A Massively Parallel Computer with User Configurable Hardware

A Massively Parallel Computer with User Configurable Hardware.

NGEN is a test version of a massively parallel computer which is reconfigurable by the user in software right down to the level of the elementary digital processing and communication. NGEN arose from the desire to perform long-time simulations on large populations of strings to study basic principles of creative information processing as found in natural evolution. With this device it will be possible to simulate populations of thousands of biopolymers for the first time on the timescale of evolution on the earth (4 billion years), i. e. for terageneration (tera=10^{12}). A linear tradeoff between computation speed and population size has been achieved up to populations in the millions. NGEN was constructed at the IMB and came up on the IMB network in June 1994 behaving through its host processor like a unix workstation. NGEN is not an attempt to compete in the heady market of computer speed, but rather to allow a direct access to massively parallel hardware, matched to the problem at hand, for the increasing number of scientific problems like molecular evolution which essentially involve combinatorics and automata rules. The computer architecture is extremely flexible and will allow the computation of a wide range of massively parallel problems including optimization problems, sequence alignment, folding, neural nets, cellular automata, image processing and on-line data-analysis.

1. Basic Principle of Operation

NGEN has been designed to follow as closely as possible with available hardware the metaphor of programmable matter, that is of distributed logic, locally configurable to emulate local physical, chemical or biological laws. NGEN does not yet implement dynamically programmable matter, as the configuration process requires global intervention and takes a fraction of a second rather than a microsecond. It does, however, implement the laws in digital electronic hardware with all the associated removal of hardware and software overhead when compared with a software emulation. Typically, elementary model transitions are accomplished at frequencies of tens of MHz and in parallel for thousands of processors.

While configurable chips (e. g. GALs and EPROMs) have found their place on almost all commercial circuit boards nowadays, they must be removed and reconfigured using higher electric fields or ultraviolet light in a special device. The SRAM based Field Programmable Gate Arrays (FPGAs), on the other hand, may be reconfigured in software. Configurable electronic connections are achieved using multiple transistors grouped to form so called programmable interconnect points (PIPs) controlled by memory bits. Such configuration bits are daisy chained to allow them to be loaded in series, for example from a host computer. Configurable combinatorial logic functions are constructed using SRAM based look up tables.

NGEN is constructed in a three tiered fashion using host workstation, control FPGAs and agent FPGAs. Only a very basic bus interface to the custom cards comes up when the system is turned on: just sufficient to allow the control FPGA on each card to be configured from the host workstation. Equipped with their own high speed memory, the control FPGAs may be configured to implement a vast range of bus interface communication tasks including interrupt handling. Over 10^5 bits of information control the digital circuits implemented in the FPGAs. The control FPGA is not normally configured by the user, although it is perfectly possible to adapt the communication between control and agent FPGAs to the task at hand. The agent FPGAs, each with their own high speed memory, are then configured in their turn to implement the massively parallel design.

Finally, for problems of higher dimensional topologies (> three dimensions), the very broad band connection structure between the agent FPGAs may be flexibly reconfigured in hardware.

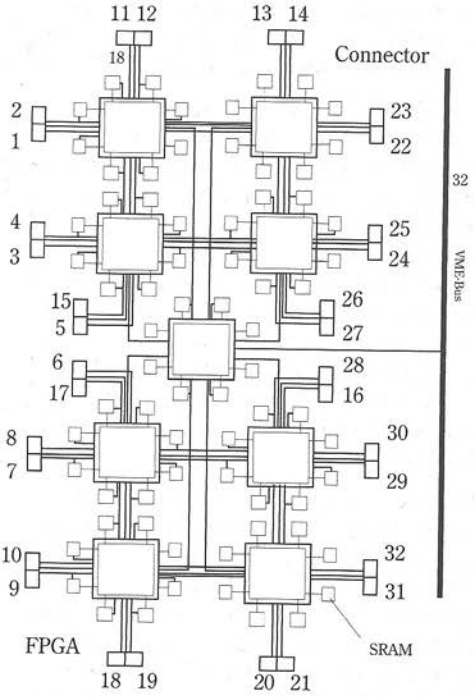

Fig. 1: Design layout of NGEN custom boards. Shown are the four basic elements which comprise the NGEN custom boards: FPGAs, high speed SRAM, connectors and the bus interface. The diagram shows the hierarchical arrangement of FPGAs, the distribution of SRAM, the very broad band distributed communication to other boards via the connectors and the central control FPGA which specifies the communication with the host computer via the VME bus.

