1.34 Tflops Molecular Dynamics Simulation for NaCl with a Special-Purpose Computer: MDM

Tetsu Narumi, Ryutaro Susukita, Takahiro Koishi, Kenji Yasuoka[†],
Hideaki Furusawa, Atsushi Kawai, and Toshikazu Ebisuzaki

Computational Science Division,
Advanced Computing Center,
RIKEN (The Institute of Physical and Chemical Research)
Hirosawa 2-1, Wako, Saitama 351-0198, Japan
tel: +81-48-467-9417, fax: +81-48-467-4078
email: narumi@atlas.riken.go.jp

[†] Department of Mechanical Engineering, Keio University

Abstract

We performed molecular dynamics (MD) simulation of 9 million pairs of NaCl ions with the Ewald summation and obtained a calculation speed of 1.34 Tflops. In this calculation we used a special-purpose computer, MDM, which we are developing for the calculations of the Coulomb and van der Waals forces. The MDM enabled us to perform large scale MD simulations without truncating the Coulomb force. It is composed of WINE-2, MDGRAPE-2 and a host computer. WINE-2 accelerates the calculation for wavenumber-space part of the Coulomb force, while MDGRAPE-2 accelerates the calculation for real-space part of the Coulomb and van der Waals forces. The host computer performs other calculations. We performed MD simulation with the early version of the MDM system: 45 Tflops of WINE-2 and 1 Tflops of MDGRAPE-2. The peak performance of the final MDM system will reach 75 Tflops in total by the end of the year 2000.

Keywords: molecular dynamics simulation, special-purpose computer, Ewald method

1 Introduction

Molecular dynamics (MD) simulations have been widely used to study the physical properties of condensed matter at the atomic level. There are strong motivations to perform MD simulations with a large number of atoms to study large molecular systems, such as proteins or lipid bilayers. In MD simulations each atom is treated as a classical particle. An atom interacts with the other atoms through the Coulomb, van der Waals and bonding forces. Newton's equation of motion of atoms are solved by numerical integration. The most time consuming part is the calculation of Coulomb and Van der Waals forces. In particular, the calculation cost for the Coulomb force is

0-7803-9802-5/2000/$10.00 (c) 2000 IEEE.

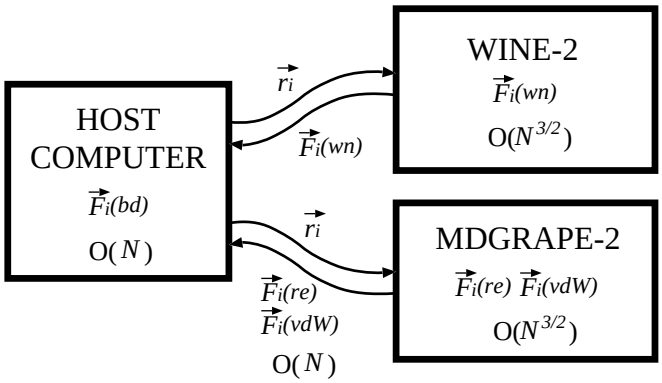

Figure 1: Basic structure of the Molecular Dynamics Machine

prohibitively high for systems larger than 10,000 atoms, since the Coulomb force is a long-range force; you cannot basically ignore the force from an atom no matter how far it is.

There are many algorithms to reduce the calculation cost for the Coulomb force. The Ewald method [1] is often used under the periodic boundary condition to reduce the calculation cost for the Coulomb force to $O(N^{3/2})$ instead of the native method's $O(N^2)$. Moreover, many other faster methods [2, 3, 4, 5] which scales as $O(N)$ or $O(N \log N)$ have been developed. However, the accuracy of these methods has not been well discussed on the actual system with large number of particles, such as 100,000 or 1,000,000.

In order to meet the huge computational requirements for the Coulomb force and to realize a large scale simulation with one million particles, we are developing a special purpose computer, the Molecular Dynamics Machine (MDM), based on the Ewald method. MDM is divided into three parts: WINE-2, MDGRAPE-2 and a host computer (figure 1). WINE-2 calculates the wavenumber-space part of the Coulomb force. MDGRAPE-2 calculates the real-space part of the Coulomb and van der Waals forces. The host computer performs the bonding force calculation and the other operations; for example, updating the positions and velocities of the particles. The peak speed of MDM will be about 75 Tflops and it will be completed by the end of the year 2000. The detailed description was given by Narumi et al. [6, 7].

Sugimoto *et al.* [8] have developed a series of hardwares called GRAPE, which is a special-purpose hardware designed to accelerate the gravitational force calculation. Their approach combines special-purpose and general-purpose computers and it has already been shown very effective. In fact, GRAPE-4 [9, 10] and GRAPE-5 [11] were awarded the Gordon Bell prize in 1995, 96 and 99. Their group also developed a special-purpose hardware for MD simulations, such as MD-GRAPE [12] and WINE-1 [13]. MDM is the successor of these machines.

We started a preliminary performance measurement of the system, since a part of MDM is ready for use in July 2000. We performed molecular dynamics simulation of 9 million pairs of NaCl ions and obtained a calculated speed of 1.34 Tflops. One of our target is to investigate the solid-liquid phase transition of ionic system with over million particles. The reason why we performed MD simulation of NaCl is that it is a typical ionic system and many experimental data are available.

MD simulations with very large number of particles benefits us in several points. The first

is, for example, the equilibrium properties of a system can be calculated with better accuracy in the case of large number of particles since the fluctuations of temperature and pressure are generally very small. Temperature and pressure computed for a system with smaller number of particles tend in fact to show larger fluctuations because they are calculated from a finite number of particles. Also in the case of phase transition, we need a very large number of particles to investigate properly the dynamic of the process. In the previous work [14], we performed 1 ns of solidification simulations with 13,824 particles of NaCl, and obtained small size of polycrystals which contain few grains. We therefore need much larger number of particles to get large size of polycrystals with many grains. In this respect, the simulation described in this paper was the preliminary test for the target runs, even though it was not the solid-liquid phase transition process but molten salt phase.

In the present paper we give a general description of the simulation and the results obtained. Section 2 presents the method of calculation adopted; section 3 gives an outline of the MDM hardware specifications while in section 4 is described the software for MDM. The results of molecular dynamics simulation as well as the measured speed are reported in section 5 and in section 6 we discuss the prospect of improving performance in the near future.

2 Calculation of Forces in the Ewald Method

In this section, we describe the Ewald method especially targeted on the number of floating point operations since we need to translate the performance of our hardware by a unit of floating point operations per second. First, we describe the basic equations of the Ewald method. Second, we discuss the number of floating point operations for them.

2.1 Basic equation

The force, $\vec{F}_i$, on a particle i is expressed as:

$$\vec{F}_i = \vec{F}_i(\mathrm{Clb}) + \vec{F}_i(\mathrm{vdW}) + \vec{F}_i(\mathrm{bd}),$$

where $\vec{F}_i(\mathrm{Clb})$ is the Coulomb force, $\vec{F}_i(\mathrm{vdW})$ is the van der Waals force, and $\vec{F}_i(\mathrm{bd})$ is the bonding force by covalent and hydrogen bonds.

