBWRTF,IBWRTA,IBWRT.IBCMDA.IBCM
D,IBRDF,IBRDA,IBRD,IBRPP,IBRSP,IBDIAG,IBXTRC,IBSTA%,IBERR%,IBCNT%)
120       CMD$ = SPACE$(10)        ' command buffer
130       RD$ = SPACE$(255)        ' read buffer
140       WRT$ = SPACE$(255)       ' write buffer
150       BDNAME$ = SPACE$(7)      ' board/device name
160 BDNAME$="GPIBO":CALL IBFIND(BDNAME$.GPIBO%)
170 BDNAME$="DEVO":CALL IBFIND(BDNAME$.DEVO%)
180 CALL IBSIC(GPIBO%)
190 WRT$=CHR$(&H23)
200 CALL IBWRT(DEVO%,WRT$)          'CLEAR AND INIT CAMAC CONTROLLER!
210 READ V$:IF V$="END" THEN 440
220 GOSUB 350
230 VOLT=10*VALUE/4095
240 WRT$=CHR$(16)+CHR$(0)+CHR$(20)+LB$+MB$
250 CALL IBWRT(DEVO%,WRT$):PRINT V$,VALUE,VOLT
260 RD$=SPACE$(1)
270 CALL IBRD(DEVO%,RD$)
280 PRINT "READ VOLTAGE IN CH1 AND PRESS ANY KEY TO CONTINUE!"
290 IF LEN(INKEY$)=0 THEN 290
300 GOTO 210
310 DATA "000","001","002","004","008","010","020","040","080","100"
320 DATA "200","400","800"
330 DATA "END"
340 REM
350 REM HEX TO ASCBYTE:
360 REM
370 B1%=ASC(MID$(V$,3,1))-48:B2%=ASC(MID$(V$,2,1))-48:B3%=ASC(MID$(V$,1,1))-48
380 IF B1%>9 THEN B1%=B1%-7
390 IF B2%>9 THEN B2%=B2%-7
400 IF B3%>9 THEN B3%=B3%-7
410 VALUE=B3%*256+B2%*16+B1%
420 LB$=CHR$(B2%*16+B1%):MB$=CHR$(B3%)
430 RETURN
440 END
```

```
10 '
20 ' START
30 ' made bv Petter Broman  Summer Student 1989"
40 '
50 DIM Q$(10),PGM$(10)
60 Q$(1)="MASS SCAN          ":PGM$(1)="SCANNER.BAS"
70 Q$(2)="BEAM LINE CONTROL":PGM$(2)="HVSET.BAS"
80 Q$(3)="EXIT               ":PGM$(3)=""
90 SCREEN 0,0,0:WIDTH 80:KEY OFF:CLS:COLOR 3,0,14
100 LOCATE  2,14:PRINT "WELCOME TO THE SYSTEM FOR MAGNET CONTROL OF ISOLDE 2"
110 LOCATE  9,32:PRINT "*********************"
120 LOCATE 10,32:PRINT "*                   *"
130 LOCATE 11,32:PRINT "*                   *"
140 LOCATE 12,32:PRINT "*                   *"
150 LOCATE 13,32:PRINT "*********************"
160 ROW%=10
170 FOR R%=10 TO 12:LOCATE R%,34:PRINT Q$(R%-9):NEXT R%
180 COLOR ,4:LOCATE ROW%,34:PRINT Q$(ROW%-9)
190 I$=INKEY$:IF I$="" THEN 190
200 IF I$=CHR$(0)+"H" THEN NEWROW%=ROW%-1
210 IF I$=CHR$(0)+"P" THEN NEWROW%=ROW%+1
220 IF I$=CHR$(13) THEN 300
230 I$=INKEY$:IF I$<>"" THEN 230
240 IF NEWROW%<10 THEN NEWROW%=10
250 IF NEWROW%>12 THEN NEWROW%=12
260 IF NEWROW%=ROW% THEN 190
270 COLOR ,0:LOCATE ROW%,34:PRINT Q$(ROW%-9)
280 ROW%=NEWROW%
290 GOTO 180
300 IF ROW%=12 THEN SCREEN 0,0,0:END
310 RUN PGM$(ROW%-9)
```

```
10 REM
20 REM  H V  S E T     bv Petter Broman 1989
30 REM
40        CLEAR   ,57611!
50        IBINIT1 = 57611!
60        IBINIT2 = IBINIT1 + 3
70        BLOAD "BIB.M",IBINIT1
80        CALL IBINIT1(IBFIND,IBSTOP,IBTRG,IBCLR,IBPCT,IBSIC,IBLOC,IBPPC,IBBNA,IB
ONL,IBRSC,IBSRE,IBRSV,IBPAD,IBSAD,IBIST,IBDMA,IBEOS,IBTMO,IBEOT)
90        CALL IBINIT2(IBGTS,IBCAC,IBWAIT,IBPOKE,IBWRTF,IBWRTA,IBWRT,IBCMDA,IBCMD
,IBRDF,IBRDA,IBRD,IBRPP,IBRSP,IBDIAG,IBXTRC,IBSTA%,IBERR%,IBCNT%)
100       CMD$ = SPACE$(10)         ' commanPRINT SPACE$(40)
110       RD$ = SPACE$(255)         ' read buffer
120       WRT$ = SPACE$(255)        ' write buffer
130       BDNAME$ = SPACE$(7)       ' board/device name
140 BDNAME$="GPIB0":CALL IBFIND(BDNAME$,GPIB0%)
150 BDNAME$="DEV0":CALL IBFIND(BDNAME$,CAMAC%)
160 N=23
170 GOTO 830
180 '
190 'COMMAND
200 '
210 WRT$=CHR$(16)+CHR$(0)+CHR$(N)+CHR$(M1)+CHR$(M2)+CHR$(0)
220 RD$=SPACE$(1)
230 CALL IBWRT(CAMAC%,WRT$)
240 CALL IBRD(CAMAC%,RD$)
250 RETURN
260 '
270 'RESPONSE
280 '
290 WRT$=CHR$(2)+CHR$(0)+CHR$(N)
300 RD$=SPACE$(3)
310 CALL IBWRT(CAMAC%,WRT$)
320 CALL IBRD(CAMAC%,RD$)
330 W1=ASC(MID$(RD$,1,1))
340 W2=ASC(MID$(RD$,2,1))
350 W3=ASC(MID$(RD$,3,1))
360 RETURN
370 '
380 'CAMAC FUNCTION
390 '
400 WRT$=CHR$(F)+CHR$(A)+CHR$(N)
410 RD$=SPACE$(1)
420 CALL IBWRT(CAMAC%,WRT$)
430 CALL IBRD(CAMAC%,RD$)
440 RETURN
450 '
460 F=0:A=0:GOSUB 380:L2=ASC(RD$) AND 2:RETURN  'LAM2
470 F=10:A=1:GOSUB 380:RETURN                   'CLEAR LAM2
480 F=9:A=1:GOSUB 380:RETURN                    'CLEAR INPUT BUFFER
490 '
500 'SET ONE CHANNEL OR PRESET ALL CHANNELS (TG=0 OR 7)
510 '
520 M1=TG+(MF AND 15)*16:M2=MF\16+CH*4
530 GOSUB 190           'COMMAND
540 M1=(D AND 15)*16+1:M2=D\16
550 GOSUB 190           'COMMAND (TG=1)
560 GOSUB 460           'LAM2
570 IF L2<>2 THEN 560
580 GOSUB 470           'CLEAR LAM2
590 RETURN
600 '
610 'SET HV ON\OFF (SW=4 OR 0)
620 '
630 M1=5+(MF AND 15)*16:M2=MF\16+SW
640 GOSUB 190           'COMMAND (TG=5)
650 GOSUB 460           'LAM2
660 IF L2<>2 THEN 560
670 GOSUB 470           'CLEAR LAM2
680 RETURN
690 '
700 'READ TRUE/SET VOLTAGE OR STATUS (TG=3,8 OR 9)
710 '
720 M1=TG+(MF AND 15)*16:M2=MF\16+CH*4
730 GOSUB 480           'CLEAR INPUT BUFFER
740 GOSUB 190           'COMMAND
750 GOSUB 460           'LAM2
760 IF L2<>2 THEN 750
770 GOSUB 270           'RESPONSE
780 GOSUB 470           'CLEAR LAM2
790 IF TG=3 OR TG=8 THEN V=W2*16+W1\16 ELSE SW=W2 AND 4
800 RETURN
810 '
820 '
830 QUIT%=0:MX=33
840 DIM N$(MX),MF%(MX),CHP%(MX),CHN%(MX)           'BEAM LINE DATA
850 DIM USEDMF%(16)
860 DIM VALUE%(MX),MIDVALUE%(MX),SPAN%(MX)         'SET VALUES
870 DIM VALUE2%(MX),MIDVALUE2%(MX),SPAN2%(MX)      'STO COPY
880 DIM MENU$(100),BEAMFILE$(100)                  'MENU DATA
890 UP$=CHR$(0)+"H":DO$=CHR$(0)+"P"
900 LE$=CHR$(0)+"K":RI$=CHR$(0)+"M"
910 DTOP%=1
920 DIM CUR%(20),MARK%(15)                         'CURSERS GRAPHIC DATA
930 DIM UNDERCUR%(20),UNDERMARK%(15)
940 DIM TEMP%(15,MX),TEMPX(MX),INV%(300)
950 DEF FNPUTX(VALUE,SPAN)=VALUE*300/SPAN+158-1
960 DEF FNPUTY(NR)=8+NR*16
970 GOSUB 1270                               'SCREEN 1 GRAPHICS
980 START%=1:GOSUB 3330                      'STARTING BEAM LINE
990 IF QUIT% THEN SCREEN 0,0,0:WIDTH 80:RUN "START"
1000 GET (X%,Y%)-STEP(7,7),UNDERCUR%
1010 PUT (X%,Y%),CUR%,OR
1020 '
1030 ' HEAD ROUTINE
1040 '
1050 I$=INKEY$:IF I$="" THEN 1050
1060 XO%=X%:YO%=Y%
1070 IF I$="Q" OR I$="q" THEN GOSUB 2420
1080 IF I$=CHR$(13) THEN GOSUB 2260
1090 IF I$=CHR$(3) THEN DTOP%=DTOP%+1:GOSUB 1450
1100 IF I$=CHR$(27) THEN DTOP%=DTOP%-1:GOSUB 1450
1110 IF QUIT% THEN SCREEN 0,0,0:WIDTH 80:RUN "START"