2. Hardware and Software

A hardware manual was produced which documents the detailed charactersitics of NGEN. At this stage of development, the computer is particularly well suited to integer arithmetic and logical computations that exploit the massively parallel architecture with up to 40,000 user-specified processors. Since the configuration is done in hardware, processing speeds at the clock frequency of up to 40MHz for the user specified operations may be achieved. The computer operates currently on a 32bit VMEBus and consists of an HP742i (cf HP715/50MHz) host, 18 identical custom-made boards and a high resolution graphics board. 40MB fast (15nsec) RAM is distributed between 162 FPGAs (Field Programmable Gate Arrays) on the 18 reconfigurable boards (see Fig. 1). In addition to the relatively slow VMEBus interface, a flexibly reconfigurable 288 bit-wide interface in 9-bit modules allows a hardware reconfiguration of the total architecture for superplanar architectures such as 3D and hypercube connection geometries between the user-defined processors. A summary of the essential hardware statistics is shown in Fig. 2.

User access to the hardware capabilities of NGEN has been implemented at two levels. A run time environment, as extension to the unix operating system, has been written to allow interactive access to the machine. Design loading, debugging and execution control as well as IO control are available. Debugging is essentially parallel and symbolic with a single step facility. A graphical interface should be completed in early 1995, which will also provide the full palette of interactive execution control as a callable library of C routines. Parallel programming consists of the three steps:

i) processor circuit design (usually done with the aid of a vendor CAE graphical tool)

NGEN Hardware Statistics

Max. Processors	**93312**
Architecture	**1D, 2D, 3D,Torus, Ncube** **etc** and novel
EPGAs Chips	**162** XC4008
Config. Logic Blocks	**324**(18* 18) /FPGA
FPGA 10-Pins	**208** Pins/Chip
Distr. Mem Chips	**SRAM** (15ns 8*32k)
Distributed Memory	**42 MB** (=1296 SRAMs)
Boards	**18** (6U)
Bus	21 Slot **VME** 32
Board Interconnect	**288** (=32*9) lines/board
Clock	**16 (40) MHz**
Power Consumption	**<= 1800 Watt**
Cooling	**Air**
NGEN Runtime Env.	**C++ Shell for NGEN**
Hos	**HP743i** Workstn Card
Host Clock	**100 MHz**
Host Memory	**64 MB**
Host Operating System	**Unix** HPUX 9

Fig. 2: NGEN Hardware Statistics.

ii) design proliferation with the user connection structure (C program)

iii) execution and IO control program (C program)

One example of such a design procedure is for the case of interacting Turing machines.

3. Application to Genetic Processing

The way in which genetic information processing models can be captured with simple processor designs can be best understood by way of a simple example. In this section we present a prototype design, based on a simplified model of interactive molecular evolution driven by purely local interactions between molecules. Biopolymers such as DNA and RNA store information in a transmissable form in the sequence of monomers which comprise them. The discovery of catalytic RNA has lead to much speculation about an RNA world in which a single type of molecule, RNA, provided both information transmissability and catalytic functions for early life. In this simple model, we also deal with a single type of sequence which encodes both transmissable information and catalytic function. All complications of molecular folding are ignored in the example for clarity.

The molecules are assumed to be subject to diffusion (resulting in mixing) and pairwise processing at reactive centers with properties determined by the local constellation of other molecular sequences. The serial processing of pairs of such molecules, henceforward usually referred to as strings, allows diffusion and the three features of Darwinian evolution: amplification, variation and selection to be captured in a single unit called the genetic switch. In its simplest form, shown in Fig. 3, this genetic switch implements in its four different states different two output binary functions of its two binary inputs.

The two left states of the switch allow the incoming strings to pass unchanged or to swap their contents. Random changes between these two states between molecules for an array of these switches in two dimensions allows an implementation of molecular diffusion. Changes in state during the transmission of a pair of molecules allows genetic recombination and mutation. The two right hand states allow specific copying of the contents of one of the two incoming strings onto both outputs. This allows both amplification and selection at the level of whole strings and mutation and recombination when the state changes during the processing of two strings. Elongation and shortening of strings are also thereby possible, but insertions and deletions are not included in the elementary version of the model. Simple extensions to allow variable length processing have been worked out.