The Coulomb force, $\vec{F}_i(\mathrm{Clb})$, on a particle i is divided into two parts: the real-space part, $\vec{F}_i(\mathrm{re})$, and the wavenumber-space part, $\vec{F}_i(\mathrm{wn})$, in the Ewald method:

$$\vec{F}_i(\mathrm{Clb}) = \vec{F}_i(\mathrm{re}) + \vec{F}_i(\mathrm{wn}). \tag{1}$$

Here, $\vec{F}_i(\mathrm{re})$ and $\vec{F}_i(\mathrm{wn})$ are expressed as

$$\vec{F}_i(\mathrm{re}) = \frac{q_i}{4\pi\varepsilon_0} \sum_j^{r_{ij}<r_{cut}} q_j \left[\frac{\mathrm{erfc}(\alpha r_{ij}/L)}{r_{ij}} + \frac{2\alpha}{\pi^{1/2}L} \exp(-\alpha^2 r_{ij}^2/L^2) \right] \frac{\vec{r}_{ij}}{r_{ij}^2}, \tag{2}$$

$$\vec{F}_i(\mathrm{wn}) = \frac{q_i}{2\pi\varepsilon_0 L^3} \sum_{\vec{k}}^{k<k_{cut}} \frac{\vec{k}}{k^2} \exp(-\pi^2 L^2 k^2/\alpha^2) \times \left[\sin(2\pi\vec{k}\cdot\vec{r}_i) \sum_j q_j \cos(2\pi\vec{k}\cdot\vec{r}_j) - \cos(2\pi\vec{k}\cdot\vec{r}_i) \sum_j q_j \sin(2\pi\vec{k}\cdot\vec{r}_j) \right], \tag{3}$$

where $\vec{r}_i$ is the position of particle i, $\vec{r}_{ij}$ is the relative vector from particle j to particle i, r_{ij} is the distance between particles i and j, q_i is the electrostatic charge of particle i, $\vec{k}$ is the wavenumber vector, k is the length of $\vec{k}$, L is the length of a side of the computational box, r_{cut} and k_{cut} are the cut-off lengths of relative vector and wavenumber vector, respectively, ε_0 is the dielectric constant of vacuum, α is a parameter to balance the computational cost for the real and wavenumber parts of the Coulomb force calculation, and erfc(x) is the complementary error function.

The van der Waals force, $\vec{F}_i$(vdw), on particle i is calculated as follows when the potential function is Lennard-Jones:

$$\vec{F}_i(\mathrm{vdW}) = \sum_{j}^{r_{ij}<r_{cut}} \varepsilon(at_i, at_j) \left\{ 2 \left[\frac{\sigma(at_i, at_j)}{r_{ij}} \right]^{14} - \left[\frac{\sigma(at_i, at_j)}{r_{ij}} \right]^{8} \right\} \vec{r}_{ij}, \tag{4}$$

where $\varepsilon(at_i, at_j)$ and $\sigma(at_i, at_j)$ are parameters which are determined by the particle types, at_i and at_j. at_i is the particle type of particle i.

In the following subsections, we describe how many floating-point operations are needed for the real-space and wavenumber-space parts, respectively.

2.2 Real-space part

The number of floating-point operations between a pair of particles in the real-space part of the Coulomb force (equation 2) is estimated as 59, including one complementary error function, one exponential function, one square root function, one division, ten multiplications, six additions, and three subtractions. Here, we assumed the number of floating-point operations for erfc(x), exp(x), sqrt(x) and division to ten, respectively. The number of floating-point operations per time-step for the real-space part is calculated as $59NN_{int}$, where N_{int} is the number of particles to be taken into account in the force calculation exerted on one particle. It is half number of particles within the cut-off radius, r_{cut}:

$$N_{int} \simeq \frac{1}{2} \cdot \frac{4}{3} \pi r_{cut}^3 \left(\frac{N}{L^3}\right), \tag{5}$$

since they use Newton's third law in the calculation of the force.

However, the number of particles, N_{int_g}, to be taken into account on one particle differs from N_{int} when we use MDGRAPE-2 as:

$$N_{int\_g} \simeq 27 r_{cut}^3 \left(\frac{N}{L^3}\right). \tag{6}$$

N_{int_g} is about 13 times larger than N_{int}, which means that MDGRAPE-2 performs 13 times more operations for calculating the same force. This is caused by two reasons. First, MDGRAPE-2 does not use Newton's third law since the logic to support it makes the hardware complicated.

Second, MDGRAPE-2 uses cell-index (link-cell) method [15] to search the neighboring particles instead of comparing the distance of each particle. A simulation box is divided into cells, and particles are allocated to them. We can get the neighboring particles by searching only neighboring cells. The calculation cost for searching neighboring cells is very small compared with that for directly searching neighboring particles. In the case of MDGRAPE-2, we set

the size of a cell to a little larger than r_{cut}. So a particle in a cell interact with particles in neighboring 27 cells including itself; equations 2 and 4 can be rewritten as:

$$\vec{F}_i(\mathrm{re}) = \frac{q_i}{4\pi\varepsilon_0} \sum_{c=1}^{27} \sum_{j=jstart_c}^{jend_c} \vec{f}_{i,j}(\mathrm{re}), \tag{7}$$

$$\vec{F}_i(\mathrm{vdW}) = \sum_{c=1}^{27} \sum_{j=jstart_c}^{jend_c} \vec{f}_{i,j}(\mathrm{vdW}). \tag{8}$$

Here c is the index of a neighboring cell, $jstart_c$ is the first index of a particle in cell c, $jend_c$ is the last index of a particle in cell c, $\vec{f}_{i,j}(\mathrm{re})$ and $\vec{f}_{i,j}(\mathrm{vdW})$ are pairwise Coulomb and van der Waals forces between particles i and j, respectively. We assumed that the indices of particles in a cell are contiguous. The disadvantage of cell-index method with MDGRAPE-2 is the increase of calculation cost for force calculation. MDGRAPE-2 does not skip the force calculation even if the distance between two particles are larger than r_{cut}, since the logic to skip it is also very complicated.

Therefore, we use N_{int_g} instead of N_{int} for MDGRAPE-2. Readers may suspect that we would increase the number of floating point operations unreasonably. Is is not true because we correct this increase when evaluating the effective performance of our machine in the later section.

2.3 Wavenumber-space part

The wavenumber-space part of the Coulomb force, $\vec{F}_i(\mathrm{wn})$, is actually calculated with WINE-2 by the following two steps. In the first step, it performs discrete Fourier transform (DFT) as:

$$S_n = \sum_{j=1}^{N} q_j \sin(2\pi \vec{k}_n \cdot \vec{r}_j), \tag{9}$$

$$C_n = \sum_{j=1}^{N} q_j \cos(2\pi \vec{k}_n \cdot \vec{r}_j), \tag{10}$$

where N is the number of particles. In the second step, WINE-2 performs inverse DFT (IDFT) using S_n and C_n as:

$$\vec{F}_i(\mathrm{wn}) = \frac{q_i}{\pi\varepsilon_0 L^3} \sum_{n}^{N_{wv}} a_n \left[C_n \sin(2\pi \vec{k}_n \cdot \vec{r}_i) - S_n \cos(2\pi \vec{k}_n \cdot \vec{r}_i) \right] \cdot \vec{k}_n, \tag{11}$$

where a_n is

$$a_n = \frac{\exp(-\pi^2 L^2 k_n^2/\alpha^2)}{k_n^2}, \tag{12}$$

and N_{wv} is half of the number of wavenumber vectors whose lengths are shorter than k_{cut}:

$$N_{wv} \simeq \frac{1}{2} \cdot \frac{4}{3}\pi L^3 k_{cut}^3. \tag{13}$$

The number of floating-point operations between one particle and one wave (wavenumber) in a DFT (equations 9 and 10) is estimated as 29, including one sine, one cosine, five multiplications

and four summations. Here, we assumed the number of floating-point operations for $\sin(x)$ or $\cos(x)$ corresponds to ten. On the other hand, the number of floating-point operations in IDFT (equation 11) is estimated as 35 including one sine, one cosine, 9 multiplications, 5 summations, and 1 subtraction. In summary, the number of floating-point operations per time-step for the wavenumber-space part is calculated as $64NN_{wv}$. Note that the other calculations in the wavenumber-space part (e.g., equation 12) are negligibly small compared with DFT and IDFT for large N and N_{wv}.

