Nuclear Instruments and Methods in Physics Research A 332 (1993) 554–560
North-Holland

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH
Section A

Analysis of simulated triples γ-ray data on a 100-processor transputer system

T. Lönnroth
Department of Physics, Åbo Akademi University, Turku, Finland

J. Hattula, R. Julin and A. Lampinen
Department of Physics, University of Jyväskylä, Finland

M. Aspnäs, R.J.R. Back, J. Granlund and P. Waxlax
Department of Computer Science, Åbo Akademi University, Turku, Finland

Received 22 February 1993

We present a study of the administration and analysis of simulated nuclear 3-fold coincidences implemented on a 100-processor transputer system, which has a computational capacity of about 150 Mflops. From a simple model of a level scheme the corresponding "data" is extracted in the form of three-dimensional addresses with background. The data structure and the analysis rate is studied. Further, a set of real data is used to estimate the rate of sort to be performed. The feasibility of using such a system to analyse large data spaces is discussed.

1. Introduction

During the last few years several multidetector arrays have been developed. Such arrays are NORDBALL [1,2] and 8π [3], and the future EUROGAM [4], EUROBALL [5] and GAMMASPHERE [6]. The main purpose of such an equipment is to increase the coincidence efficiency of the higher-fold events, which is of importance when high sensitivity is required. Increased sensitivity is needed in order that new phenomena be observed. In the 2-fold data, coincidences constitute peaks above a plane and various combinations of peak–tail or tail–tail constitute the background. In addition there is the independent continuum background. If this matrix is projected onto one of its axes there is a loss in resolution: the peaks are separated within the plane but in the projection they may fall on top of each other. Gating constitutes of taking peak–background differences, but a better way which reduces the statistical and Compton noise, is to make a linear fit [7]. Going to higher folds improves the sensitivity, and recently various methods to do two-dimensional fits have been proposed [8–10].

The larger efficiency of detector arrays allow for observation of 3-fold events or higher. This is due to the well-known expression for the probability $p(n)$ of having n hits, an n-fold event, in an array of N detectors:

$$p(n)=\binom{N}{n}(1-M\Omega)^{N-n}(M\Omega)^{n},$$

where M is the multiplicity in a cascade, and Ω is the detector efficiency. In the 3-fold case the triple-photopeak information is clustered into stars in a cube, and analogously for the higher folds. However, for increasingly effective detector arrays the data *size* becomes a parameter of importance. The spectroscopic data is in the form of 12 bits or more, and the corresponding two-dimensional matrix then has an address space of ≥ 32 MB. When recording three-dimensional spectra the corresponding space needed is about 180 GB, but considering that each triple event is present in six different addresses (as long as their time order is not important), namely as the permutations of (x, y, z), one may fold the cube to one sixth of it, which reduces the space to about 25 GB.

Usually the 3-fold data is reduced to the two-dimensional case by gating on transitions in the third dimension, projecting out a number of 2-fold matrices. The problem of sorting 3-fold data with a larger number of events was studied for the first time in ref. [11], but although presently much larger disks are commonly available, the main problem of disk space persists. Also, it was shown in ref. [11] that the data filled only 0.15% of the cube, i.e. out of 1×10^{10} points $\sim 2 \times 10^{8}$ are filled by events. The situation worsens for the higher folds. A recent study shows that for 10^{9}–10^{10} 4-fold events the mean data density is only $\sim 10^{-5}$ counts/address [12].

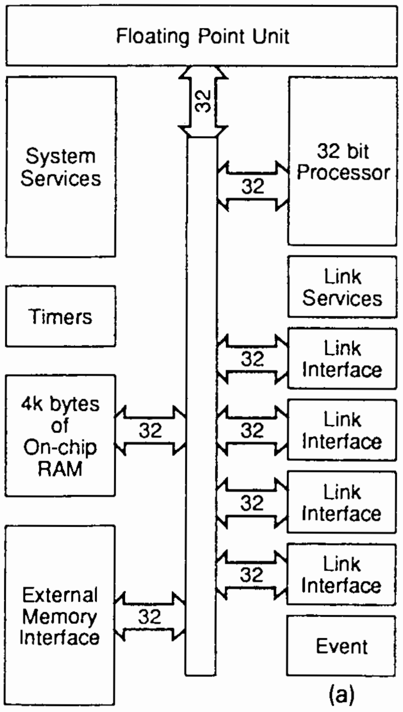
The task of working in the full three-dimensional space thus brings in two major problems, the one of handling data sets the size of which easily are in excess of several GB, and the other the total CPU time required to analyze this data space. The present work concentrates on the analysis of the data. The 3-fold data is simulated in a simple manner, using a model level scheme, and this "coincidence data" is analyzed on a multiprocessor system, consisting of 100 processors. A detailed report on the computational and program features of this work can be found in ref. [13].