1120 IF I$=UP$ THEN Y%=Y%-1
1130 IF I$=DO$ THEN Y%=Y%+1
1140 IF I$=LE$ THEN X%=X%-1
1150 IF I$=RI$ THEN X%=X%+1
1160 I$=INKEY$:IF I$<>"" THEN 1120
1170 IF X%>XMAX% THEN X%=XMAX%
1180 IF X%<0 THEN X%=0
1190 IF Y%>=YMAX% THEN Y%=YMAX%
1200 IF Y%<=0 THEN Y%=0
1210 PUT (XO%,YO%),UNDERCUR%,PSET
1220 GET (X%,Y%)-STEP(7,7),UNDERCUR%
1230 PUT (X%,Y%),CUR%,OR
1240 I$=INKEY$:IF I$<>"" THEN 1240
1250 GOTO 1050
```

```
1260 '
1270 ' SCREEN 1 GRAPHICS INITIALISATION
1280 '
1290 XMAX%=312:YMAX%=192:X%=160:Y%=96
1300 SCREEN 1:COLOR 8,0,3
1310 PALETTE 1,10
1320 PALETTE 2,9
1330 PALETTE 3,4
1340 LOCATE 1,1:PRINT SPACE$(1)   'BAR POSITIONER
1350 LINE (0,0)-(0,6):LINE (2,0)-(2,6)
1360 GET (0,0)-(4,6),MARK%
1370 LOCATE 1,1:PRINT ""          'CURSOR
1380 GET (0,0)-(7,7),CUR%
1390 LOCATE 1,1:PRINT STRING$(16,219)    'INVERS MENU MARKER
1400 GET (0,0)-(128,7),INV%
1410 KEY OFF
1420 CLS
1430 RETURN
1440 '
1450 ' INITIAL SCREEN
1460 '
1470 CLS
1480 C%=3:COLOR ,,C%,C%,C%
1490 TEXT$=BEAM$+"          "+SELECTFILE$
1500 LOCATE 1,20-LEN(TEXT$)\2:PRINT TEXT$
1510 C%=1:COLOR ,,C%,C%,C%
1520 LOCATE 2,2:PRINT "QUIT RCL STO LOAD SAVE BEAM HT"
1530 LOCATE 2,33
1540 C%=3:COLOR ,,C%,C%,C%
1550 IF SW=-1 THEN PRINT "BOTH"
1560 IF SW=0 THEN PRINT "OFF "
1570 IF SW=4 THEN PRINT " ON "
1580 C%=2:COLOR ,,C%,C%,C%
1590 RESTORE 1910
1600 FOR J%=1 TO 7
1610 READ TX,TY,BX,BY
1620 LINE (TX,TY)-(BX,BY),C%,B
1630 NEXT J%
1640 IF DTOP%<1 THEN DTOP%=1
1650 IF DTOP%>NR% THEN DTOP%=NR%
1660 FOR DNR%=DTOP% TO DTOP%+10
1670 C%=1+(DNR% MOD 2):COLOR ,,C%,C%,C%:
1680 ROW%=2*(DNR%-DTOP%)+3:COL%=2
1690 IF DNR%>NR% THEN 1870
1700 LOCATE ROW%,COL%
1710 PRINT N$(DNR%);TAB(COL%+9);"VALUE       SPAN      + - RESET";
1720 LOCATE ROW%,COL%+14:PRINT USING "#####";VALUE%(DNR%);
1730 LOCATE ROW%,COL%+24:PRINT USING "####";SPAN%(DNR%);
1740 RESTORE 1980
1750 DX=(COL%-2)*8:DY=(ROW%-3)*8
1760 FOR J%=1 TO 4
1770 READ TX,TY,BX,BY
1780 TX=TX+DX:TY=TY+DY:BX=BX+DX:BY=BY+DY
1790 IF J%<4 THEN LINE (TX,TY)-(BX,BY),C%,B
1800 IF J%=4 THEN LINE (TX,TY)-(BX,BY),C%,BF
1810 NEXT J%
1820 FOR J%=1 TO 9
1830 LINE (TX+30*J%,TY)-STEP(0,2-(J%=5)),0
1840 NEXT J%
1850 PUTX=FNPUTX(VALUE%(DNR%)-MIDVALUE%(DNR%),SPAN%(DNR%))
1860 GOSUB 2090:GOSUB 2170
1870 NEXT DNR%
1880 GET (X0%,Y0%)-STEP(7,7),UNDERCUR%
1890 C%=1:COLOR ,,C%,C%,C%
1900 RETURN
1910 DATA 7,8,39,15:       'QUIT
1920 DATA 47,8,71,15:      'RCL
1930 DATA 79,8,103,15:     'STO
1940 DATA 111,8,143,15:    'LOAD
1950 DATA 151,8,183,15:    'SAVE
1960 DATA 191,8,224,15:    'BEAM
1970 DATA 231,8,246,15:    'HT
1980 DATA 240,16,246,22:   '+
1990 DATA 256,16,262,22:   '-
2000 DATA 272,16,310,22:   'RESET
2010 DATA 8.24.308.30:     'BAR
2020 '
2030 ' RESTORE GRAPHICS UNDER THE OLD BAR POSITIONER
2040 '
2050 FOR K%=1 TO 15:UNDERMARK%(K%)=TEMP%(K%,DNR%):NEXT K%
2060 PUT (TEMPX(DNR%),FNPUTY(DNR%-DTOP%+1)),UNDERMARK%,PSET
2070 RETURN
2080 '
2090 ' SAVE GRAPHICS UNDER THE NEW BAR POSITIONER
2100 '
2110 TEMPX(DNR%)=PUTX
2120 PUTY=FNPUTY(DNR%-DTOP%+1)
2130 GET (PUTX,PUTY)-STEP(4,6),UNDERMARK%
2140 FOR K%=1 TO 15:TEMP%(K%,DNR%)=UNDERMARK%(K%):NEXT K%
2150 RETURN
2160 '
2170 ' SET GRAPHICS FOR THE NEW BAR POSITIONER
2180 '
2190 PUT (PUTX,PUTY),MARK%,OR
2200 PUTY=FNPUTY(DNR%-DTOP%+1)
2210 GET (X0%,Y0%)-STEP(7,7),UNDERCUR%
2220 RETURN
2230 '
2240 ' CLICK
2250 '
2260 PUT (X0%,Y0%),UNDERCUR%,PSET
2270 XA=X%+3:YA=Y%+1
2280 IF YA<16 OR YA>190 THEN XB=XA:YB=YA:GOTO 2340
2290 DNR%=(YA-16)\16+DTOP%                    'DEVICE NUMBER
2300 C%=1+(DNR% MOD 2):COLOR ,,C%,C%,C%:
2310 IF DNR%>NR% THEN 2400
2320 XB=XA
2330 YB=16+(YA-16) MOD 16
2340 RESTORE 1910:BOX%=0
2350 FOR J%=1 TO 11
2360 READ TX,TY,BX,BY
2370 IF XB>=TX AND XB<=BX AND YB>=TY AND YB<=BY THEN BOX%=J%
2380 NEXT J%
2390 ON BOX% GOSUB 2420,2470,2590,2690,3140,3330,3760,3860,4010,4170,4250
2400 RETURN
2410 '
2420 'QUIT
2430 '
2440 QUIT%=1
2450 RETURN
2460 '
2470 'RCL
2480 '
2490 IF BEAM2$<>BEAM$ THEN BEEP:RETURN
2500 FOR I%=1 TO NR%
2510 SPAN%(I%)=SPAN2%(I%)
2520 MIDVALUE%(I%)=MIDVALUE2%(I%)
2530 IF VALUE%(I%)<>VALUE2%(I%) THEN DNR%=I%:GOSUB 4370
2540 VALUE%(I%)=VALUE2%(I%)
2550 NEXT I%
2560 GOSUB 1450
2570 RETURN
```

```
2580 '
2590 'STO
2600 '
2610 BEAM2$=BEAM$
2620 FOR I%=1 TO NR%
2630 VALUE2%(I%)=VALUE%(I%)
2640 MIDVALUE2%(I%)=MIDVALUE%(I%)
2650 SPAN2%(I%)=SPAN%(I%)
2660 NEXT I%
2670 RETURN
2680 '
2690 'LOAD
2700 '
2710 OPEN "\DATADIR.TXT" FOR INPUT AS #2
2720 FLAST%=0
2730 FOR L%=1 TO 4
2740 LINE INPUT #2,INLINE$
2750 NEXT L%
2760 LINE INPUT #2,INLINE$
2770 IF EOF(2) THEN 2850
2780 FOR F%=1 TO 5
2790 Q1$=MID$(INLINE$,1+13*(F%-1),13)
2800 IF LEFT$(Q1$,1)="." THEN 2830
2810 IF Q1$="" THEN 2830
2820 FLAST%=FLAST%+1:MID$(Q1$,9,1)=".":MENU$(FLAST%)=Q1$
2830 NEXT F%
2840 GOTO 2760
2850 CLOSE 2
2860 TEXT$="LOAD":GOSUB 4770
2870 IF SELECT$="CANCEL" THEN 3110
2880 OPEN "\DATA\"+SELECT$ FOR INPUT AS #1
2890 INPUT #1,BEAM3$
2900 IF BEAM$<>BEAM3$ THEN BEEP:GOTO 3110
2910 BEAM$=BEAM3$
2920 SELECTFILE$=SELECT$
2930 EXT$=MID$(SELECTFILE$,INSTR(SELECTFILE$,".")+1,3)
2940 INPUT #1,COMM1$
2950 INPUT #1.COMM2$
2960 INPUT #1,COMM3$
2970 INPUT #1,COMM4$
2980 CLS:TEXT$="LOAD":GOSUB 4510
2990 TG=7:CH=0:D=0                'IF HV OFF PRESET ALL CHANNELS
3000 FOR MF=10 TO 16
3010 IF USEDMF%(MF) THEN GOSUB 500
3020 NEXT MF
3030 FOR I%=1 TO NR%
3040 INPUT #1,N$(I%),VALUE%(I%),SPAN%(I%)
3050 MIDVALUE%(I%)=VALUE%(I%)
3060 DNR%=I%:GOSUB 4370          'SET HV VALUE
3070 NEXT I%
3080 CLOSE 1
3090 LOCATE 9,1:PRINT "PRESS RETURN OR MIDDLE BUTTON ON MOUSE"
3100 IF INKEY$<>CHR$(13) THEN 3100
3110 GOSUB 1450
3120 RETURN
3130 '
3140 'SAVE
3150 '
3160 TEXT$="SAVE":GOSUB 4500
3170 IF SELECT$="" THEN 3310
3180 SELECTFILE$=SELECT$
3190 OPEN "\DATA\"+SELECT$ FOR OUTPUT AS #1
3200 PRINT #1,BEAM$
3210 PRINT #1,COMM1$
3220 PRINT #1,COMM2$
3230 PRINT #1,COMM3$
3240 PRINT #1,COMM4$
3250 FOR I%=1 TO NR%
3260 PRINT #1,N$(I%)+",",MIDVALUE%(I%),SPAN%(I%)
3270 NEXT I%
3280 CLOSE 1
3290 SHELL "DIR \DATA\*."+EXT$+"/W >\DATADIR.TXT"
3300 GOSUB 1450
3310 RETURN
3320 '
3330 'BEAM
3340 '