One may point out that we can use addition formula to reduce the floating point operations for equations 9, 10 and 11. However, we need $6NLk_{cut} \times 8$byte of storage, which is very large for large N.

3 MDM System

3.1 Architecture

The MDM system is composed of three parts, WINE-2, MDGRAPE-2, and a host computer (see figure 1). The two special-purpose computers, which we are developing, work as a back-end processors. The host computer is a general-purpose computer, and works as a front-end processor. WINE-2 calculates the wavenumber-space part of the Coulomb force. MDGRAPE-2 calculates the real-space forces. The host computer performs short range force calculation and other operations.

The basic calculation flow in one time-step of MD simulation is as follows. First, the host computer sends the coordinates of particles to WINE-2 and MDGRAPE-2. Second, WINE-2 calculates the Coulomb force from wavenumber-space, and MDGRAPE-2 calculates the Coulomb force from real-space and van der Waals force. Third, the host computer receives the forces on particles from WINE -2 and MDGRAPE-2. Forth, the host computer performs other operations; bonding force calculation, time integration of particles, file I/O, etc.

The calculation cost on two special-purpose computers scales as $O(N^{3/2})$, while that on the host computer and the communication between them scale as $O(N)$. Therefore, we can accelerate the MD simulation by accelerating only the long-range force calculation in the case of large N; the host computer and the communication do not cause the bottleneck of the system.

3.2 Basic structure

The current MDM system, which is under construction, is shown in figure 3. The host computer is composed of four node computers, and they are connected with each other by a network. Each node computer has 5 WINE-2 clusters and 4 MDGRAPE-2 clusters via links. Each WINE-2 cluster has 7 WINE-2 boards connected by a bus. Each MDGRAPE-2 cluster has 2 MDGRAPE-2 boards connected by a bus. Table 1 shows the components that we used.

WINE-2 is composed of 2,240 WINE-2 chips and has a peak speed of 45 Tflops. MDGRAPE-2 is composed of 64 MDGRAPE-2 chips and has a peak speed of 1 Tflops. In the following sections, we describe the hardware in detail.

Table 1: Components of the MDM system

Component	Product	Manufacturer
Node computer	Enterprise 4500	Sun Microsystems
CPU	Ultra SPARC-II 400 MHz	
Network	Myrinet	Myricom
Switch	16-port LAN switch	
Network card	LAN PCI card (LANai 4.3)	
Link	Bus bridge	SBS Technologies
Interface card	PCI host card/(Compact)PCI backplane controller card	
Bus	CompactPCI (WINE-2) / PCI (MDGRAPE-2)	PCI local bus spec. rev. 2.1

3.3 Host computer

The host computer is composed of four node computers. As a node computer, we used a Sun Enterprise 4500 server which has 6 Ultra SPARC II processors of 400 MHz clock. Myrinet [16] network connects the node computers with each other.

3.4 WINE-2

WINE-2 (figure 4a) is composed of 20 WINE-2 clusters. Each WINE-2 cluster has 7 WINE-2 boards. A WINE-2 board (figure 4b) has 16 WINE-2 chips (figure 4c).

3.4.1 WINE-2 cluster

A WINE-2 cluster is composed of 7 WINE-2 boards which are connected via CompactPCI bus. It is connected to a node computer through PCI-CompactPCI bus bridge. Therefore, a WINE-2 board looks like a normal PCI device from the host computer.

3.4.2 WINE-2 board

A WINE-2 board (figure 5) is composed of 16 WINE-2 chips, the `interface logic`, `particle index counter`, and `particle memory`. A node computer communicates with WINE-2 chips and `particle memory` through `interface logic`. A WINE-2 board has a 6U CompactPCI standard form factor, except that its depth is 40cm instead of 16cm. The `interface logic` and `particle index counter` are integrated in an FPGA (XC4062XLA by Xilinx). 16 Mbyte of SDRAM is used for the `particle memory`.

3.4.3 WINE-2 chip

A WINE-2 chip (figure 6) has eight identical WINE-2 pipelines. Peak performance of a WINE-2 chip corresponds to about 20 Gflops at a clock frequency of 66.6 MHz. It is made by LSI logic

K.K. with LCB500K technology, whose design rule is 0.5μm and operation voltage is 3.3V. The number of transistors of a WINE-2 chip is 1.2 million.

3.4.4 WINE-2 pipeline

A WINE-2 pipeline performs DFT (equations 9 and 10) in DFT mode and IDFT (equation 11) in IDFT mode to calculate the wavenumber-space part of the Coulomb force.

Figure 7 shows the structure of the pipeline in DFT mode. The pipeline calculates the inner product of vectors $\vec{r}_j$ and $\vec{k}_n$, and then calculates their *sine* and *cosine*. It multiplies the results to q_j, and accumulates the products to get $S_n + C_n$ and $S_n - C_n$. The host computer calculates S_n and C_n from $S_n + C_n$ and $S_n - C_n$. Wavenumber vectors ($\vec{k}_n$) are loaded into a pipeline before starting the calculation. The pipeline in the IDFT mode is similar to that in DFT mode.

Fixed-point two's complement format is used in all the arithmetic calculations in a pipeline. The relative accuracy of $\vec{F}_i$(wn) is about $10^{-4.5}$, which is accurate enough for MD simulations. Researchers usually truncate the accumulation with the Ewald method at a relative error of 10^{-3} to 10^{-5}. In actual cases, $\vec{F}_i$(wn) is several times smaller than $\vec{F}_i$(re). The error in $\vec{F}_i$(wn) is smaller than either that of $\vec{F}_i$(re) or the truncation error of the Ewald sum. Therefore, the accuracy of the pipeline is enough for usual MD simulations.

3.5 MDGRAPE-2

MDGRAPE-2 (figure 8a) consists of 16 MDGRAPE-2 clusters. Each MDGRAPE-2 cluster has two MDGRAPE-2 boards. An MDGRAPE-2 board (figure 8b) has two MDGRAPE-2 chips (figure 8c).

3.5.1 MDGRAPE-2 cluster

An MDGRAPE cluster is composed of two MDGRAPE-2 boards which are connected via PCI bus. It is connected to a node computer through PCI-PCI bus bridge. Therefore, an MDGRAPE-2 board also looks like a normal PCI device from the host computer.

3.5.2 MDGRAPE-2 board

An MDGRAPE-2 board (figure 9) is composed of two MDGRAPE-2 chips, `interface logic`, `cell index counter`, `cell memory`, `particle index counter`, and `particle memory`. The basic function of the board is the same as WINE-2 board except that the index of a particle is determined by dual counters to support cell-index method (equations 7 and 8). `Cell index counter` specifies the neighboring cell index c, and `cell memory` outputs the range of indices in the cell c. `Particle index counter` indicates the particle index j to `particle memory`. The position, charge, and particle type of a particle j are supplied to both of the MDGRAPE-2 chips. An MDGRAPE-2 board is a PCI long size card. `Interface logic`, `cell index counter`, `cell memory`, `particle index counter` are integrated in an FPGA (FLEX10K100ABC600-1 by Altera). 8 Mbyte of SSRAM is used for the `particle memory`.

3.5.3 MDGRAPE-2 chip

An MDGRAPE-2 chip (figure 10) has four identical MDGRAPE-2 pipelines, `atom coefficient RAM`, and `neighbor list RAM`. `Atom coefficient RAM` stores the coefficients $\sigma(at_i, at_j)$ and $\varepsilon(at_i, at_j)$. The maximum number of particle types is 32, which is enough for MD simulation with proteins. `Neighbor list RAM`, which was not used in our simulation, can be used to search neighboring particles. Peak performance of an MDGRAPE-2 chip corresponds to about 16 Gflops at a clock frequency of 100 MHz. It is made by IBM with SA-12 technology, whose design rule is 0.25μm and operation voltage is 2.5V. The number of transistors of an MDGRAPE-2 chip is 5 million.