2. Description of the transputer multiprocessor system HATHI-2

The transputer [#1] is a microprocessor, the name of which derives from the combination "*trans*istor com*puter*". This suggests that the microprocessor could be used to build powerful multiprocessor systems analogously to the way that computers are built from transistors. A Transputer is a single VLSI device with a processor, a local memory and a number of communication links for connection to other transputers.

The IMS T800 transputer is a 32-bit RISC-type microprocessor which also contains a 64-bit floating-point processor, four 20 Mbit/s DMA communications links, 4 kB of on-chip RAM and a 32-bit memory interface on a single 1.5 μm CMOS chip. This 4 kB of RAM provide a data rate of up to 80 MB/s. The processor can directly access a linear address space of up to 4 GB. The 32-bit wide memory interface uses multiplexed data and address lines, and provides a data rate of up to 25 MB/s. For a processor clock speed of 20 MHz, the T800 processor has a performance of about 10 MIPS or 1.5 Mflops. The architecture of the T800 transputer is illustrated in fig. 1a. See ref. [14] for further details.

The HATHI-2 multiprocessor system, located at the Department of Computer Science of Åbo Akademi University, consists of 100 T800 transputers, each currently with 1.25 kB of RAM, but expandable to 4 kB per processor. It is connected to the local network (whose main Internet node is FINABO.ABO.FI) via a Sun3 workstation. The system can be characterized as a modular, reconfigurable, generalpurpose multipro-



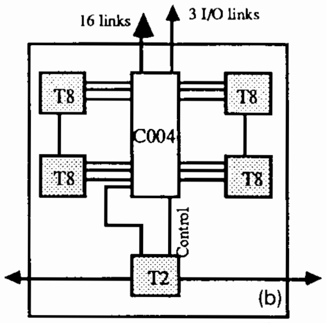


Fig. 1. (a) The IMS T800 transputer architecture. It is a 32-bit microprocessor with a 64-bit floating-point processor, four communications links, 4 kB of on-chip RAM and a memory interface. For a processor clock speed of 20 MHz, the performance is about 1.5 Mflops. See ref. [15] for further details. (b) The board architecture of HATHI-2. Each board contains four T800 microprocessors, and links to the network.

cessor system. It consists of 25 identical processor boards, as that illustrated in fig. 1b. The boards contain four T800 transputers (labelled T8), a C004 link cross-bar switch and a T212 (labelled T2) transputer to control the setting of the switch and perform other system service tasks. The boards in HATHI-2 are connected to each other in a static torus connection. This two-level interconnection scheme makes it possible to change the topology of the system by software, i.e. by controlling the C004 switches. For further details, see ref. [15] and references therein. This multiprocessor

[#1] Transputer is a trade mark of INMOS Ltd.

system can be configured or partitioned in several ways, depending on the need of the user. For the standard partitioning see refs. [15,16], sections 4.1 and 3.3.

The performance of the transputer system HATHI-2 is about 150 Mflops. The total memory capacity of the system is currently 125 MB, but this can be extended to 400 MB. The system is programmed in the high-level language Occam, but it also supports other high-level languages like Fortran or C. Experience gained from a number of application programs implemented on HATHI-2, (see ref. [17]) has shown that the most efficient use of this system is for problems where a large number of processes run (almost) independently of each other. Examples of such problems are Monte Carlo calculations, fluid dynamic calculations and multidimensional data analysis.