3350 GOSUB 470
3360 OPEN "\BEAMDIR.TXT" FOR INPUT AS #2
3370 FLAST%=0
3380 FLAST%=FLAST%+1
3390 INPUT #2,MENU$(FLAST%),BEAMFILE$(FLAST%)
3400 IF EOF(2) THEN 3410 ELSE 3380
3410 CLOSE 2
3420 TEXT$="EXPERIMENT":GOSUB 4770
3430 IF SELECT$<>"CANCEL" THEN 3450
3440 IF START% THEN QUIT%=1:RETURN ELSE 3730
3450 OPEN "\BEAM\"+BEAMFILE$(F%) FOR INPUT AS #1
3460 EXT$=MID$(BEAMFILE$(F%),INSTR(BEAMFILE$(F%),".")+1,3)
3470 SHELL "DIR \DATA\*."+EXT$+"/W >\DATADIR.TXT"
3480 SELECTFILE$=""                       'FIRST NO DATA FILE
3490 INPUT #1,BEAM$,NR%                   'BEAM LINE & NUMBER OF DEVICES
3500 FOR MF=10 TO 16:USEDMF%(MF)=0:NEXT MF
3510 FOR I%=1 TO NR%
3520 INPUT #1,N$(I%),MF%(I%),CHP%(I%),CHN%(I%)
3530 USEDMF%(MF%(I%))=1
3540 NEXT I%
3550 CLOSE 1
3560 TG=9:CH=0
3570 SW=-99
3580 FOR MF=10 TO 16              'READ HV STATUS
3590 SWOLD=SW
3600 IF USEDMF%(MF) AND SW<>-1 THEN GOSUB 700 ELSE 3620
3610 IF SW<>SWOLD AND SWOLD<>-99 THEN SW=-1
3620 NEXT MF
3630 IF SW THEN TG=3 ELSE TG=8  'READ TRUE VOLT. IF HV ON ELSE READ SET VOLT.
3640 FOR I%=1 TO NR%
3650 MF=MF%(I%)
3660 CH=CHP%(I%):GOSUB 700:VP=V
3670 CH=CHN%(I%):GOSUB 700:VN=V
3680 IF INSTR(N$(I%),"OR")=2 THEN V=(VP-VN)/2 ELSE V=VP
3690 VALUE%(I%)=V:MIDVALUE%(I%)=V
3700 SPAN%(I%)=50                 'FIRST SETTING SPAN = 50
3710 NEXT I%
3720 START%=0
3730 GOSUB 1450
3740 RETURN
3750 '
3760 'HIGH
3770 '
3780 IF SW=4 OR SW=-1 THEN SW=0 ELSE SW=4
3790 FOR MF=10 TO 16
3800 IF USEDMF%(MF) THEN GOSUB 610
3810 NEXT MF
3820 C%=3:COLOR ,,C%,C%,C%
3830 LOCATE 2,33:IF SW THEN PRINT "ON  " ELSE PRINT "OFF "
3840 RETURN
```

```
3850 '
3860 '+
3870 '
3880 IF SPAN%(DNR%)=20 THEN NEWSPAN%=50
3890 IF SPAN%(DNR%)=50 THEN NEWSPAN%=100
3900 IF SPAN%(DNR%)=100 THEN NEWSPAN%=200
3910 IF SPAN%(DNR%)=200 THEN NEWSPAN%=500
3920 IF SPAN%(DNR%)=500 THEN NEWSPAN%=1000
3930 IF SPAN%(DNR%)=1000 THEN NEWSPAN%=2000
3940 SPAN%(DNR%)=NEWSPAN%
3950 ROW%=2*(DNR%-DTOP%)+3:COL%=26
3960 LOCATE ROW%,COL%:PRINT USING "####";SPAN%(DNR%);
3970 PUTX=FNPUTX(VALUE%(DNR%)-MIDVALUE%(DNR%),SPAN%(DNR%))
3980 GOSUB 2030:GOSUB 2090:GOSUB 2170
3990 RETURN
4000 '
4010 '-
4020 '
4030 IF SPAN%(DNR%)=2000 THEN NEWSPAN%=1000
4040 IF SPAN%(DNR%)=1000 THEN NEWSPAN%=500
4050 IF SPAN%(DNR%)=500 THEN NEWSPAN%=200
4060 IF SPAN%(DNR%)=200 THEN NEWSPAN%=100
4070 IF SPAN%(DNR%)=100 THEN NEWSPAN%=50
4080 IF SPAN%(DNR%)=50 THEN NEWSPAN%=20
4090 IF 2*ABS(VALUE%(DNR%)-MIDVALUE%(DNR%))>NEWSPAN% THEN 4150
4100 SPAN%(DNR%)=NEWSPAN%
4110 ROW%=2*(DNR%-DTOP%)+3:COL%=26
4120 LOCATE ROW%,COL%:PRINT USING "####";SPAN%(DNR%);
4130 PUTX=FNPUTX(VALUE%(DNR%)-MIDVALUE%(DNR%),SPAN%(DNR%))
4140 GOSUB 2030:GOSUB 2090:GOSUB 2170
4150 RETURN
4160 '
4170 'RESET
4180 '
4190 MIDVALUE%(DNR%)=VALUE%(DNR%)
4200 PUTX=FNPUTX(0,SPAN%(DNR%))
4210 GOSUB 2030:GOSUB 2090:GOSUB 2170
4220 RETURN
4230 '
4240 'BAR
4250 '
4260 ROW%=2*(DNR%-DTOP%)+3:COL%=16
4270 VALUE%(DNR%)=MIDVALUE%(DNR%)+SPAN%(DNR%)*(XB-158)/300
4280 IF INSTR(N$(DNR%),"OR")=2 THEN 4310
4290 IF VALUE%(DNR%)<0 THEN VALUE%(DNR%)=0
4300 GOTO 4320
4310 IF VALUE%(DNR%)<-500 THEN VALUE%(DNR%)=-500
4320 GOSUB 4370          'SET HV VALUE
4330 LOCATE ROW%,COL%:PRINT USING "#####";VALUE%(DNR%)
4340 PUTX=XA-1:GOSUB 2030:GOSUB 2090:GOSUB 2170
4350 RETURN
4360 '
4370 'SET HV VALUES
4380 '
4390 MF=MF%(DNR%):TG=0
4400 IF INSTR(N$(DNR%),"OR")=2 THEN 4440
4410 D=VALUE%(DNR%):CH=CHP%(DNR%):GOSUB 500
4420 CH=CHN%(DNR%):GOSUB 500
4430 GOTO 4480
4440 VOLT=VALUE%(DNR%)                          'CORRECTION ELEMENT
4450 IF VOLT>500 THEN NEG=500 ELSE NEG=1000     'VALUE FOR NEGATIVE HV UNIT
4460 D=NEG+VOLT*2:CH=CHP%(DNR%):GOSUB 500
4470 D=NEG:CH=CHN%(DNR%):GOSUB 500
4480 RETURN
4490 '
4500 'INPUT
4510 '
4520 C%=1:COLOR ,,C%,C%,C%
4530 FOR R%=2 TO 10
4540 LOCATE R%,1
4550 IF R%=2 OR R%=4 OR R%=10 THEN 4560 ELSE 4570
4560 PRINT STRING$(40,ASC("*"));:GOTO 4580
4570 PRINT SPACE$(40);
4580 NEXT R%
4590 LOCATE 3,20-LEN(TEXT$)\2:PRINT TEXT$
4600 IF TEXT$="SAVE" THEN 4660
4610 LOCATE 5,1:PRINT COMM1$
4620 LOCATE 6,1:PRINT COMM2$
4630 LOCATE 7,1:PRINT COMM3$
4640 LOCATE 8,1:PRINT COMM4$
4650 GOTO 4750
4660 SHELL"MENU OFF >NONE"
4670 LOCATE 5,1:INPUT "TYPE FILENAME: ",SELECT$:SELECT$=SELECT$+"."+EXT$
4680 LOCATE 5,1:PRINT SPACE$(40);
4690 LOCATE 5,1:PRINT "TYPE 4 LINES OF COMMENTS"
4700 INPUT COMM1$
4710 INPUT COMM2$
4720 INPUT COMM3$
4730 INPUT COMM4$
4740 SHELL"MENU GWBASIC >NONE"
4750 RETURN
```

```
4760 '
4770 ' MENU
4780 '
4790 C%=1:COLOR ,,C%,C%,C%
4800 LOCATE  2,12:PRINT "********************"
4810 LOCATE  3,12:PRINT "*                  *"
4820 LOCATE  4,12:PRINT "********************"
4830 LOCATE  5,12:PRINT "* --TOP            *"
4840 LOCATE  6,12:PRINT "* --PREVIOUS       *"
4850 LOCATE  7,12:PRINT "*                  *"
4860 LOCATE  8,12:PRINT "*                  *"
4870 LOCATE  9,12:PRINT "*                  *"
4880 LOCATE 10,12:PRINT "*                  *"
4890 LOCATE 11,12:PRINT "*                  *"
4900 LOCATE 12,12:PRINT "*                  *"
4910 LOCATE 13,12:PRINT "*                  *"
4920 LOCATE 14,12:PRINT "*                  *"
4930 LOCATE 15,12:PRINT "*                  *"
4940 LOCATE 16,12:PRINT "*                  *"
4950 LOCATE 17,12:PRINT "* --NEXT           *"
4960 LOCATE 18,12:PRINT "* --BOTTOM         *"
4970 LOCATE 19,12:PRINT "* CANCEL           *"
4980 LOCATE 20,12:PRINT "********************"
4990 LOCATE 3,20-LEN(TEXT$)\2:PRINT TEXT$
5000 ROW%=7
5010 FTOP%=1
5020 IF FTOP%<1 THEN FTOP%=1
5030 IF FTOP%>FLAST% THEN FTOP%=FTOP%-10
5040 FOR R%=7 TO 16
5050 F%=FTOP%+R%-7
5060 IF F%<=FLAST% THEN Q1$=MENU$(F%) ELSE Q1$=SPACE$(16)
5070 LOCATE R%,14:PRINT Q1$
5080 NEXT R%
5090 PUT (104,ROW%*8-8),INV%,XOR
5100 I$=INKEY$:IF I$="" THEN 5100
5110 IF I$=UP$ OR I$=DO$ OR I$=CHR$(13) THEN PUT (104,ROW%*8-8),INV%,XOR ELSE 51
00
5120 IF I$=UP$ THEN ROW%=ROW%-1
5130 IF I$=DO$ THEN ROW%=ROW%+1
5140 F%=FTOP%+ROW%-7:IF ROW%>6 AND ROW%<17 AND F%>FLAST% THEN 5120
5150 IF I$=CHR$(13) THEN 5200
5160 I$=INKEY$:IF I$<>"" THEN 5160
5170 IF ROW%<5 THEN ROW%=5
5180 IF ROW%>19 THEN ROW%=19
5190 GOTO 5090
5200 IF ROW%=5 THEN 5010
5210 IF ROW%=6 THEN FTOP%=FTOP%-10:GOTO 5020
5220 IF ROW%=17 THEN FTOP%=FTOP%+10:GOTO 5030
5230 IF ROW%=18 THEN FTOP%=10*((FLAST%-1)\10)+1:GOTO 5030
5240 IF ROW%=19 THEN SELECT$="CANCEL":GOTO 5260
5250 SELECT$=MENU$(F%)
5260 RETURN
```