3.5.4 MDGRAPE-2 pipeline

A MDGRAPE-2 pipeline calculates the pairwise force $\vec{f}_{i,j}$ as:

$$\vec{f}_{i,j} = b_{ij} g(a_{ij} r_{ij}^2) \vec{r}_{ij}, \tag{14}$$

where $g(x)$ is an arbitrary central force, and a_{ij} and b_{ij} are coefficients determined by particle types of particles i and j.

Figure 11 shows the structure of the pipeline. The pipeline calculates $\vec{r}_{ij}$ and $a_{ij} r_{ij}^2$, and then evaluate $g(x)$ in the `function evaluator`. `Function evaluator` performs fourth-order interpolation segmented by 1,024 region. The coefficients of the interpolation function are stored in the RAM in `function evaluator`. Therefore, we can use any arbitrary central force by changing the contents of the RAM. After the `function evaluator`, the pipeline multiplies b_{ij} and $\vec{r}_{ij}$, and then accumulates them. The relative accuracy of a pairwise force is about 10^{-7}, since most of the arithmetic units in the pipeline use IEEE754 single floating point format. The double floating point format is used for accumulating the force in order to prevent the underflow when large number of particles are used.

In the case of real-space part of the Coulomb force, $\vec{f}_{i,j}(\mathrm{re})$ in equation 7 is calculated as:

$$\begin{aligned} g(x) &= \frac{2\exp(-x)}{\pi^{1/2} x} + \frac{\mathrm{erfc}(x^{1/2})}{x^{3/2}}, \\ a_{ij} &= \alpha^2 L^{-2}, \\ b_{ij} &= q_j. \end{aligned}$$

In the case of van der Waals force, $\vec{f}_{i,j}(\mathrm{vdW})$ in equation 8 is calculated as:

$$\begin{aligned} g(x) &= 2x^{-7} - x^{-4}, \\ a_{ij} &= \sigma(at_i, at_j)^{-2}, \\ b_{ij} &= \varepsilon(at_i, at_j). \end{aligned}$$

4 Software

We developed MD program written in C for MDM, which is parallelized with Message Passing Interface (MPI). We used 16 processes for real-space part, and 8 processes for wavenumber-part. The simulation box is divided into 16 domains, and one process for real-space part performs all

the calculation in each domain except wavenumber-space part. The difference of the program when we use MDM is that we call library routines to calculate real-space and wavenumber-space forces instead of calling internal force subroutines. Tables 2 and 3 include all the library routines that were called from the MD program.

For wavenumber-space part, the library routine for force calculation is already parallelized with MPI, and users do not care any communication between processes. We used 8 processes for wavenumber-space part, so each of them has about $N/8$ particle positions. All the processes call WINE-2 library routines with the same parameters except the force calculation routine. In the force calculation routine, the positions of particles are different from each other.

For real-space part, communication between processes must be done by user. We divide a simulation box to 16 domains, and each process calculates real-space part forces in its domain ($N/16$ particles) by calling `MR1calcvdw_block2` once. Note that each process should know positions of neighboring particles before calling `MR1calcvdw_block2`, that is what you have to manage with MPI routines. The function table for $g(x)$ is generated beforehand by a separate utility program, and loaded to MDGRAPE-2 chips at the beginning of the simulation by calling `MR1SetTable`.

Table 2: Library routine for WINE-2

Category	Name	Function
Initialization	`wine2_set_MPI_community`	set the MPI community for wavenumber-space part
	`wine2_allocate_board`	set the number of WINE-2 boards to acquire
	`wine2_initialize_board`	acquire WINE-2 boards
Force calculation	`wine2_set_nn`	set the number of particles for each process
	`calculate_force_and_pot_wavepart_nooffset`	calculate the wavenumber-space part of force
Finalization	`wine2_free_board`	release WINE-2 boards

Table 3: Library routine for MDGRAPE-2

Category	Name	Function
Initialization	`MR1allocateboard`	set the number of MDGRAPE-2 boards to acquire
	`MR1init`	acquire MDGRAPE-2 boards
	`MR1SetTable`	set the function table $g(x)$
Force calculation	`MR1calcvdw_block2`	calculate the real-space part of force with cell-index method
Finalization	`MR1free`	release MDGRAPE-2 boards

Table 4: Performance of simulation

System	MDM current	Conventional system	MDM future
N	1.88×10^7		
α	85.0	30.1	50.3
r_{cut}	26.4	74.4	44.5
Lk_{cut}	63.9	22.7	37.9
N_{int}	-	2.65×10^4	-
N_{int_g}	1.52×10^4	-	7.32×10^4
N_{wv}	5.46×10^5	2.44×10^4	1.14×10^5
Number of floating-point operations for the real-space part	$59NN_{int\_g}$	$59NN_{int}$	$59NN_{int\_g}$
	1.69×10^{13}	2.94×10^{13}	8.13×10^{13}
Number of floating-point operations for the wavenumber-space part	$64NN_{wv}$		
	6.58×10^{14}	2.94×10^{13}	1.37×10^{14}
Total number of floating-point operations per time-step	6.75×10^{14}	5.88×10^{13}	2.18×10^{14}
sec/step	43.8		4.48
Calculation speed (Tflops)	15.4	1.34	48.7
Effective speed (Tflops)	1.34		13.1

5 MD Simulation

We performed 3,000 time-steps of a canonical ensemble (NVT constant) MD-simulation of 9,410,548 pairs of NaCl ions (18,821,096 particles) for 36.5 hours (131,443 seconds). We obtained an effective calculated speed of 1.34 Tflops as shown in table 4.

We adopted Tosi-Fumi potential [17] as a force field between ions:

$$\phi(r) = \frac{q_i q_j}{r} + A_{ij} b \exp\left(\frac{\sigma_i + \sigma_j - r}{\rho}\right) - \frac{c_{ij}}{r^6} - \frac{d_{ij}}{r^8}, \tag{15}$$

where A_{ij}, b, σ_i, σ_j, ρ, c_{ij} and d_{ij} are parameters. We used a time-step of 2 fsec and a temperature of 1200 K. The cut-off length, r_{cut}, of the real-space part of the Coulomb and other forces is 26.4 Å. The length, L, of a side of the simulation box is 850 Å. The non-dimensional cut-off length in the wavenumber-space part of the Coulomb force is set to be $Lk_{cut} = 63.9$.

We performed other two small simulations; 1,481,544 and 110,592 particles, respectively. Figure 2a shows the temperature (Kelvin) calculated during the simulation plotted versus the time-step (ps) in the case of $N = 1.88 \times 10^7$ particles. For comparison, in figures 2b and 2c are plotted the same parameters as obtained using a lower number of particles, respectively $N = 1.48 \times 10^6$ (figure 2b) and $N = 1.10 \times 10^5$ (figure 2c). As clearly shown, the fluctuation of the the temperature becomes smaller as the number of particles becomes larger, confirming the necessity of using very large number of particles in order to characterize with good accuracy the physical-chemical property of a system of interacting ions.