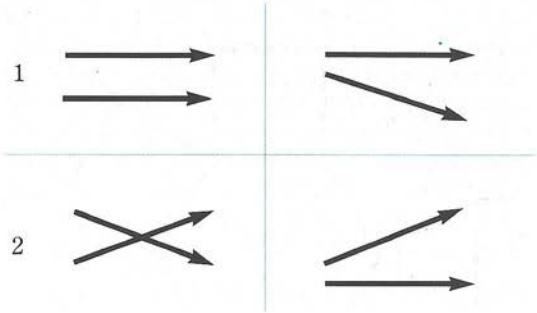

Fig.3: Genetic Switch. The genetic switch is a two input, two output switch which processes genetic sequences base by base. The switch has four states and depending on whether switching is applied inside or at the end of successive sequences, a wide range of dynamical mixing or variation (mutation, recombination etc) of the strings can be realized.

The state of the switch is regarded as a simplification of the complex local chemical environment to which the biopolymers are subject, including in general bound enzymes. It is important for proper diffusion of the molecules, that the diffusive state of the switch be as uncorrelated as possible with the sequences of the molecules passing through the switch. Moreover, the state of the switch should change much more frequently between molecules than during molecular processing. The copying state of the switch should, depending on the model being studied, depend on the strings passing through the switch and/or the sequences of other (third party) strings near the switch. Differential amplification is then either a direct result of differences in sequences passing through a switch or of the different enzymatic properties of strings in modulating the states of switches or both. Specific third party control of the switches by enzymatic sequences opens up another level of complexity in the evolutionary behaviour. Even in the absence of such effects, the selection may be frequency dependent as in the theory of evolutionary games: the reproductive success of a sequence depends on the constitution of the rest of the population.

The strings are implemented in memory cells addressed as shift registers which are distributed in a regular lattice through the FPGA and connected pairwise at the genetic switch nodes. The population of strings extends into the off chip memory and local memory in the FPGA is used to retain individual pairs of strings prior to sequence dependent switching decisions. Data flow of the strings through the shift registers and switches is constant at the clock speed (up to $4x10^7$ bits/sec). The switches simply control the way in which incoming bits from two converging shift registers are disributed on the two diverging shift registers. The states of the switch are determined by rules which in general depend on both sequences as in evolutionary game theory, giving rise to concentration dependent effects.

A number of geometrical considerations have proved important in establishing a correct physical description of the diffusional processes underlying population mixing. An array of switches with two inputs and two outputs can be linked up in a variety of different topologies. A simple rotationally symmetric solution consists of a chequerboard on a square lattice with two basic units with opposing inputs vertical (outputs horizontal) in one case and horizontal (outputs vertical) in the other. This configuration allows the population to be loaded initially following square wave shaped paths. Unfortunately, as pointed out by B. Böddeker, for a population of strings of constant length, this topology partitions the strings into four categories which never meet in the population and as such would provide an unphysical bias for variable string length simulations. B. Böddeker discovered a simple connection topology, illustrated in Fig. 4, which contains primitive cycles of lengths three and four allowing complete mixing of the population. This type of geometry is just as

easy to implement on NGEN because of the configurable nature of the local connection structure. Alternatively, the square lattice can be retained and a differential time delay introduced on one of the two incoming strings. The latter method has the advantage, that the residence time for strings in the active processor plane (between sojourns in the deep SRAM based shift registers) is also different, resulting in a diffusive mixing of the entire population. The utilization of this device of hyperplane mixing solves the problem of efficient data flow for populations larger than can be retained in the processors simultaneously, extending the population sizes from the 10^4 to the 10^6 level. Current work involves a four input, four output processor with two delay channels but here we continue with the simple example of two input and two output processor described above.

This simple processor can be implemented in four of the primitive configurable logic blocks (CLBs) which are arranged in an 18x18 array in each FPGA. The processor consists of local string storage (2 CLBs), the genetic switch (1 CLB) and its control (1 CLB), as shown in Fig. 5. The control can include random and/or sequence dependent elements. Random bits are generated in parallel using a cellular automata like construct. Configuration of these four blocks can be achieved interactively by mouse click using a CAE design editor (XACT design editor: firmware of Xilinx® Inc.). Programs in conventional computer language, in this case C, were then written to generate macros for the proliferation of the processors and to wire up the desired connection topology. The actual wire routing is performed automatically with the aid of a simulated annealing procedure. The design process is completed by using FPGA firmware to generate a bitstream which can transfer the digital design to the FPGAs in NGEN. For more complicated processors, a schematic editor and automatic partition, place and route firmware