3. Implementation on HATHI-2

3.1. Structure of the modelled 3-fold data

To simulate the data and process this on the multiprocessor system, a simple level scheme is used. In this level scheme (see fig. 2) the labels for each transition are the full-energy address E^T (γ-ray energy), intensity I^T and transition number N_i, $i = x, y, z$. The properties of the transitions are given in table 1. In the present model we have simplified the γ-ray detector response function and presented it as a peak, called true (T), in address E^T with a certain height I^T, plus a constant "Compton background", called false (F), extending to the left of the peak randomly in all addresses $E^F < E^T$. The height of this background is chosen to be $\frac{1}{300}$ of that of the peak, i.e. $I^F = \frac{1}{300} I^T$. This value is very close to the values of the NORDBALL Compton-suppressed Ge + BGO modules [1]. The data so obtained then consists of the true 3-fold events x, y, z or clusters, arising from triple-photopeak events (T–T–T), whereas the rest of the events are in the background in the form of various photopeak (T) and Compton-tail (F) combinations, i.e. T–T–F, T–F–F and F–F–F, and their permutations. These have the geometrical shapes "tubes", "planes" and "cubes", respectively.

Table 2 gives some examples of how events arising from various combinations spread in the total space. The combinations of N_x–N_y–N_z chosen for this presentation purpose are those that have the properties of 1) highest intensity (1–2–3), 2) "most typical" (1–3–6), 3) largest background space (4–5–6) and 4) lowest intensity (10–11–12), thus covering a maximum range both in intensity and space. In the third column the relative intensity is obtained by multiplying the intensities given in table 1, and noting that $I^F = \frac{1}{300} I^T$, i.e. the intensity

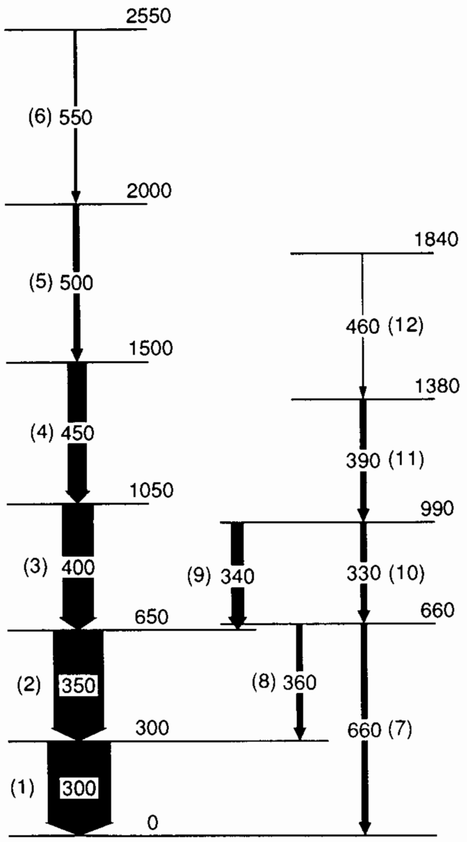


Fig. 2. A simple vibrational-type level scheme for generating the "data". The labels for each transition are energy and transition number. The properties of the transitions are listed in table 1, and the numbering and the intensities used in table 2 for the 3-fold coincidence intensities.

Table 1
Numbering of the energies, intensities and placements of the transitions shown in fig. 2

Number	Energy (E^T)	Intensity (I^T)	Placement
1	300	100	300 → 0
2	350	80	650 → 300
3	400	50	1050 → 650
4	450	30	1500 → 1050
5	500	10	2000 → 1500
6	550	5	2550 → 2000
7	660	10	660 → 0
8	360	10	660 → 300
9	340	20	990 → 650
10	330	10	990 → 660
11	390	10	1380 → 990
12	460	2	1840 → 1380

of the mentioned transition is divided by 300 for each F in the combination. The total number of addresses for a certain combination is the product $E_x E_y E_z$, where $E_i = 1$ for each T, and $E_j = E_j^T - 1$ for each F. (In the experiments widths other than 1 were used to test the identification routines, but this width does not affect the general results here.)