```
10 REM
20 REM          SCANNER
30 REM          by Petter Broman Summer Student 1989
40 REM
50        CLEAR   .57611!
60        IBINIT1 = 57611!
70        IBINIT2 = IBINIT1 + 3
80        BLOAD "BIB.M",IBINIT1
90        CALL IBINIT1(IBFIND,IBSTOP,IBTRG,IBCLR,IBPCT,IBSIC,IBLOC,IBPPC,IBBNA,IB
ONL,IBRSC,IBSRE,IBRSV,IBPAD,IBSAD,IBIST,IBDMA,IBEOS,IBTMO,IBEOT)
100        CALL IBINIT2(IBGTS,IBCAC,IBWAIT,IBPOKE,IBWRTF,IBWRTA,IBWRT,IBCMDA,IBCM
D,IBRDF,IBRDA,IBRD,IBRPP,IBRSP,IBDIAG,IBXTRC,IBSTA%,IBERR%,IBCNT%)
110       CMD$ = SPACE$(10)        ' command buffer
120       RD$ = SPACE$(3)          ' read buffer
130       WRT$ = SPACE$(255)       ' write buffer
140       BDNAME$ = SPACE$(7)      ' board/device name
150       DIM SPECTRUM%(4096)      ' spectrum array
160 BDNAME$="GPIB0":CALL IBFIND(BDNAME$,GPIB0%)
170 BDNAME$="DEV1":CALL IBFIND(BDNAME$,AMP%)
180 BDNAME$="DEV0":CALL IBFIND(BDNAME$,CC%)
190 WRTCC$=CHR$(16)+CHR$(0)+CHR$(20)     'CHANNEL 1 AND SLOT 20 IS USED ON CC
200 COLOR 5,0,0:CLS
210 INPUT "FIRST CHANNEL";FIRST%
220 INPUT "LAST CHANNEL ";LAST%
230 INPUT "STEP LENGTH  ";STP%
240 IF FIRST%>LAST% THEN STP%=-STP%
250 IF STP%=0 THEN 740
260 CLS
270 PRINT "MAKE SURE THAT THE AMPEREMETER AND"
280 PRINT "THE CAMAC CONTROLLER ARE SWITCHED ON!"
290 PRINT "THEN PRESS ANY KEY!"
300 IF LEN(INKEY$)=0 THEN 300
310 CLS:PRINT "INITIALISING!"
320 FOR R%=0 TO 4095:SPECTRUM%(R%)=0:NEXT R%
330 CALL IBSIC(GPIB0%)
340 CMD$="?B6"+CHR$(4):CALL IBCMDA(GPIB0%,CMD$)  'SDC TO AMP
350 CMD$="?B6"+CHR$(4):CALL IBCMD(GPIB0%,CMD$)   'SDC TO AMP
360 CMD$="?B%"+CHR$(4):CALL IBCMD(GPIB0%,CMD$)   'SDC TO CC
370 RD$=SPACE$(1)
380 WRT$=CHR$(35)
390 CALL IBWRT(CC%,WRT$)
400 CALL IBRD(CC%,RD$)
410 WRT$=WRTCC$+CHR$(0)+CHR$(0)
420 CALL IBWRT(CC%,WRT$)
430 CALL IBRD(CC%,RD$)
440 WRT$="T4X":CALL IBWRT(AMP%,WRT$)
450 WRT$="G1X":CALL IBWRT(AMP%,WRT$)
460 WRT$="R4X":CALL IBWRT(AMP%,WRT$)
470 WRT$="C0X":CALL IBWRT(AMP%,WRT$)
480 CHANNEL%=FIRST%:GOSUB 820
490 RD$=SPACE$(1)
500 OLDTIME$=TIME$
510 REM
520 REM
530 CLS:PRINT "START OF SCANNING!"
540 FOR CHANNEL%=FIRST% TO LAST% STEP STP%
550 TIME$="00:00:00"                        'RESET TIMER
560 MSB%=CHANNEL%\256:LSB%=CHANNEL% AND 255

570 WRT$=WRTCC$+CHR$(LSB%)+CHR$(MSB%)
580 CALL IBWRT(CC%,WRT$)
590 CALL IBRD(CC%,RD$)
600 'CMD$=CHR$(8):CALL IBCMD(AMP%,WRT$)   'POSSIBLE TRIGGER
610 GOSUB 820                             'READ AMP METER
620 PRINT CHANNEL%,TIME2,B$,STO%
630 NEXT CHANNEL%
640 GOSUB 820                                   'LAST READING DUE TO LAG
650 PRINT "LAST",TIME2,B$,STO%
660 WRT$=WRTCC$+CHR$(0)+CHR$(0)                 'RESET MAGNET
670 CALL IBWRT(CC%,WRT$)
680 CALL IBRD(CC%,RD$)
690 PRINT "END OF SCANNING!"
700 INPUT "SAVE DATA AS (RETURN=SKIP)";N$
710 IF LEN(N$)=0 THEN 730
720 GOSUB 990
730 TIME$=OLDTIME$
740 COLOR 7
750 RUN "START.BAS"
760 REM
770 REM
780 REM READ FROM AMPEREMETER
790 REM IN: CHANNEL%
800 REM OUT:B$,STO%,CAL%(),TIME2
810 REM
820 CMD$="V"+CHR$(34)                           'GPIB0 LISTEN & AMP TALK
830 CALL IBCMD(GPIB0%,CMD$)
840 B$=""
850 A$=SPACE$(1)
860 CALL IBRDA(GPIB0%,A$)
870 IF ASC(A$)>42 THEN B$=B$+A$
880 IF A$<>CHR$(10) THEN 850
890 CMD$="_?"
900 CALL IBCMD(GPIB0%,CMD$)
910 I=VAL(B$)
920 IF I<.000003 THEN STO%=INT(I*1E+10+.5) ELSE STO%=32767
930 IF CHANNEL%=FIRST% THEN 970
940 FOR R%=CHANNEL%-STP% TO CHANNEL%-SGN(STP%) STEP SGN(STP%)
950 IF R%>=0 AND R%<=4095 THEN SPECTRUM%(R%)=STO%
960 NEXT R%
970 TIME2=TIMER
980 RETURN
990 REM
1000 REM  FILE OUTPUT
1010 REM
1020 OPEN N$ FOR OUTPUT AS #1
1030 FOR R%=0 TO 4095
1040 PRINT #1,USING "########"; SPECTRUM%(R%)
1050 NEXT R%
1060 CLOSE #1
1070 RETURN
```

Appendix D

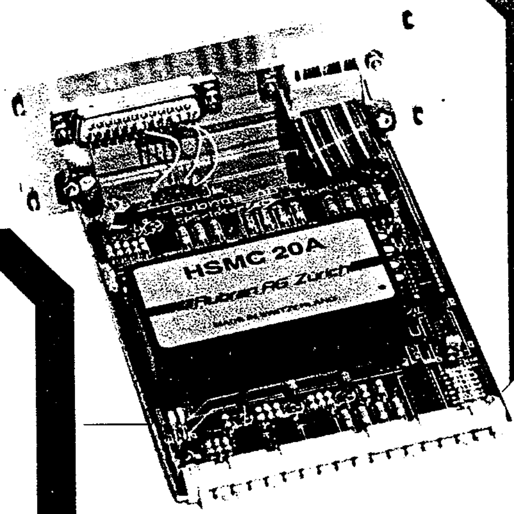
DETAILS ABOUT THE STEPMOTOR DEVICES

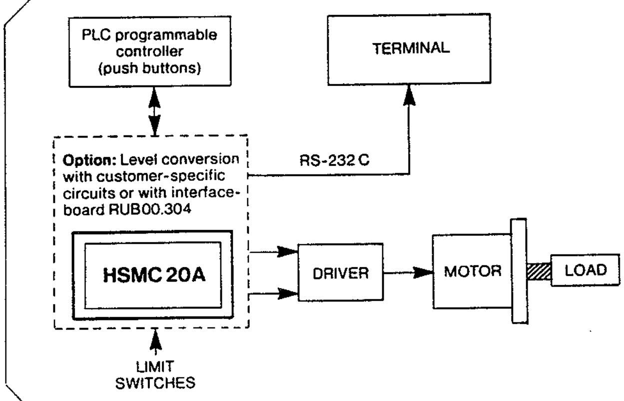
TECHNICAL DETAILS

The stepping motor system devices are here shown with layout, connections, and in some cases circuit diagrams. The following is shown in order of appearance:

HSMC 20A (manual)
HSMC 20A and 5V Net Unit (connections)
The 8-Switch-Box
The Electronics Rack
The Step Motor Driver

HSMC 20A





Schrittmotor-Controller HSMC 20A RUB00.102

HSMC-Interface-Karte RUB00.304

Der HSMC 20A und die Interface-Karte ist eine ideale Kombination, um Schrittmotorachsen mit speicherprogrammierbaren Steuerungen (SPS) zu betreiben.

Die HSMC-Interface-Karte dient der Anpassung des Schrittmotor-Controllers an die 24 VDC Ein- und Ausgänge von jeder beliebigen SPS. Die Steuereingänge sowie die Controller-Ausgänge sind über Optokoppler galvanisch getrennt.

Programmierung

(Speicherung in EEPROM).
Zur Programmierung der 24 Distanzen sowie dem Geschwindigkeitsprofil steht als Kommunikations-Schnittstelle sowohl eine RS-232 (+/−12 V) als auch eine RS-232 mit TTL-Pegel zur Verfügung. Die Distanzen sind über 5 Steuereingänge anwählbar.

Commande-moteur pas à pas HSMC 20A RUB00.102

Interface HSMC RUB00.304

Le module HSMC 20A et l'interface forme une combinaison idéale pour un entrainement avec moteur pas à pas conduit par un sytème AP.

L'interface HSMC convertit le niveau des signaux d'entrée et de sortie (24 VDC − TTL), permettant l'emploi du module de commande avec chaque système AP. Les entrées et sorties sont protégées par des opto-coupleurs (séparation galvanique).

Programmation

Les 24 distances et le profil de vitesse sont mémorisés dans un EEPROM. Une interface RS-232 (+/−12 V ou TTL) est à disposition. Un système AP peut sélectioner les distances par l'intermédiaire de 5 entrées.

Stepping motor controller HSMC 20A RUB00.102

HSMC Interface card RUB00.304

Our HSMC 20A, together with the interface board, is a powerful subsystem for stepping motor applications in combination with PLC-controllers.

The HSMC interface board operates with 24 VDC signals and converts these for the stepping motor controller to TTL-level. All inputs and outputs on the card connector are optically coupled and can be connected to any PLC with 24 V signal levels.

Programming

24 distances and the speed profile can be entered and stored internally in EEPROM. Communication is via a standard RS-232 port with either +/−12 V or TTL-signals. The 24 distances can be selected by the PLC via 5 select inputs.