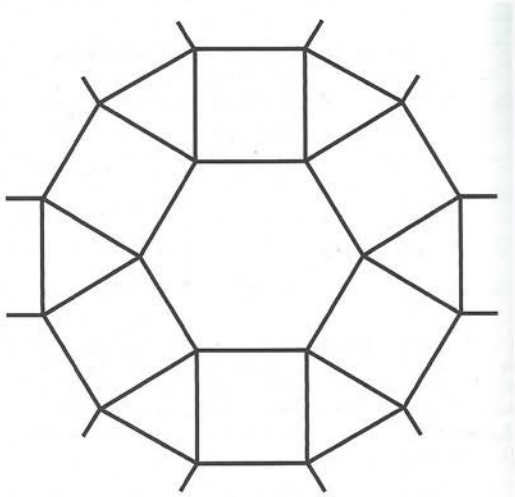

Fig. 4: Olympic Rings Topology.
This topology, in contrast with a square planar one, ensures that all strings moving stepwise from segment to segment in a synchronized fashion can eventually meet at some node. Each node, as in the square planar graph, has four nodes but the graph consists of elementary three and four cycles whose lowest common denominator is one.

may be used. The user can choose the level of interaction with the hardware right down to individual gates. An increasingly high level set of design modules allows a cumulatively more efficient design process, akin to the use of library routines in mathematical packages on conventional computers. For example, adders and multipliers in various bit widths are provided as design modules in addition to various register and stack structures.

By this means, 64 such processors were packed onto each FPGA, leaving ample room at the periphery for additional test and control structures, giving a total of 144 x 64= 9216 two string processors in the machine. The design may be modified in parallel, using the same macro structure employed in processor proliferation and

interconnection, leading to an edit, compile and debug cycle time of just a couple of minutes. After a number of editing steps, the design should be fed again through the rip-up routing firmware to reoptimize the connection structure since performance is successively degraded during local changes to the routing.

4. Application Range

The above example, merely serves to illustrate the application of very simple processors in massively parallel simulation. The group has made detailed plans to extend this type of calculation to realistic models of coupled macromolecular evolution to support our experimental program. Sequence dependent binding and catalytic activities may be modelled which take intramolecular folding and stochastic kinetics into account. Most importantly, the modelling level here is at an individual level so that populations with millions of different sequences can be treated without additional computational overhead. This is important for evolutionary applications at the molecular level, where a significant fraction (for example 50%) of the population consists of unique sequences (i. e. present in only a single copy).

On the other hand, a much wider range of applications are possible which reach into and beyond the disciplines represented at the IMB. For example, the work of Huang and Lopresti demonstrates that unprecedented computation power for sequence analysis can be harnessed without custom made chips using systolic algorithms to implement dynamic programming. Here only the one dimensional connection structure between FPGAs need be utilized. Two essentially linear configurable computers, one in the USA and one in Israel, have been specially designed to foster such calculations. The ability to perform such calculations on especially large sequences is a by-product of NGEN's architecture. In particular, the distributed high speed SRAM opens up new possibilites for multiple sequence alignment. Other dynamical programming algorithms such as the N3 alogorithm for RNA secondary structure can also be implemented on NGEN to run directly in hardware. Care must be taken that the problems be formulated to minimize the need for massively parallel floating point calculations: currently it is not easy to pack them densely in FPGAs.

Various combinatorial optimization problems, also those relevant to structural calculations in molecular biology, can also be implemented. Here, a planned development in the parallel memory ac-