In any case, the first 2,000 time-steps (0 - 4 ps) are NVT constant ensemble by scaling the velocity and the last 1,000 time-steps (4 - 6 ps) are NVE constant ensemble. The total energies

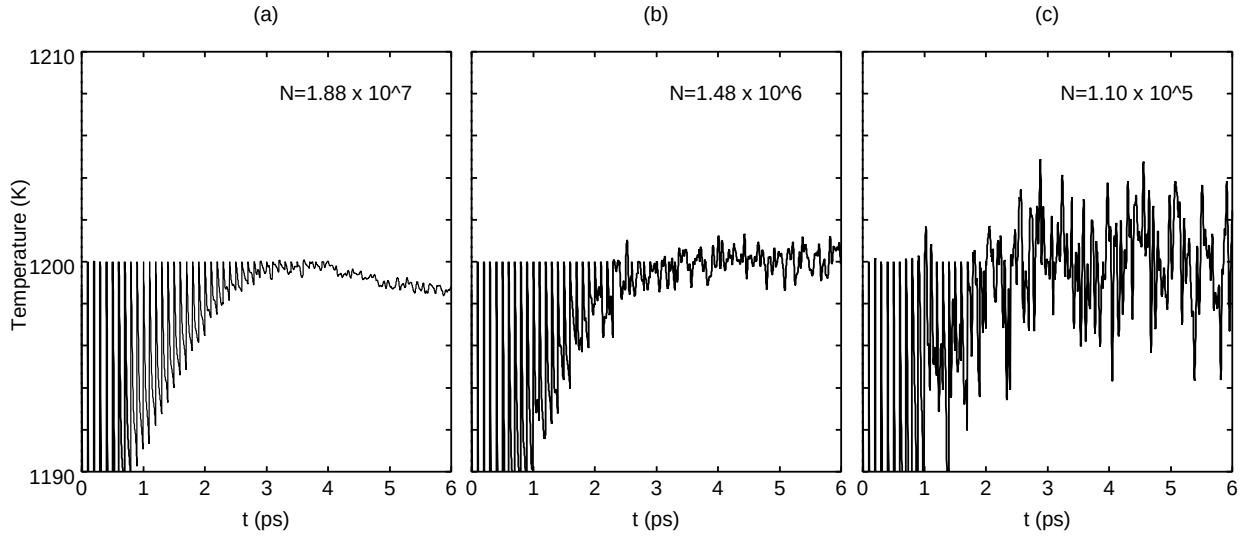

Figure 2: Temperature fluctuation against time-steps (ps) respectively for $N = 1.88 \times 10^7$ (a), $N = 1.48 \times 10^6$ (b) and $N = 1.10 \times 10^5$ (c)

are well conserved; relative error of the total energy is less than 5×10^{-5} percent in the case of $N = 1.88 \times 10^7$. The gradual decrease of the temperature in figure 2a is probably caused by the shortage of the time-steps for NVT ensemble. In the initial condition the particles are in the crystal state whose potential energy is lower than that of liquid state. It seems that 2,000 time-steps is not enough to make the configuration of particles to the liquid state. If we perform longer simulation with scaling velocity, the temperature would not decrease any more.

The second column of table 4 summarizes the floating-point operations per time-step and performance with MDM. The number of floating-point operations per time-step is calculated as $59NN_{int\_g} + 64NN_{wv}$ (see section 2). Here we note that the force calculation other than the Coulomb is excluded because the calculation cost for them is not so large. Moreover, the potential energy calculation, which is performed every 100 time-steps, is also excluded for the same reason. Most of the floating point operations are included for wavenumber-space part of the Coulomb force calculation, because we adopted very large $\alpha = 85.0$. This value is optimized for our hardware system, which itself has especially high performance in the calculation of wavenumber-space part. The average calculation speed is 15.4 Tflops, since one time-step took 43.8 seconds.

We point out that if we perform the same simulation on a conventional general-purpose computer, we would need only about 10 times smaller number of floating-point operations with the same Ewald method's accuracy since the optimal α would be 30.1 not 85.0. The third column of table 4 shows, for comparison, the number of floating point operations and the performance, that a conventional general-purpose computer with the same effective performance as MDM would provide. The reason that optimal α is small for a conventional computer is that the speed for the real-space and that for wavenumber-space are almost the same in a conventional system in contrast to our system. The number of floating point operations for real-space and wavenumber-space parts are made the same ($59NN_{int} = 64NN_{wv} = 2.94 \times 10^{13}$), which means

that $\alpha = 30.1$ is best for reducing the total number of floating point operations. It must be noted that the fast algorithm which use addition formula (see section 2.3) is not practical to reduce the floating point operations for equations 9, 10 and 11 since the required data storage for it exceeds 20 Gbyte.

Therefore, we have to make correction to get the effective performance of the MDM. The lowest number of floating point operations per time-step is 5.88×10^{13} with the same accuracy of simulation as that performed with the MDM, and one time-step took 43.8 seconds. So the effective performance of the MDM is 1.34 Tflops instead of 15.4 Tflops.

6 Discussions

6.1 Performance improvement

When considering the result of our simulation test we notice that the value obtained for the speed is sensibly lower when compared with the 46 Tflops peak speed of the system. The difference between the peak and the obtained performance can be explained in terms of the following considerations, listed below together with the modifications, partially in progress, planned in order to furtherly improve the performance of our system.

1. The miss-balance of the calculation speed between WINE-2 and MDGRAPE-2 reduces the effective performance by a factor of ten. To overcome this point we plan to increase MDGRAPE-2 chips to 1,536 by the end of 2000 so that the performance will be comparable to WINE-2.
2. The communication between a node computer and the WINE-2 boards or MDGRAPE-2 boards is slow when compared with the calculation speed of the boards. We therefore plan to increase this bandwidth by a factor of two with 64-bit PCI-bus instead of 32-bit one.
3. The communication among node computers is slow compared with WINE-2 or MDGRAPE-2. As a consequence, we plan to increase this bandwidth by a factor of three with new Myrinet network cards.

Table 5 compares the current and future versions of MDM. The number of chips will be increased. The last two ranks show the efficiency of MDGRAPE-2 and WINE-2. The efficiencies of current version of MDM were calculated from the MD simulation described in the last section, while those of future version of MDM are roughly estimated from the increase of the performance of the host computer and the communication. The forth column of table 4 shows that future version of MDM will reach 13.1 Tflops. Note that the number of floating point operations for real-space and wavenumber-space parts are almost the same, which means that MDGRAPE-2 and WINE-2 are well balanced. The gap between calculation speed (48.7 Tflops) and the effective speed (13.1 Tflops) is caused by the difference bewteen N_{int} and N_{int_g} (see section 2). We already have a project to decrease this difference with small hardware modification.

6.2 MD simulation with the future version of MDM

The total MDM will be completed at the end of 2000. MDM should take 0.19 seconds per time-step for MD simulations with a million particles using the Ewald method. Accordingly, it should take only one week ($\sim 6.0 \times 10^5$ s) to perform the simulations for a time span of 1.6 ns ($\sim 3.2 \times 10^6$ time-steps). We believe that the very large-scale MD simulations possible with

Table 5: Comparison of current and future versions of MDM

System	Current	Future
Number of MDGRAPE-2 chips	64	1,536
Number of WINE-2 chips	2,240	2,688
Peak performance of MDGRAPE-2 (Tflops)	1	25
Peak performance of WINE-2 (Tflops)	45	54
Efficiency of MDGRAPE-2 (%)	26	50
Efficiency of WINE-2 (%)	29	50

MDM will lead to major breakthroughs, such as the understanding of the function of protein molecules as enzymes. For example, when it will be possible to perform MD simulation with very large number of particles in a rate of video (30 frame/sec), we will be able to interact with the molecule via computer by means of a virtual reality technology.

6.3 Comparison between Ewald and other methods

One of the purpose of our hardware is to investigate the accuracy and speed of the Ewald summation compared with other fast methods. An important result is that we can accelerate fast methods with MDGRAPE-2. In fact, Makino *et al.* [18] performed gravitational calculation with tree-code, one of a major $O(N \log N)$ method, and found that GRAPE machine can accelerate tree-code. If we use tree-code with MDM, we can not only compare the accuracy with Ewald method but also perform larger simulation that cannot be done with Ewald method.