Rubran AG Zürich
Rietstrasse 15 · Postfach
CH-8108 Dällikon
Telefon 01/844 44 14
Telex 825 218

Kurz-Daten HSMC 20A:

- 20 KHz max. Frequenz
- 3 Betriebsarten:
  - SPS-Betrieb
  - Terminal-Dialogbetrieb
  - Handbetrieb
- Terminalschnittstelle:
  - Vollduplex (4800 Baud)
  - serielle Übertraqunq ASCII-Code
  - 8 Bit Wort, 1 Start/Stop Bit, keine Parität

Steuerfunktionen über SPS:

- Verfahrgeschwindigkeit
- Drehrichtung
- relativ/absolut Verfahren
- Nullposition
- Terminal- oder SPS-Betrieb

Spécifications HSMC 20A

- Fréquence max: 20 KHz
- 3 modes d'exploitation:
  - système AP
  - mode terminal (RS-232)
  - système manuel
- Interface RS-232:
  - 4800 Baud (full duplex)
  - code ASCII (8 bits)
  - 1 Start/Stop bit, pas de parité

Commande via système AP

- vitesse d'exploitation
- sens de rotation
- positionnement absolu/relatif
- référence
- mode terminal ou système AP

Main features of HSMC 20A

- 20 kHz max. stepping rate
- 3 modes of operation:
  - PLC mode
  - terminal/dialog mode
  - manual mode
- RS-232 port for terminal:
  - full duplex 4800 Baud
  - 8 bit ASCII-code
  - 1 start bit, 1 stop bit, no parity

Control functions via PLC

- slow/fast stepping rate
- Direction cw/ccw
- Relative/absolute positioning
- Home search
- Terminal mode or PLC mode

Der Vorteil des HSMC 20A-Funktionsprinzips

Mit dem HSMC 20A sind die Betriebsarten Start/Stop und Hochlauf möglich. Grundlage für den Hochlaufbetrieb des HSMC 20A bildet ein trapezförmiges Geschwindigkeitsprofil, dessen lineare Rampen den Schrittmotor mit konstanter Winkel-Beschleunigung bewegen.

Das Frequenz-Zeit-Profil wird charakterisiert durch die zwei Parameter:

- Beschleunigung bzw. Steilheit der Rampe
- Maximalgeschwindigkeit bzw. maximale Schrittfrequenz

Diese zwei Kenngrössen können beim HSMC 20A in weiten Grenzen stufenlos programmiert werden. Dadurch lässt sich eine Schrittmotor-Ansteuerung weitgehend den kinematischen Gegebenheiten von Schrittmotor und mechanischer Last anpassen.

Das im HSMC 20A verwendete Funktionsprinzip hat den besonderen Vorteil, dass jeder Positionierbefehl in eine mathematisch exakt definierte Pulsfolge umgewandelt wird. Dadurch lassen sich bei einer SPS-Ablaufsteuerung die Positionierzeiten des Motors für die programmierten Distanzen genau vorausberechnen und jederzeit reproduzieren.

Déscription des avantages du HSMC 20A

Le module HSMC 20A offre les modes d'explitation marche/arrêt et en accélération. Un profil trapézoïdal de l'accélération constitue la base de la fonction fréquence/temps, profil dont les rampes linéaires assurent la rotation du moteur avec une accélération angulaire constante.

Cette fonction fréquence/temps se définit par les paramètres suivants:

- accélération (inclinaison de la rampe)
- vitesse max (fréquence max)

La plage de programmation des deux paramètres permet de les définir librement. Ceci offre une possibilité d'adaptation cinématique du moteur pas à pas et de la charge mécanique parfaite.

Le principe de fonctionnement du HSMC 20A offre l'avantage particulier que chaque commande de positionnement est convertie, au niveau mathématique, par un train d'impulsions précis. Ce principe permet à un système AP de calculer le temps requis au positionnement du moteur pour une distance préprogrammée.

The HSMC 20A operating principle and its advantages

The HSMC 20A can be operated in the start/stop mode or in the ramping mode. The ramping mode is based on a trapezoidal frequency-time-profile. This means linear ramps which cause constant acceleration and deceleration of the motor.

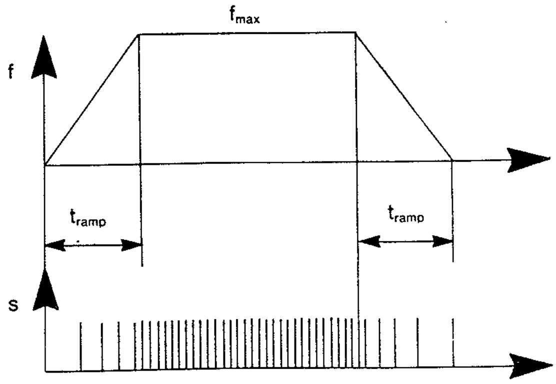
The frequency-time-profile is defined by only two parameters:

- acceleration (ramp slope)
- maximum stepping rate

Both parameters are fully programmable over a wide range. Thus, the controller can be adapted to virtually every combination of power drive, motor and mechanical load.

The operating principle of the HSMC 20A offers a unique advantage: It delivers exactly defined pulse trains and each individual time interval between two step pulses is calculated in real-time. This results in a smooth mechanical motion. The positioning times are precisely reproducible, which can be important in a PLC program sequence.

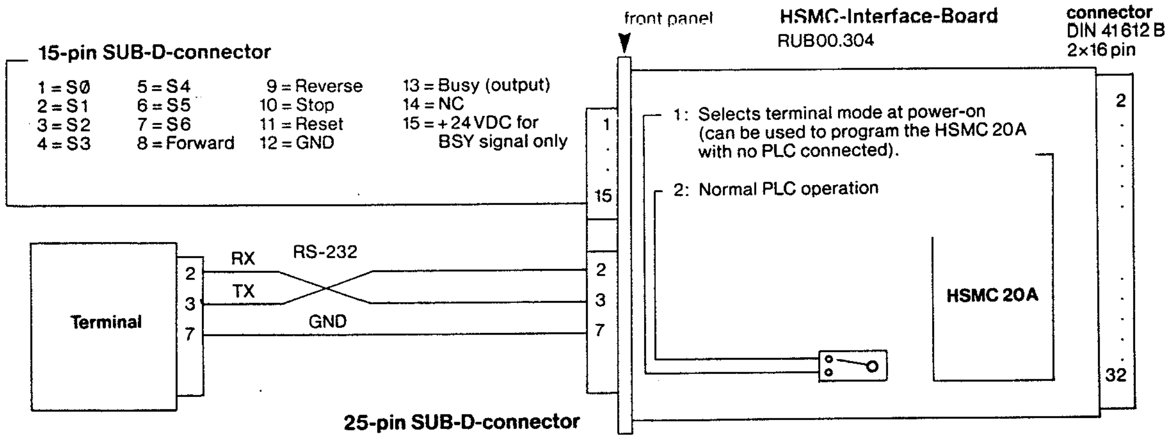
Specifications	min	max	Unit
Start/Stop-Frequency	33	10 000	Hz
Acceleration	1	1 000	kHz/s
Maximum Frequency	33	20 000	Hz
Positioning range		~16 Mio	Steps
Max. ramp steps		65 535	Steps
Ramp time to f = 20 kHz		6 500	msec
Resolution of time interval		0,5	µs



Zur Inbetriebnahme des HSMC 20A benützen Sie bitte unsere ausführliche Betriebsanleitung.

Pour la mise en service du module HSMC 20A, veuillez consulter notre manuel d'emploi.

Please refer to our comprehensive HSMC 20A operating manual for further detalis.



Pin-out in DIN 41612 B connector

outputs (b)	pin	inputs (a)
	2	ground
	4	common of output signals
	6	common of input signals
	8	LSX zero
	10	XLS emergency stop
	12	LS2 limit switch max.
	14	LS1 limit switch min.
	16	GND RS-232
	18	RX RS-232/TTL
RS-232/±12 V TX	20	
alarm output EMY	22	RX RS-232/±12 V
step pulse PUL	24	
RS-232/TTL TX	26	
(cw/ccw) DIR	28	
	30	
	32	+5V supply

DIN 41 612 B
2×16 pin

Schema HSMC Interface

Hinweis:

Wird der HSMC 20A mit der Interface-Karte RUB00.304 verwendet, muss beachtet werden, dass bei unbeschalteten Eingängen von Seiten SPS die Eingänge am Controller HSMC 20A «High» sind.

Änderungen vorbehalten

Schéma HSMC Interface

Remarque:

Les lignes d'entrée HSMC 20A avec la carte d'interface RUB00.304 non branchées au système AP ont un niveau logique haut (high).

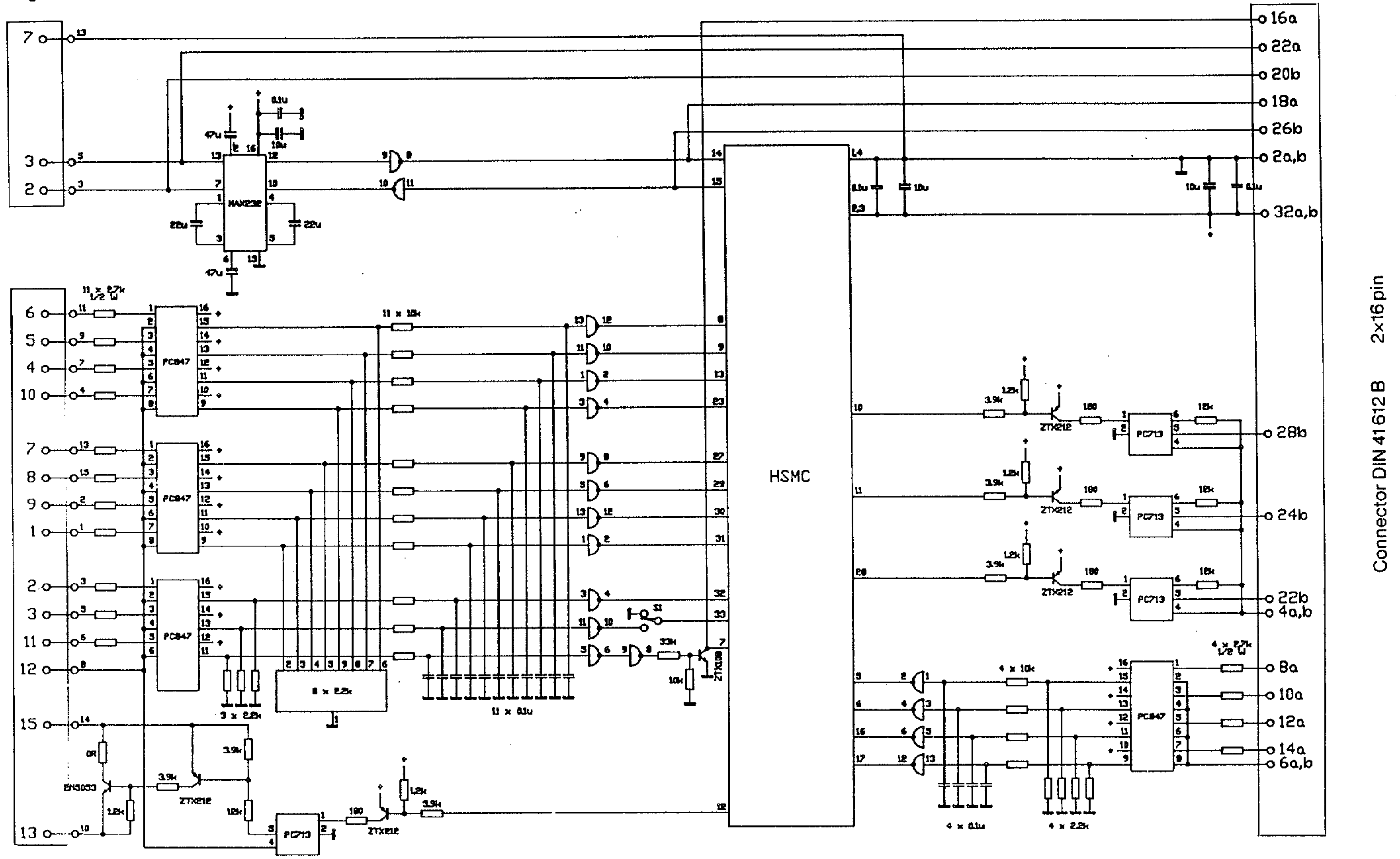
Sous réserve de modification

Circuit Diagram of HSMC Interface Board

Note:

Unconnected PLC-inputs will cause a Logic «high» at the input of the HSMC 20A.