The figures of Table 2 can be used to elucidate some of the properties of the data intensities. First, comparing the strongest and weakest triples (1–2–3 and 10–11–12) we notice that the latter appear on average only once for 2000 of the former. This means that for one true combination of 10–11–12 there is about 1.5 of each of its false combinations, and more than two of its own "dust", F–F–F. At the same time there are 2000 true 1–2–3 triples, and 2000–3000 of each of its false combinations. The same argumentation holds for the other combinations of the present examples. In this way one may estimate the relative abundance of various kinds of events, and where they are diluted. Further, given a total number of events, i.e. the statistics of a real experimental situation, one may find out the total number of counts in various true cases – the desired data. Finally, the figure called P/C

Table 2
Sample frequencies (intensities) for 3-fold events generated from the transitions tabulated in table 1

Transition combination	Combination true/false	Relative intensity I_{rel}	Number of addresses N_{tot}	Total intensity $I_{rel}N_{tot}$	Shape of data [a]
1–2–3	T–T–T	400000	1	400000	cluster
	T–T–F	1333	399	531867	tube
	T–F–T	1333	349	465217	tube
	F–T–T	1333	299	398567	tube
	T–F–F	4.44	139251	618893	plane
	F–T–F	4.44	119301	530227	plane
	F–F–T	4.44	104351	463782	plane
	F–F–F	1.48×10^{-2}	4.16×10^{7}	616830	subcube
				$P/C = 0.133$ (51.0%) [b]	
1–3–6	T–T–T	25000	1	25000	cluster
	T–T–F	83.33	549	45750	tube
	T–F–T	83.33	399	33250	tube
	F–T–T	83.33	299	24917	tube
	T–F–F	0.28	219051	60848	plane
	F–T–F	0.28	164151	45598	plane
	F–F–T	0.28	119301	33139	plane
	F–F–F	9.26×10^{-4}	6.55×10^{7}	60645	subcube
				$P/C = 0.082$ (44.5%)	
4–5–6	T–T–T	1500	1	1500	cluster
	T–T–F	5	549	2745	tube
	T–F–T	5	499	2495	tube
	F–T–T	5	449	2245	tube
	T–F–F	0.0167	275000	4566	plane
	F–T–F	0.0167	247500	4108	plane
	F–F–T	0.0167	225000	3734	plane
	F–F–F	5.55×10^{-5}	1.23×10^{8}	6834	subcube
				$P/C = 0.056$ (38.3%)	
10–11–12	T–T–T	200	1	200	cluster
	T–T–F	0.67	459	306	tube
	T–F–T	0.67	389	259	tube
	F–T–T	0.67	329	219	tube
	T–F–F	2.22×10^{-3}	178551	397	plane
	F–T–F	2.22×10^{-3}	151011	336	plane
	F–F–T	2.22×10^{-3}	127981	284	plane
	F–F–F	7.41×10^{-6}	5.87×10^{7}	435	subcube
				$P/C = 0.089$ (44.7%)	

[a] We use the names "cluster" and "tube" here, although for the specific case of this table the clusters are in fact points, and the tubes lines.
[b] Figure in parenthesis is $100\,(P/C)^{1/3}\%$, which is the effective peak-to-Compton ratio per detector.

is the effective peak-to-Compton value for a specific combination of transitions, and as is seen it varies between some 5% and 13%. The percentage in parenthesis is the corresponding effective value for one detector ($=(P/C)^{1/3}$). These values vary, as expected, between 39% and 51%, well describing real detectors.

3.2. Data storage solution

Two different approaches for storage of the events from the experiment on a distributed-memory multiprocessor system were studied, a list-based structure and a bit map representation. Two possibilities have been considered:

1) If the subspaces are disjoint the processors are not aware of the events on the other side of the boarder between two subspaces. When investigating an event close to the boarder a request must be made to the neighbour for investigation of events on the other side of the boarder. On the other hand an event is only registered on one processor and redundancy can be avoided.

2) There can be a shared area between neighbouring processors. Since the radius of a cluster is approximately known, and it is within reasonable limits, this approach can be used. The common boarder area can be chosen to be the radius of a cluster, and a processor can analyse the surroundings of an event without knowing the events stored by another processor. This speeds up the analysation phase, although there will be some redundancy in the data storage. To minimize and simplify the communication between processors we have chosen the latter solution for the present implementation.