6.4 Other applications

MDM can be used for other applications, such as cosmological simulation [13], Smoothed Particle Hydrodynamics (SPH) [19, 20], and vortex dynamics simulation [21]. There are many application even though MDM is a special-purpose computer designed for MD simulation.

Acknowledgment

This work is supported by the fund for "Heterogeneous Computer System" of the Science and Technology Agency of Japan.

References

[1] Ewald, P.P., *Ann. Phy.*, **64**, 253 (1921).

[2] Barnes, J. and Hut, P., A Hierarchical $O(N \log N)$ Force-calculation Algorithm, *Nature*, **324**, pp. 446-449 (1986).

[3] Greengard, L. and Rokhlin, V., A Fast Algorithm for Particle Simulations, *J. Comput. Phys.*, **73**, pp. 325-348 (1987).

[4] Essmann, U., Perera. L., and Berkowitz, L., A Smooth Particle Mesh Ewald Method, *J. Chem. Phys.*, **103**, pp. 8577-8593 (1995).

[5] Strain, J., Fast Potential Theory II: Layer Potentials and Discrete Sums, *J. Comput. Phys.*, **99**, pp. 251-270 (1992).

[6] Narumi, T. et al., Molecular Dynamics Machine: Special-purpose Computer for Molecular Dynamics Simulations, *Mol. Sim.*, **21**, pp. 401-415 (1999).

[7] Narumi, T. et al., 46 Tflops Special-purpose Computer for Molecular Dynamics Simulations: WINE-2, in *Proceedings of the 5th International Conference on Signal Processing*, Beijing, (2000), in press.

[8] Sugimoto, D., Chikada, Y., Makino, J., Ito, T., Ebisuzaki, T., and Umemura, M., A Special Purpose Computer for Gravitational Many-body Problems, *Nature*, **345**, pp.33-35 (1990).

[9] Makino, J. and Taiji, M., Astrophysical N-body simulations on GRAPE-4 Special-Purpose Computer, in *Proceedings of Supercomputing '95*, IEEE Computer Society Press, Los Alamitos (1995).

[10] Fukushige, T. and Makino, J., N-body Simulation of Galaxy Formation on GRAPE-4 Special-Purpose Computer, in *Proceedings of Supercomputing '95*, IEEE Computer Society Press, Los Alamitos (1995).

[11] Kawai, A., Fukushige, T., and Makino, J., \$7.0/Mflops Astrophysical N-body Simulation with Treecode on GRAPE-5, in *Proceedings of Supercomputing '99* (1999).

[12] Taiji, M., Fukushige, T., Makino, J., Ebisuzaki, T., and Sugimoto, D., MD-GRAPE: A Parallel Special-Purpose Computer System for Classical Molecular Dynamics Simulations, *Physics Computing '94* Lugano, Switzerland, in *Proceedings of the 6th Joint EPS-APS international conference on Physics Computing*, European Physical Society, Geneva, pp. 200-203 (1994).

[13] Fukushige, T., Makino, J., Ito, T., Okumura, S., Ebisuzaki, T., and Sugimoto, D., WINE-1: Special-Purpose Computer for N-body Simulations with a Periodic Boundary Condition, *Publs. Astron. Soc. Japan*, **45**, pp. 361-375 (1993).

[14] Koishi, T., Yasuoka, K., Narumi, T., Susukita, R., Furusawa, H., and Ebisuzaki, T., MD simulation of solid-liquid phase transition for NaCl-KCl mixture with a special purpose computer (MDM), *CCP2000: Conference on Computational Physics 2000*, Queensland, Australia (2000).

[15] Hockney, R. W., and Eastwood, J. W., Computer Simulation Using Particles, New York, McGraw-Hill (1981).

[16] Myricom, Inc., `http://www.myri.com/`

[17] Tosi, M. P. and Fumi, F. G., *J. Phys. Chem. Solids*, **25**, p. 45, (1964).

[18] Makino, J., Treecode with a Special-Purpose Processor. *Astrophysical Journal*, **369**, p. 200 (1991).

[19] Umemura, M., *ApJ*, **406**, p. 361 (1993).

[20] Steinmetz, M., *MNRAS*, **278**, p. 1005 (1996).

[21] Hachisu, I., Makino, J., Ebisuzaki, T., and Sugimoto, D., Three-dimensional vortex method on GRAPE-3AF - a special purpose computer for vortex method. *Parallel Computational Fluid Dynamics: New Algorithms and Applications*, eds. N. Satofuka, J. Periaux, and A. Ecer (Elsevier Sciences), pp. 155-160 (1995).