Subject to chance without notice

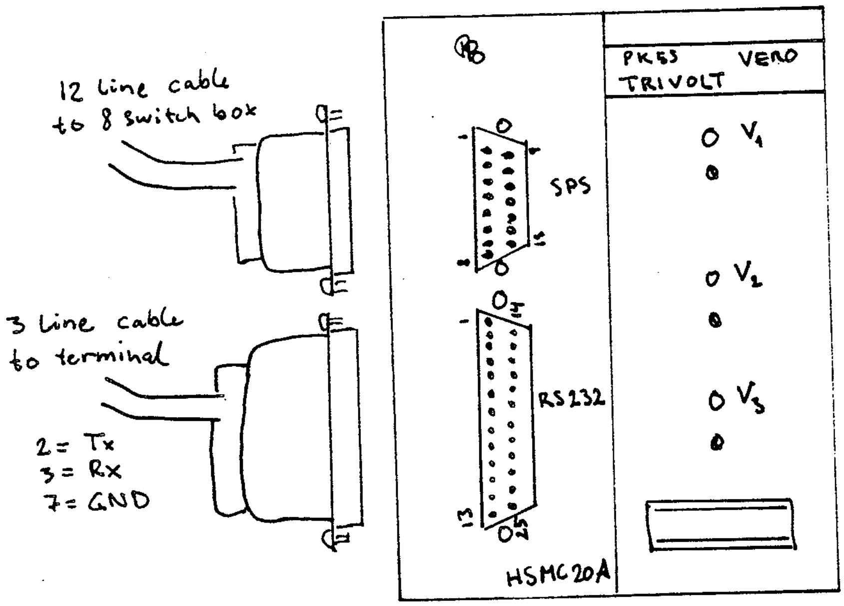


Rubran AG Zürich

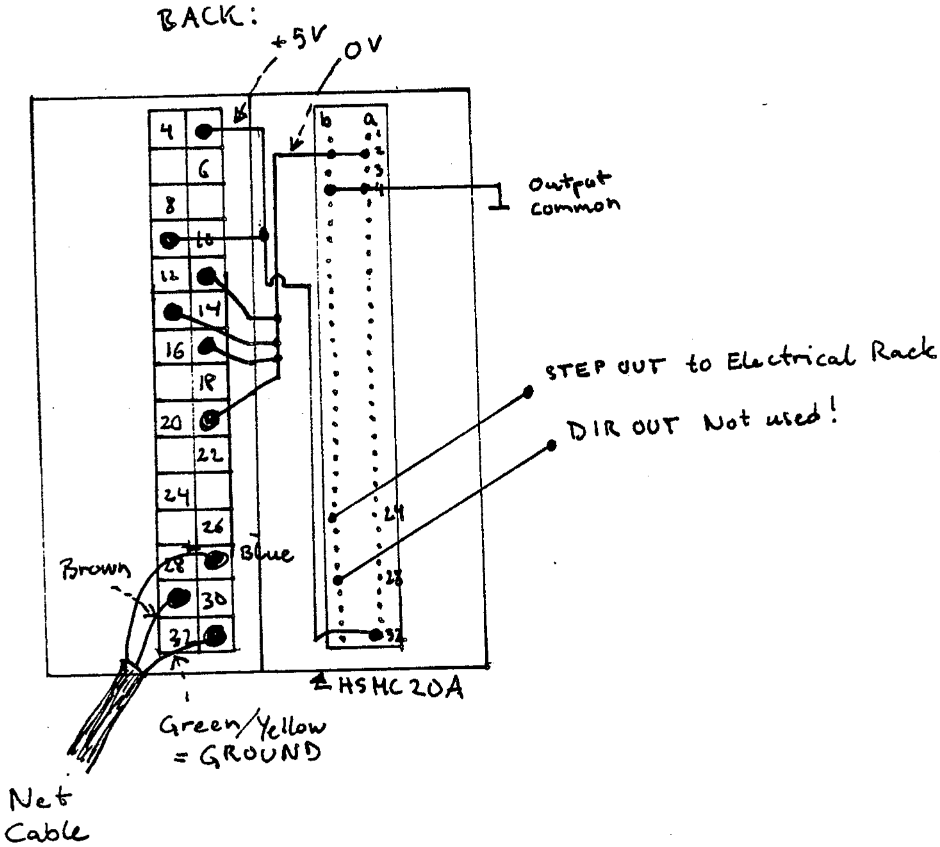
Rietstrasse 15 · Postfach
CH-8108 Dällikon
Telefon 01/844 44 14
Telex 825 218

The HSMC 20A & 5U Net Unit -connections

FRONT:

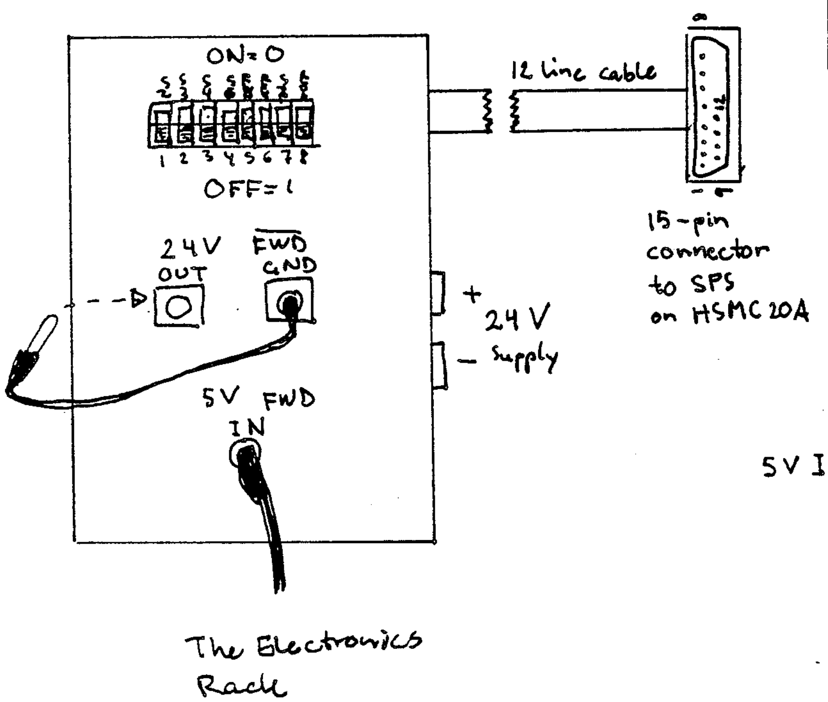


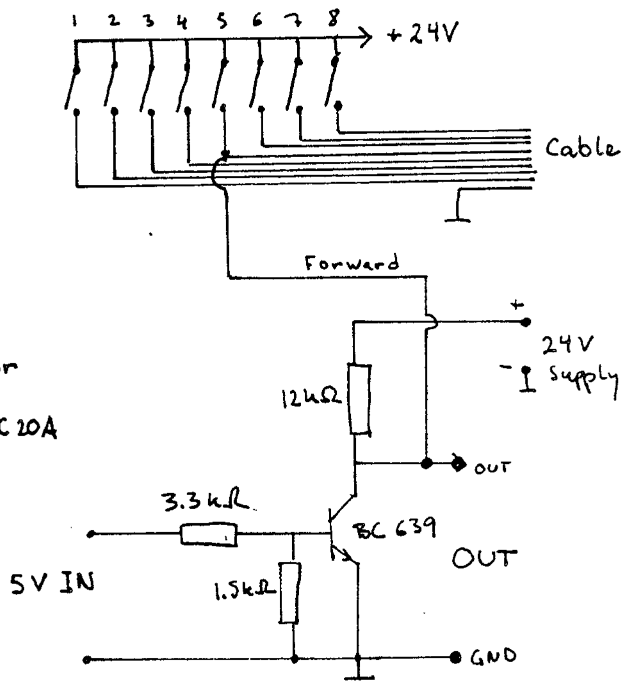
BACK:



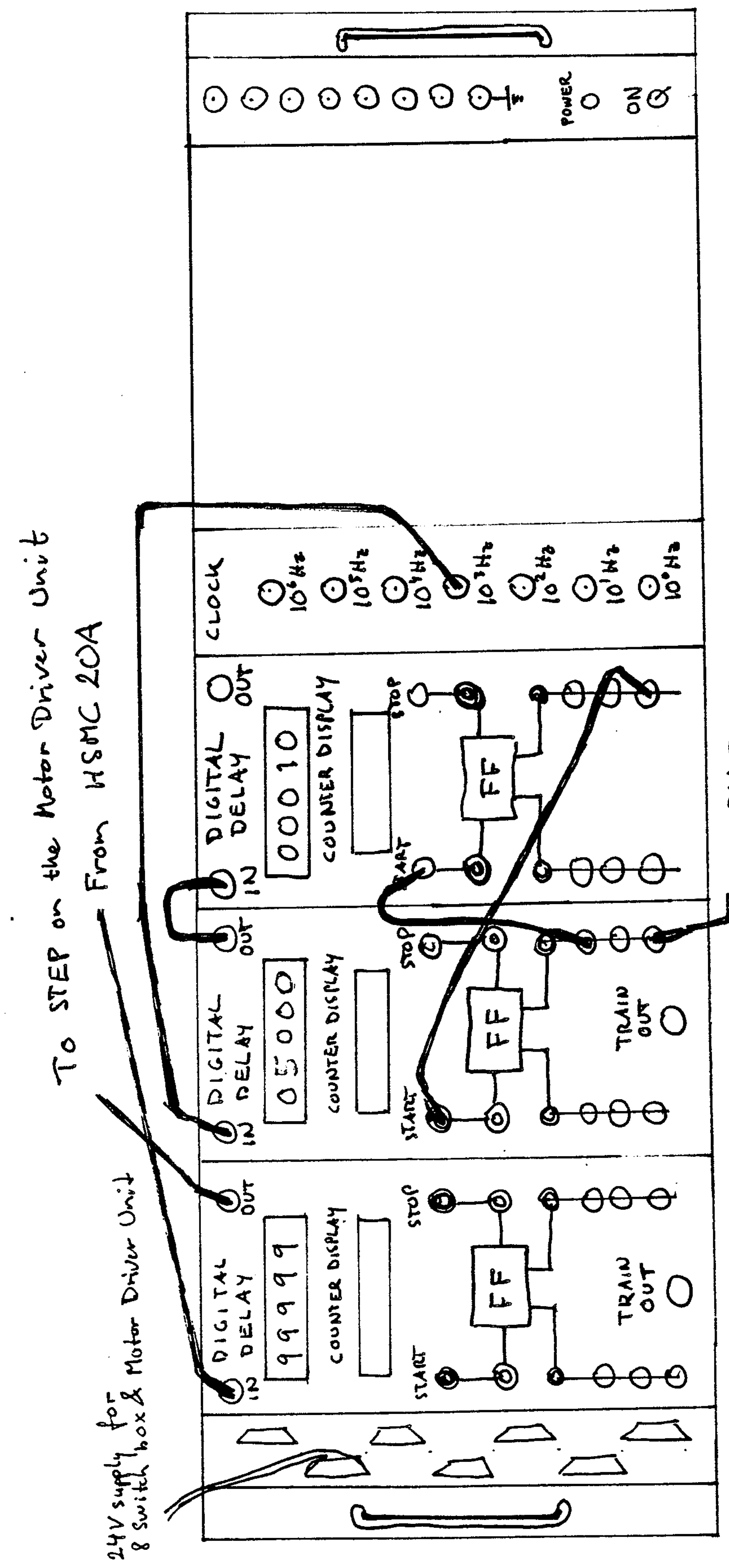
The 8 Switch Box

15-pin	line	Colour in cable	Switch