Two criteria are used to detect a cluster, either a high frequency of events registered in a single location, or a high proportion of events in the surrounding neighbourhood. The observations are stored in a two-dimensional array consisting of linked lists of observations. Each element in the array consists of the (x, y) coordinates of the observations and a pointer to a list of z-values. The elements in the z-list consist of the z-coordinate, a counter indicating the number of observations in this point and a pointer to the next observation with the same (x, y) coordinates. With this representation, we need one pointer for each (x, y) coordinate pair. For each observation, we need to store the z value, the counter and a pointer to the next observation, giving a total of 7 bytes per observation. For 10^8 observations, the maximum memory requirement will be about 764 MB. However, measurements show that the average number of observations for the points that occur in the observations is between 4 and 5, so the actual memory requirement can be reduce by a factor of 4, giving a total memory requirement of about 190 MB.

Because of the limited amount of memory in HATHI-2, the implementation was scaled down to a size of $400 \times 400 \times 4000$, and decomposed geometrically parallel.

The data domain is divided in equally large blocks in the (x, y) plane. Each processor keeps record of the observations falling inside its own domain, and whenever a new observation arrives, the processor inserts the observation into the appropriate z-list and checks by searching the z-lists of the neighbouring points if a cluster was formed. When a cluster is found, the observations belonging to this cluster are removed from the z-lists and only the position and intensity of the cluster is stored in a separate list. New observations are also checked against this cluster list to see if the observation falls into some already detected cluster. For details of the processes, see ref. [13].

3.3. File handling and sorting from disk

The present simulation runs were designed for maximum processor capacity, and since the data space is disjoint, it can use all processors. The task of each processor is to search a given data package for events containing the coordinates of its own data space. If found they are stored in the primary memory. The remaining events are sent to the next processor and the procedure is repeated until all data is sorted.

Since the task is to sort a large amount of data, the file handling turns out to be of major importance. It should be possible to read data from secondary memory, in this case a hard disk, with at least the same rate as the node processes are able to analyze it, otherwise all processors are not efficiently utilized. A local problem presently is that HATHI-2 does not have a secondary memory of its own, but uses the hard disk of the host computer. This is also a time-sharing unit, so efficient disk access also implies denying access to other users, which is not desirable. The present tests were made using the so-called folder method. This means that the data format on magnetic tape first must be translated to text format which is easiest to read. This conversion is impossible, however, in a real situation if high speed is needed.

4. Results and discussion

4.1. Sorting data on HATHI-2

The network used for testing the algorithms was a ring consisting of a number of node processors and one main processor. There is no disadvantage of such a simple solution if all the events are initially coming through one link from the experiment. This first link constitutes the constraint. In the network used the load is highest on the first link because all data have to use it. Eventually the data reaches the processor which administrates it and it will, if it is not needed by further processors, be removed from the data stream.

The system was first tested with the artificially generated data described in section 3.1. A separate processor was used to produce input data with the same distribution as data generated in the experiment. However, the random number generator used to produce the input data was unable to calculate more than about 3000 observations per second, so these tests could not be carried out for input rates higher than this. The program was able to process about 3000 observations per second.

The runs were made in two different ways:

- In the first one data was read from the Sun3 hard disk, transmitted to the HATHI-2 system and sorted on the individual transputers.
- The second consisted of sorting on the transputer system only, having the data ready in the local memories.

In this way the differences in performance of the two methods could be efficiently compared.

In the case of sorting the data from the hard disk, the tests were made with no other users on the Sun3 host. A total of 1 523 168 events were read from the disk in 731 s. This corresponds to a rate of 2084 events/s.

Doing both reading and sorting of the same amount of events, the following rates were observed: a package size of 1000 events was sorted in 965 s, and 5000 in 981 s. Conversely, this means, expressed as events/s, the following rates: 1578 cps at 1000 package size, and 1553 cps at 5000. As is seen, there is no big difference arising from the package size.

When sorting in memory only the data was first transferred to the primary memory of the transputers, and thereafter the sorting started. Here 500 events were sent to the prosessor network to be sorted 2000 times, that is, a total of 1 000 000 events. The sorting time was about 36 s which corresponds to 28 000 cps!

It was quite evident, and in fact expected, that the main bottleneck is at the transmission from the host to the transputer system. The present figures imply that if runs are made including reading-from-disk the optimal number of transputers is only 5, the rest of the capacity being idle!