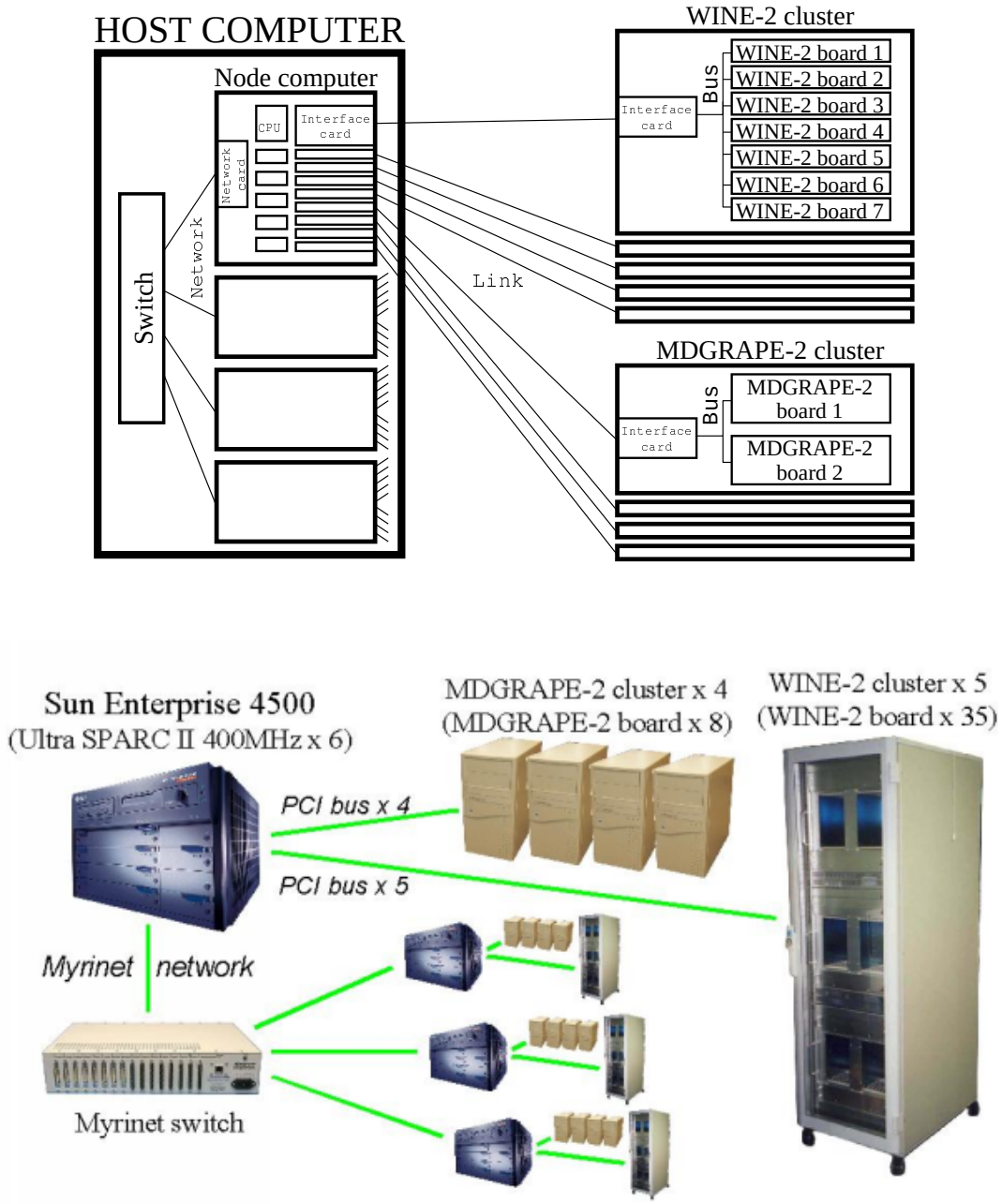

Figure 3: Block diagram of the Molecular Dynamics Machine
MDM is composed of WINE-2, MDGRAPE-2 and a host computer. The host computer is composed of four node computers, and they are connected with each other by a network. Each node computer has 5 WINE-2 clusters and 4 MDGRAPE-2 clusters via links. Each WINE-2 cluster has 7 WINE-2 boards connected by a bus. Each MDGRAPE-2 cluster has 2 MDGRAPE-2 boards connected by a bus.

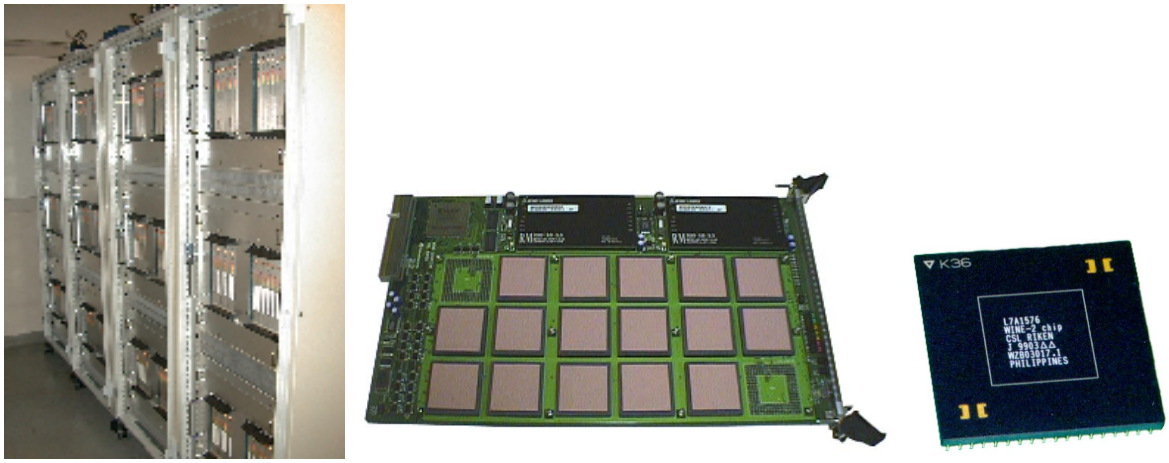

Figure 4: (a) WINE-2 (left). (b) A WINE-2 board with 16 WINE-2 chips (middle). (c) A WINE-2 chip (right).