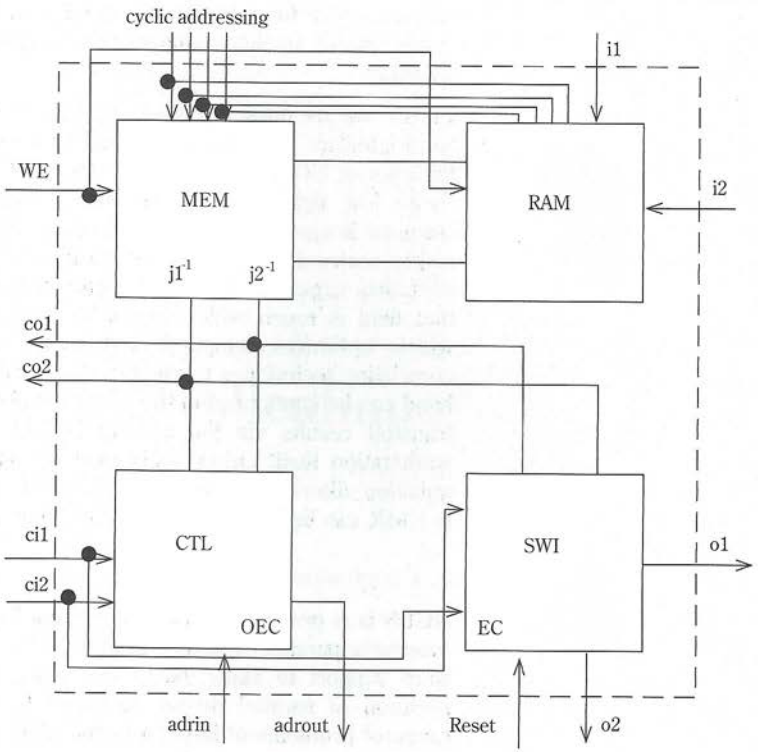

Fig. 5: 4 CLB Prototype Genetic Switch.

The diagram shows a block diagram of the way in which the logic of the genetic switch can be packed into four of the basic configurable blocks at the lowest level in Xilinx FPGA design. Two of the blocks (Ram and Mem) simply function as two parallel shift registers to propagate and store the sequences locally, one as control (CTL), to determine the sequence and environment dependent changes in state of the the switch, and the fourth (SWI) implements the actual switch which controls the flow of the sequences between the two inputs i1 and i2 and the two outputs o1 and o2.

cess structure for a production version of NGEN would greatly further enhance the computational potential.

Finally, we are pursuing plans to utilize the broad band interface and massively parallel computation facilities of NGEN to process single photon data (from low light level fluorescence) from time resolved images. Our immediate application is in single molecule selection and spatially resolved evolution experiments, but the potential application field is much wider. Specialized processing which optimizes temporal and spatial photon correlation techniques to the detection problem at hand can be configured in the agent FPGAs which transmit results via the control FPGAs to the workstation host. Other sophisticated on-line acquisition filters for experimental data, for example in NMR, can be implemented by this technique.

5. Conclusion and Future Developments

NGEN is at present established as a configurable, massively parallel computer at the IMB with software support to allow users to explore the application of parallel digital hardware to a wide range of problems of interest to the IMB and the wider scientific community. It is possible both to run massively parallel applications on NGEN under the control of a host workstation or to use NGEN as a powerful coprocessor callable from within C programs for user definable tasks.

The latter approach has been pioneered for small numbers of FPGAs by other groups in Germany. NGEN's speciality is to allow massively parallel programming with a flexible topology.

A major extension of configurable hardware to increase the communication flexiblity, high speed distributed memory (population size) and to allow rapid reconfiguration is planned for the coming three year period. This development will embrace optical communication. In the short term, a significant step forward in the construction of an improved version of the current NGEN architecture, which will greatly enhance its application to problems of combinatorial optimization, is planned for 1996. Making use of now established electronic expertise and equipment at the IMB, this development is achievable at costs equivalent to that of a single high end workstation, and will cover a computational niche, complementary to the serial processing of conventional workstations, namely custom massively parallel computation, which is a key one to evolutionary biotechnology.

John S. McCaskill
Molecular Information Processing

References

[1] J. S. McCaskill, U. Gemm, T. Maeke, and U. Tangen. NGEN - A massively parallel reconfigurable computer: hardware description manual. Technical Report IMB Jena (1994).

[2] K. Mekelburg, T. Maeke, and J. S. McCaskill. NGEN - A massively parallel reconfigurable computer: run time environment. Technical Report IMB Jena (1994).

[3 J. S. McCaskill. NGEN – A massively parallel reconfigurable computer: programming NGEN hardware with C(++) and the XACT environment. Technical Report IMB Jena (1994).

[4] J. M Arnold, D. A. Buell, and E. G. Davis. "SPLASH 2" in: Proc. 4th annual ACM symp. on parallel algorithms and architectures (1992) 316-322.

[5] L. Esterman "Bioaccelerator: A currently available solution for fast profile and SW searches". Biological Computing Division. Weizmann Institute of Science Rehovot 76100 Israel (1995).