4.2. Other improvements in data and space size

One of the main problems, which has not been tackled is this work, is the storage of the total data. This needs very large hard disk units, and constitutes a project on its own. Some recently proposed methods which address the question of data compression in order to reduce disk space will be addressed for the sake of completeness here. In ref. [18] the background subtraction and peak search in 3-fold gamma-event data was studied. It is based on removing "background" from triple-coincidence γ events, and was used to remove 40% of the noise without affecting photopeaks with intesities of > 18 counts. In the work of ref. [19] the compacting of high-spin γ coincidence data was discussed. It made use of a Monte Carlo event-wise unfolding of a measured response function, but the compaction is achieved at the expense of statistical quality in the medium energy region, and optimization is to restrict the energy range. The method is the same as used in ref. [20] to unfold 1- and 2-fold Ge spectra.

The work of ref. [21] represents one of the early attempts to analyze triple coincidence events. There the emphasis was laid on tracing rotational correlation properties. The data was reduced to fairly small cubes of 256 channels/dimension. The data was presented as different cuts in the volume, the transverse planes and rotational planes. One early method to improve on the peak-to-background ratio was formulated in ref. [22] where the entire background may be subtracted from the discrete lines in a γ–γ matrix. It is based on fitting the smooth background under the projections. The study in ref. [23] presents an algorithm which reduces the space required to store nuclear spectra, without loss of any information content. The basis of the procedure is very simple. The spectrum is divided into regions in which the content of the channels are all less than a power of 16, and the contents of each region can then be represented by a minimum number of bits. Tests indicated that spectrum files could be compressed by a factor of ~ 5.

The usual way of storing matrices is by a data field containing 4096×4096 elements, each 2 bytes, implying 32 MB of storage space. A cube with a side of 4096 elements would demand 11G bytes if the 6-fold symmetry and 1 byte per element is used. If the dispersion of the spectrum is changed so that the width of the peaks is kept constant all over the energy range, a considerable amount of space could be saved. This idea was first put forward in ref. [8]. The peak width of a Ge spectrum varies, for example, as

$$w(E) = \sqrt{w_0 + w_1 E + w_2 E^2},$$

where E is the γ-ray energy. The transformation needed to make the peak widths constant in a new dispersion $y(E)$ is given by $\mathrm{d}y(E) = \Gamma\, \mathrm{d}E/w(E)$ where Γ is a constant equal to the peak width in the transformed system. The relation $y(E)$ will be of the form $\sinh^{-1}$, which means an accelerated compression with increasing energy E. Another way of reducing space is to skip the first 80–90 keV of the spectrum, or to skip energies above 1800 keV. Thus, if one chooses to compress the spectrum and only handle the energy range of 80–1800 keV, a cube using 1 byte per channel would demand $139.5 \times \Gamma^3$ MB. If $\Gamma = 3.0$ this means 3.68 GB for the whole cube, or 0.61 GB if the 6-fold symmetry is used, and this actually fits on one standard laser disk. The above described method was also used in the study reported in ref. [24]. Finally, the work of ref. [12] analyses events up to 5-fold, but restricts itself to the structure of superdeformed bands, which by virtue of its regularity greatly reduces the problem.

5. Conclusions

We have demonstrated that a processor system of the HATHI-2 kind is well capable of analyzing triples γ-coincidence data, i.e. (x, y, z) coordinates, at collecting speed. As mentioned in section 2.1, the typical event rate at an experiment is 1000–2500 cps, and our tests show that in HATHI-2 about 1500–4000 cps can be analyzed. The rate is restricted by the communication with the hard disk.

A preliminary test to feed the transputer system with raw data from tape gave a result that was expected – the bottleneck is in the "main unit", i.e. the one that takes the data sends it on to the network. As the next step, the system was tested with real data from the experiment. In this case, the bottleneck proved to be the interface to the data files on the user host processor, the Sun3 workstation. Observations could be read from the disk at a maximum rate of about 1570 observations per second.

Further, some experiments were carried out to measure the performance of the system without any input or output. These tests were carried out by processing the same input data a large number of times, so that data had to be input only once. The results from this test indicated that the system could handle over 4000 observations per second, which, for three-dimensional events, corresponds to 18 kB/second.