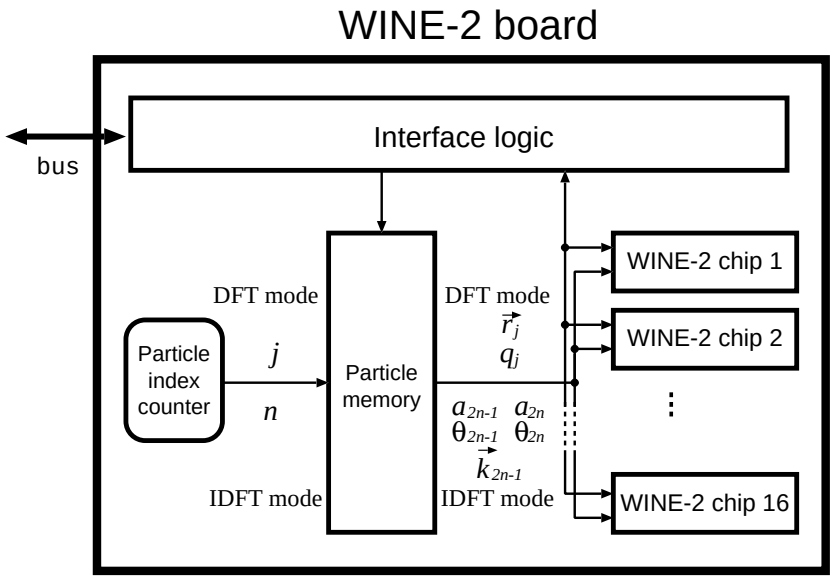

Figure 5: Block diagram of a WINE-2 board

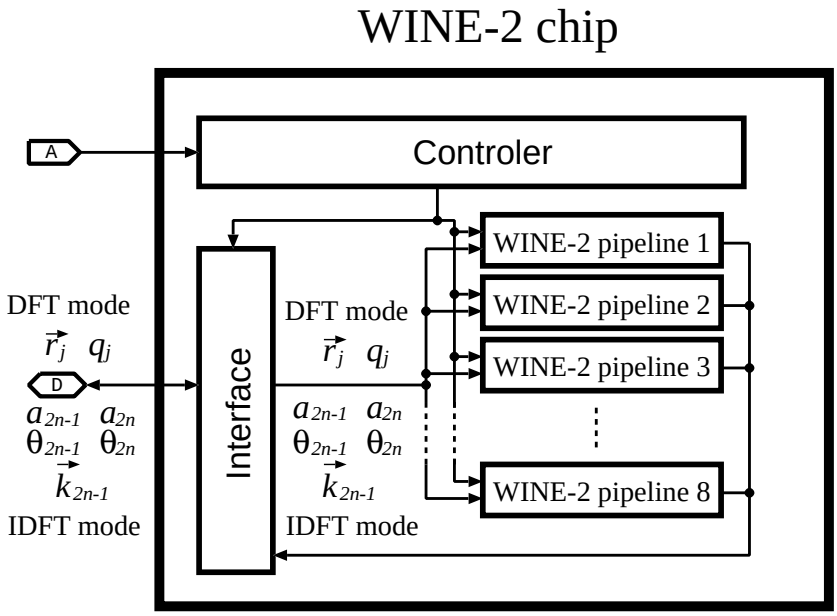

Figure 6: Block diagram of a WINE-2 chip

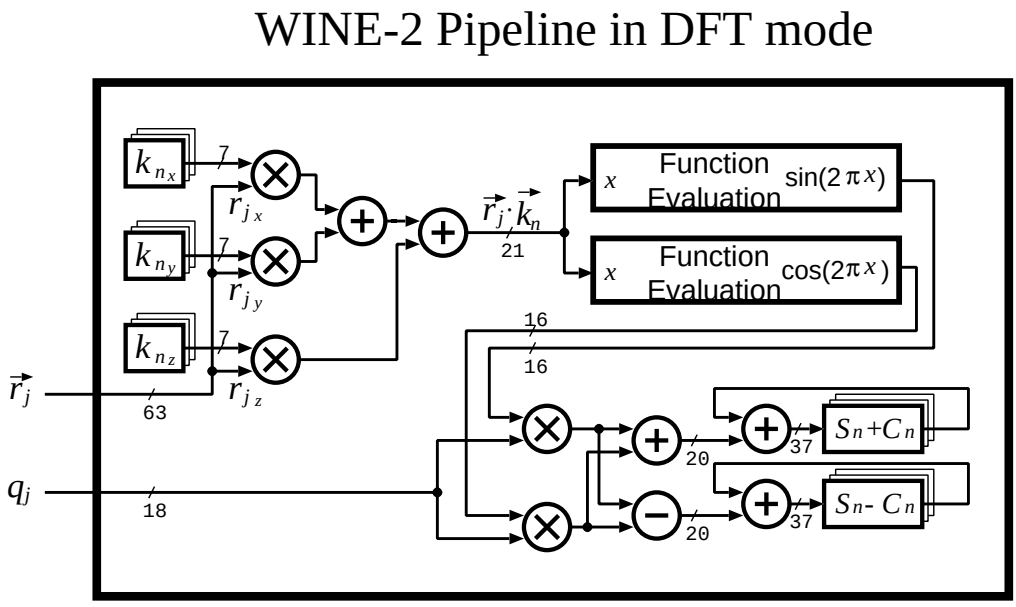

Figure 7: Block diagram of a pipeline of a WINE-2 chip in DFT mode

Figure 8: (a) MDGRAPE-2 and a host computer (left). (b) An MDGRAPE-2 board with two MDGRAPE-2 chips (middle). (c) An MDGRAPE-2 chip (right).

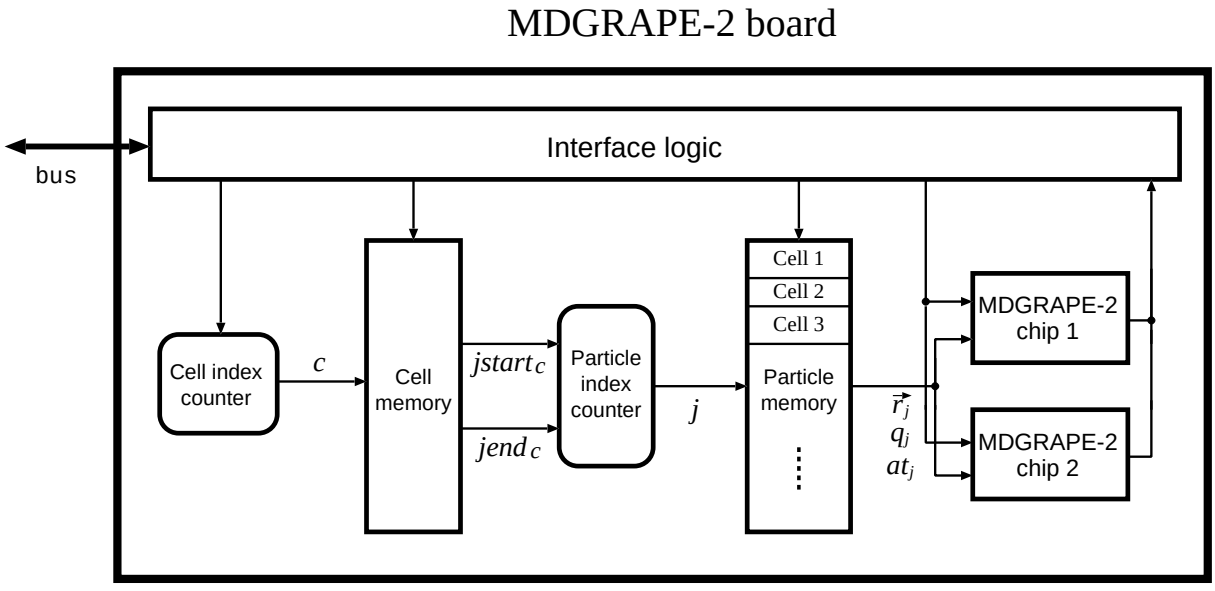

Figure 9: Block diagram of an MDGRAPE-2 board

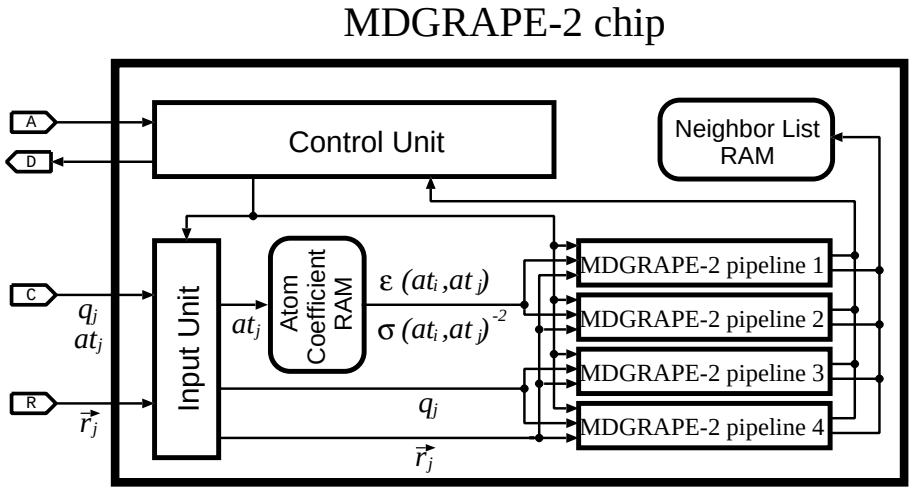

Figure 10: Block diagram of an MDGRAPE-2 chip

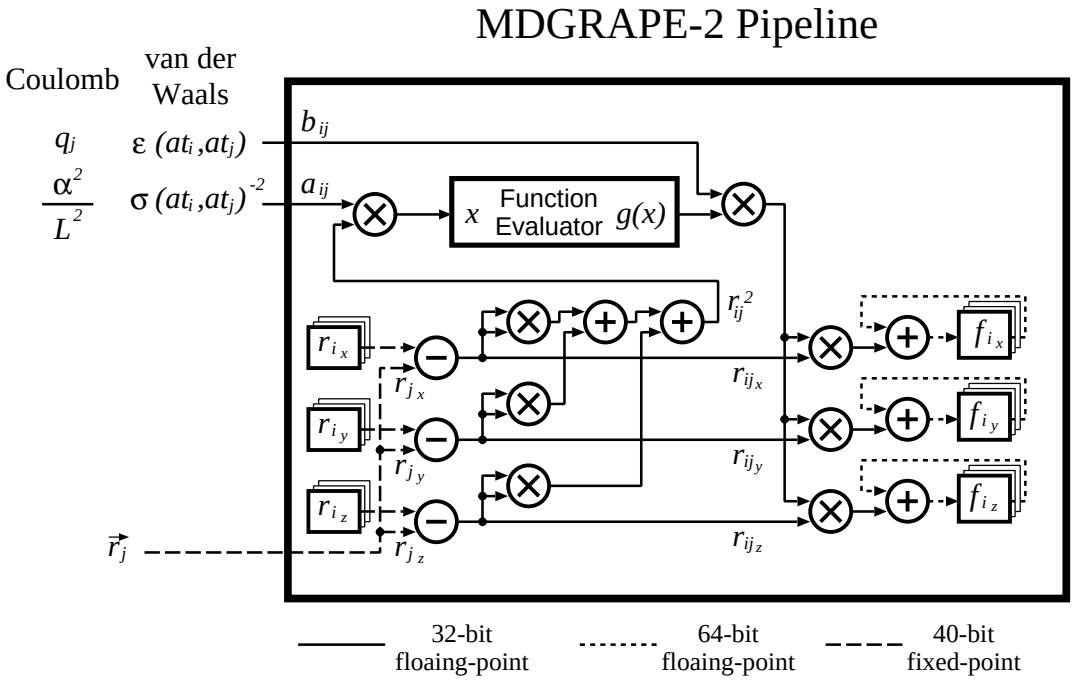

Figure 11: Block diagram of a pipeline of an MDGRAPE-2 chip