1	S0	red	
2	S1	violet	
3	S2	white	1
4	S3	green	2
5	S4	grey/pink	3
6	S5	blue/red	
7	S6	yellow	4
8	Forward	grey	5
9	Reverse	black	6
10	Stop	blue	7
11	Reset	pink	8
12	GND	brown	
13 - 15	Not in cable		

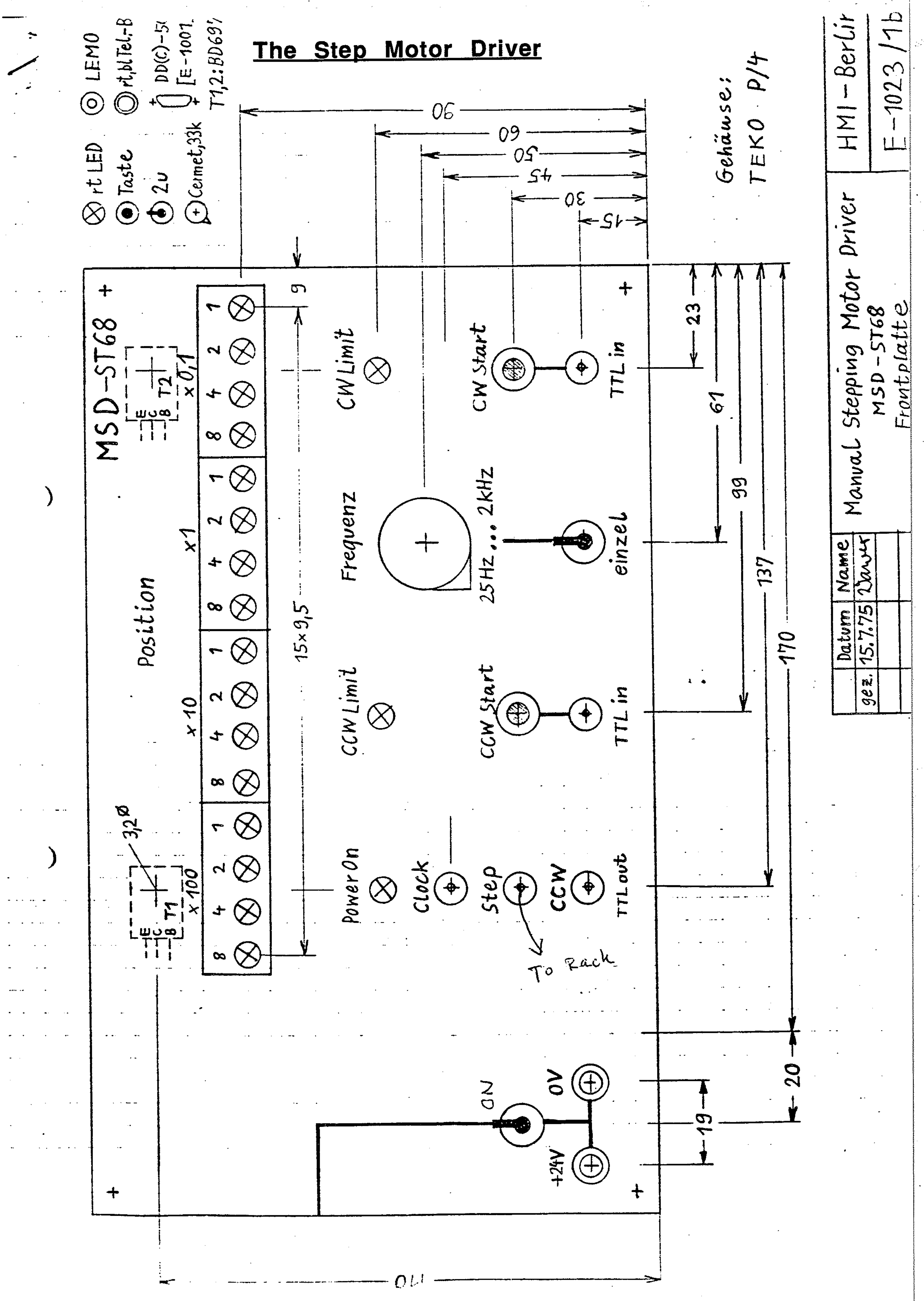


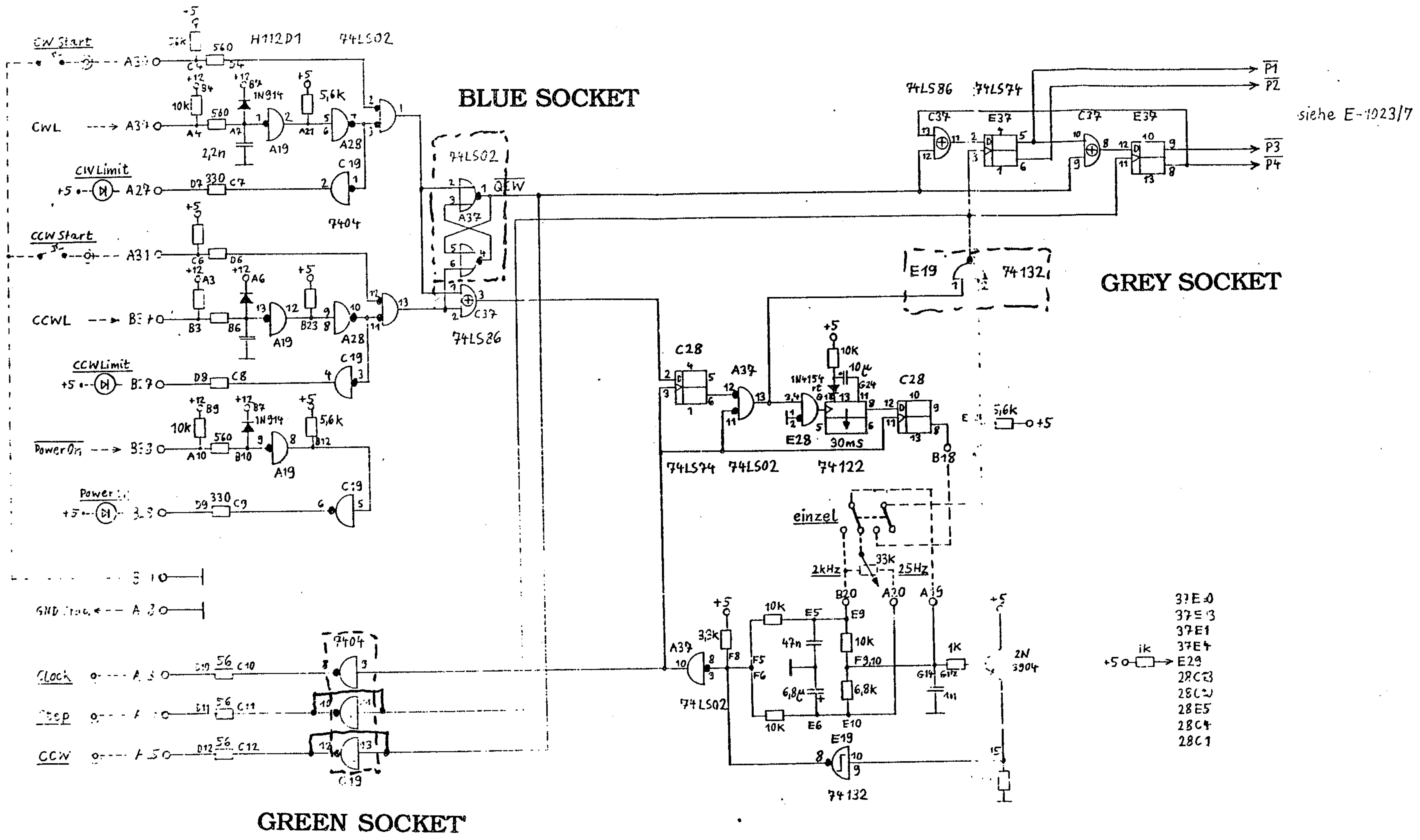


The Electronics Rack



The Step Motor Driver



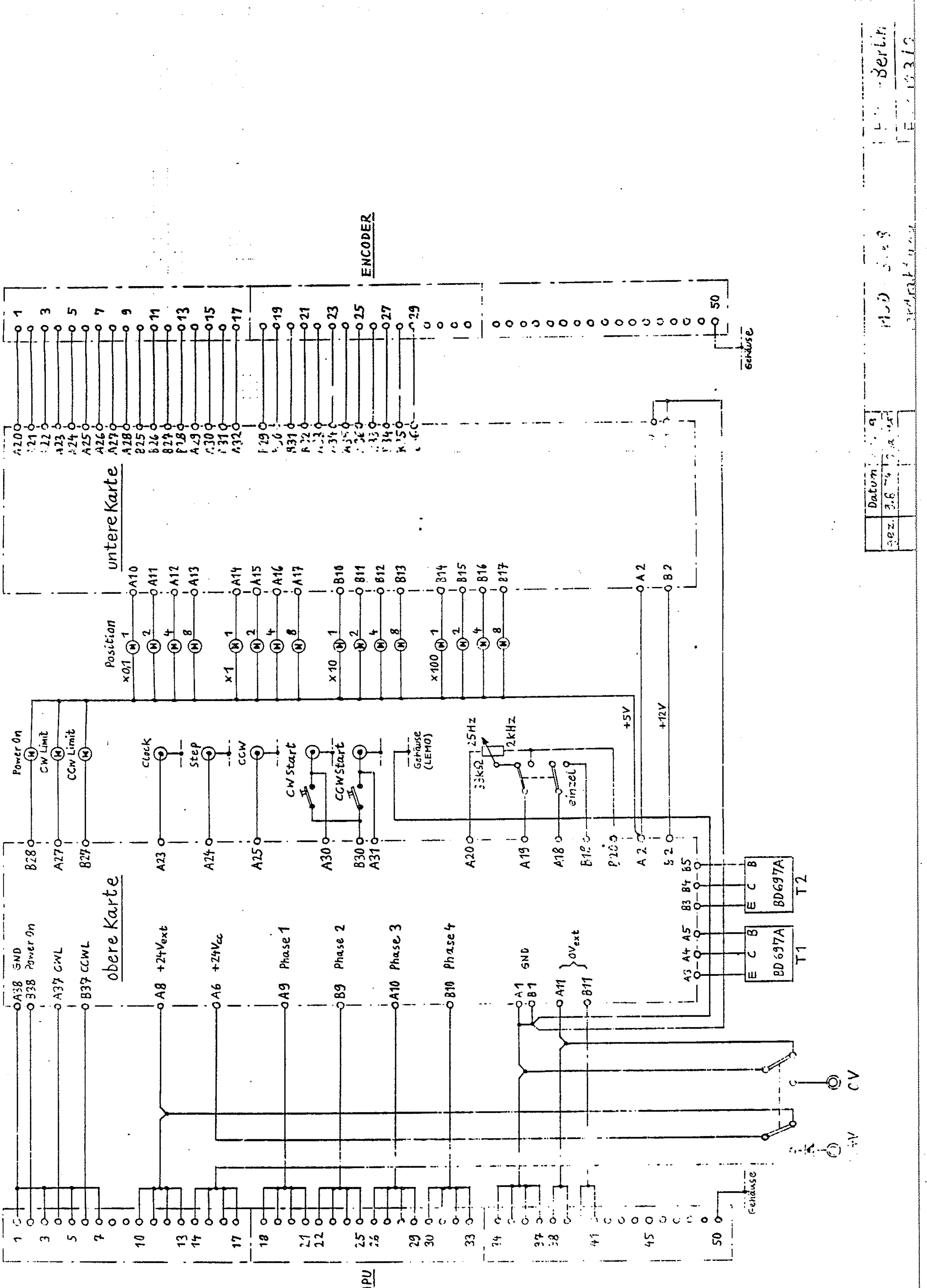


New IC's to plug
into the sockets
are left in the
bottom of the box.