Finally, if the data space can be reduced as described in subsection 4.2, i.e down to ~0.5 GB by nonlinear compression, it almost fits in the expandable core memory of HATHI-2. Hence full advantage of parallelization and the 150 Mflops computing power could be taken.

References

[1] B. Herskind, Nucl. Phys. A447 (1985) 395.
[2] G. Sletten, J. Gascon and J. Nyberg, Proc. Int. Conf. on Spectroscopy of Heavy Nuclei, Crete, Greece, 1989, Inst. Phys. Conf. Ser. 105 (1989) 125.
[3] D. Ward, Proc. Conf. on Nuclear Structure in the Nineties, ed. N.R. Johnson (Oak Ridge, April 23-27, 1990), Nucl. Phys. A520 (1990) 139c.
[4] EUROGAM Project Scientific Committee (ed.), Eurogam Proposal (Strasbourg, 1990).
[5] J. Gerl and R.M. Lieder (eds.), Upgrading to EUROBALL, Technical Report with Emphasis on Composite Encapsuled Ge Detectors (GSI, 1992).
[6] M.A. Deleplanque and R.M. Diamond (eds.), Gammasphere Proposal, Lawrence Berkeley Laboratory (1988).
[7] T. Lönnroth and P.P. Jauho, Nucl. Instr. and Meth. A261 (1987) 589.
[8] D.C. Radford et al., Workshop on Nuclear Structure, The Niels Bohr Institute, Copenhagen, 1988, p. 124.
[9] M. Bergström, Report LUNFD6/(NFFK-7110)1-38(1990) (Department of Cosmic and Subatomic Physics, University of Lund).
[10] M. Bergström and P. Ekström, Nucl. Instr. and Meth. A301 (1991) 132.
[11] W. Urban, W. Gast, R.M. Lieder, T. Rzaca-Urban, G. Hebbinghaus and A. Krämer-Flecken, Annual Report 1986 (Institut für Kernphysik, Kernforschungsanlage Jülich, 1987) p. 128.
[12] S. Flibotte, U.J. Hüttmeier, P. Bednarczyk, G. de France, B. Haas, P. Romain, Ch. Thiesen, J.P. Vivien and J. Zen, Nucl. Instr. and Meth. A320 (1992) 325.
[13] R.J.R. Back, M. Aspnäs, J. Granlund, J. Hattula, R. Julin, A. Lampinen, T. Lönnroth and P. Waxlax, Reports on Computer Science & Mathematics, Ser. A, no. 121 (Department of Computational Science, Åbo Akademi, 1991).
[14] Transputer Reference Manual, INMOS Limited (Prentice Hall, New York, 1988).
[15] M. Aspnäs, R.J.R. Back and T.-E. Malén, Microprocessors and Microsystems 14 (1990) 457.
[16] M. Aspnäs and T.-E. Malén, HATHI-2, User's Guide, version 1.0, Reports on Computer Science & Mathematics, Ser. B, no. 6 (1989).
[17] Multiprocessor Applications in the Hathi Project, Scientific Computing in Finland, eds. K. Kankaala and R. Nieminen, CSC Research Reports R1/89 (Centre for Scientific Computing, Espoo, 1989) ISSN 0787-7498, ISBN 951-47-3158-1.
[18] J. Junikka and A. Lampinen, Nucl. Instr. and Meth. A292 (1990) 677.
[19] D.J.G. Love, Nucl. Instr. and Meth. A302 (1991) 143.
[20] D.J.G. Love and A.J. Nelson, Nucl. Instr. and Meth. A274 (1989) 541.
[21] K. Mossberg P.-O. Forsgren, L. Hildingsson, W. Klamra and Th. Lindblad, Nucl. Instr. and Meth. A278 (1989) 755.
[22] G. Palameta and J.C. Waddington, Nucl. Instr. and Meth. A234 (1985) 476.
[23] J.F. Mika, L.J. Martin and P.N. Johnston, Nucl. Instr. and Meth. A295 (1990) 276.
[24] G. Szekely, Th. Lindblad, L. Hildingsson and W. Klamra, Nucl. Instr. and Meth. A292 (1990) 431.