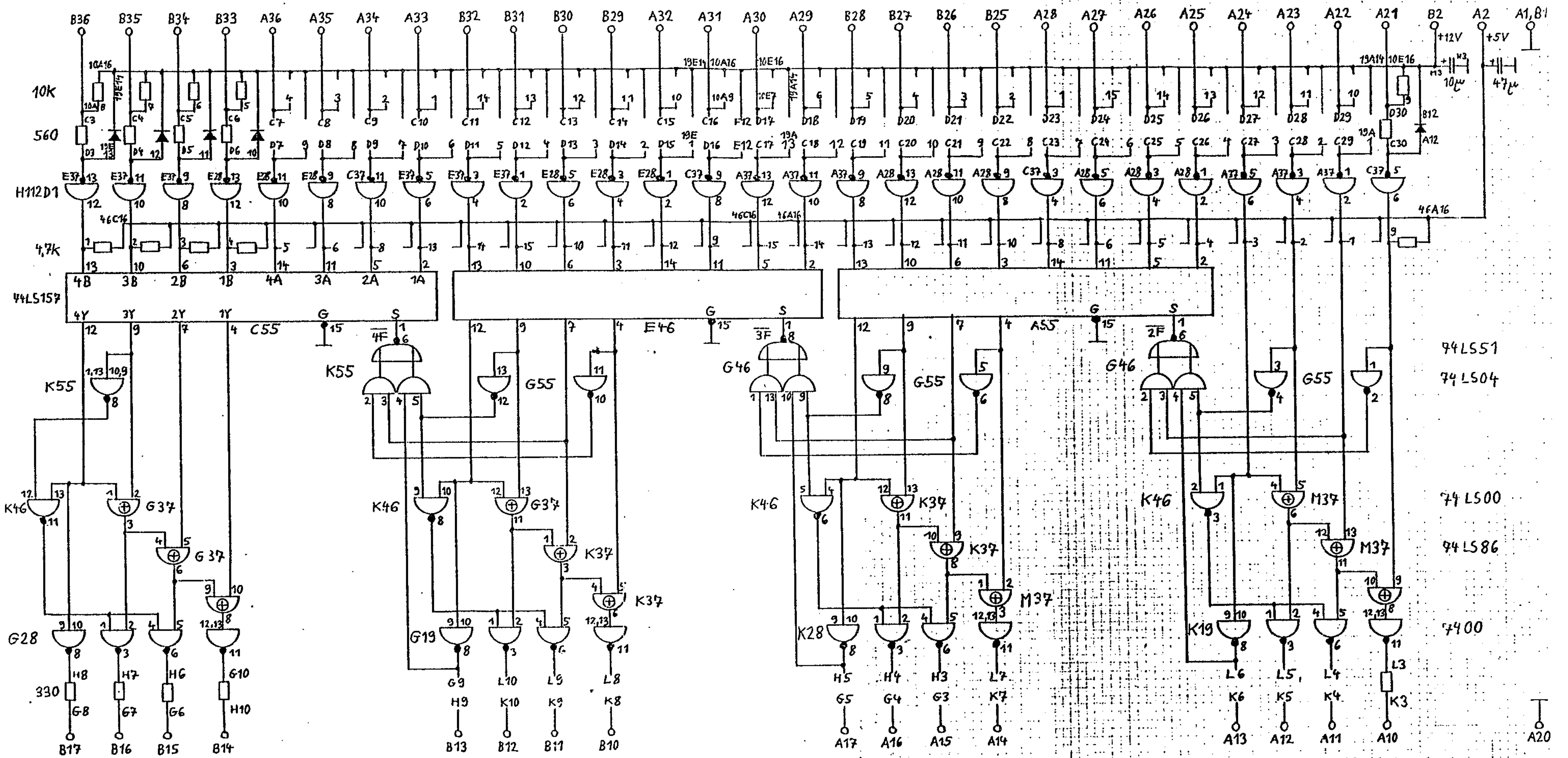
	Datum
gez.	28.8.74

Steuerung, MS...-ST63

HMI Berlin



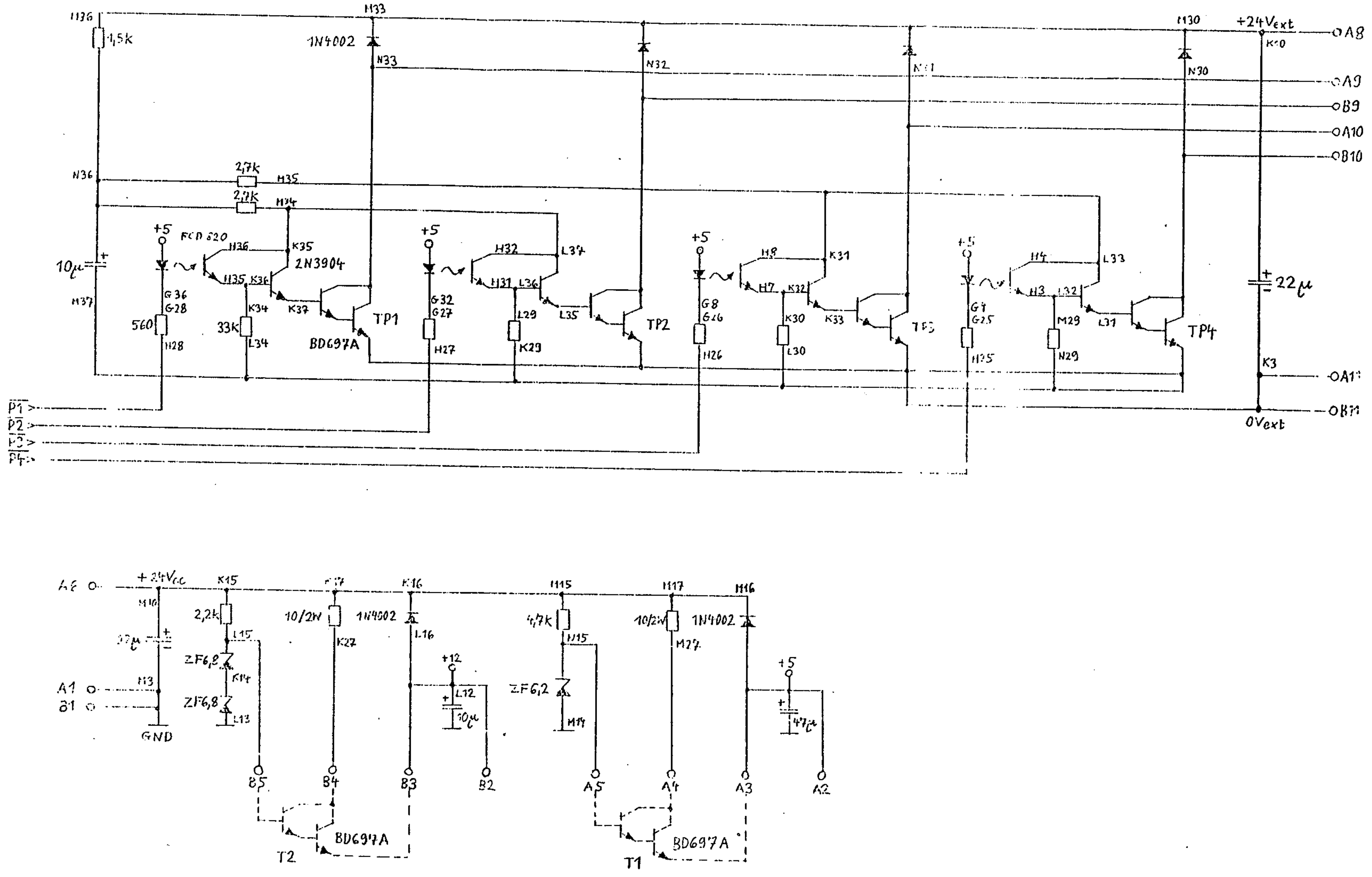
connector
to the motor



	Datum	Name
gez.	26.8.74	[illegible]

Decoder, MSD-ST68
(Schaltung)

HMI-Berlin



gez.	15.9.75	Treiber, Versorgung, MSD-STG?	HMI