

Extended Abstract

Continuum Computer Architecture for Nano-scale and Ultra-high Clock Rate Technologies

1. Introduction

Continuum Computer Architecture (CCA) is a non-von Neumann architecture that offers an alternative to conventional structures as digital technology evolves towards nano-scale and the ultimate flat-lining of Moore's Law. Coincidentally, it also defines a model of architecture particularly well suited to logic classes that exhibit ultra-high clock rates (> 100 GHz) such as Rapid Single Flux Quantum (RSFQ) gates. CCA eliminates the concept of the "CPU" that has dominated computer architecture since its inception more than half a century ago and establishes a new local element that merges the properties of state storage, state transfer, and state operation. A CCA system architecture is a simple multidimensional organization of these elemental blocks and physically may be considered as a new family of cellular computer. But CCA differs dramatically from conventional cellular automata. While both deliver emergent global behavior from the aggregation of local rules and ensuing operation, The CCA emergent behavior is a global general-purpose model of parallel computation, as opposed to simply mimicking some limited phenomenon like heat and mass transfer.

The notional concept of Continuum Computer Architecture is motivated by trends in device and communication technology that while dramatically increasing the raw aggregate capability in terms of storage capacity and logical capability, are becoming increasingly unsuitable for scalable parallel system architectures based on conventional microprocessor and MPP system architectures. This is demonstrated by very low efficiencies (single digit in many cases) delivered for real applications resulting from the "memory wall" also known as the "von Neumann bottleneck" that is a combination of contention resulting from poor bandwidth and latency resulting from long access times measured in clock cycles (aggravated by increased clock rates). The balance of conventional system resources is increasingly inappropriate as 100's of square millimeters of processor chip real estate are dedicated to optimizing the operation of floating point ALUs that take up less than a single square millimeter on the same chip. The result is a potential three orders of magnitude (or more) disparity between delivered performance and potential peak performance. But these metrics are themselves inappropriate and are only relevant because of the bottleneck of single instruction issue operation, implicit in conventional microprocessor architecture design. The real metrics of operation that matter are 1) time to execution, 2) scalability, 3) normalizing factors of power and cost, 4) reliability, and 5) programmability and generality. CCA provides a revolutionary way to address all of these in the context of advanced technologies in the near nano-scale regime.

2. Brief Overview of CCA

Continuum Computer Architecture can be viewed at multiple levels: 1) the physical hierarchical structure, 2) the local logical level, and 3) the global execution model that the system supports. Conceptually, these interrelated levels are mutually supportive in their respective effects.

2.1 Physical Structure Level

The physical CCA system comprises three levels of organization. The innermost building block of the system is the “*Fonton*”, a simple cell that incorporates a mix of data processing logic, associative storage, some external interfaces to adjacent neighbor fontons, and some control logic. The operation of the fonton is driven by a basic message class called a “*Parcel*” that specifies an action or sequence of actions to be performed on local data. Data is stored in a bank of associative buffers. A data record is identified by its virtual address that is recognized associatively. Most parcels pass through a given fonton which serves as a communication channel and router when a datum that is sought for is not locally stored. The parcel may be requesting a given datum to be returned to some other part of the total system and when locating it in a given fonton causes a new return burton to be generated. A parcel may also cause the local state of a fonton to be modified atomically. Thus, the fonton serves as a data transfer node, a small block of virtual memory, and as an arithmetic/logic unit.

The middle level of the system is the “*Simultac*”, a multidimensional uniform structure of fontons. A full scale system may incorporate billions of simultacs. While a 3-D structure torus, is a likely structure with each fonton directly interfaced to six neighbor fontons, alternate organizations are possible to increase bandwidth or reduce system-wide latency. The simultac appears simultaneously as a block of memory with very high memory bandwidth, a large mesh-connected system-wide wormhole routing communication fabric, and a large systolic array that is controllable on the fly.

The highest or system level combines the full simultac with a combination of external memory and I/O ports. This can be pictured as a cubical simultac with an outer shell layer of distributed memory around all six sides and I/O channels on the outside of the memory shell. This provides a secondary storage level, although implemented in conventional DRAM, that permits the simultac to overflow when the data set grows larger than the intrinsic storage capacity of the combined fontons, as well as providing very high I/O bandwidth.

2.2 Local Logical Level

At a higher level of abstraction than the physical level, the CCA distinguishes between its local operations and global behavior. The local logical level emphasizes very fast and efficient lightweight split-transaction processing with fine-grain parallelism. All operations are message driven. Data retained locally may be part of one or more larger data structures including metadata. A parcel would arrive and detecting sought for data

may cause an operation to be performed between operands carried by the parcel and the identified local data. A parcel may carry more than one instruction if to be performed on the same initial or intermediate data. The result of the local action is one or more of several types:

1. routing a parcel through the system,
2. identifying and acquiring local data to be sent to some remote part of the system,
3. modifying local state, and
4. sending new values to follow on computation.

2.3 Global Execution Model

The similtac operates through the synergism of the combined fontons to create a global parallel execution model. The model of computation is a fine grain multithreaded, message-driven discipline with a global virtual data name space and a *futures* based synchronization scheme for asynchronous operation. Both data-driven and demand-driven computing styles can be supported along with efficient producer/consumer and data flow control flow. Of particular importance is the elimination of global barrier synchronization as the primary control construct, greatly enhancing average parallelism through fine grain control. CCA is particularly well suited to the processing of large sparse data structures with significant meta data components such as adaptive mesh refinement and symbolic semantic nets.

3. Conclusions and Important Advantages

Continuum Computer Architecture is a truly parallel computer architecture designed from the outset to execute parallel programs efficiently. It emphasizes locality of action while providing asynchronous control between remote elements, data, and actions. As clock rates increase and as feature size is reduced to nano-scale, concurrency of action at the fine grain becomes increasingly important for efficiency and scalability. CCA will support such future technologies as superconducting rapid single flux quantum gates with clock rates exceeding 100 GHz such that it can take upto a hundred clock cycles to propagate a signal across an entire chip. Similarly, CCA will support quantum dots which provide exceptionally small components but relatively slow long distance signal propagation. Both technologies demand an architecture like CCA to be made an effective base for high performance computing.

CCA exhibits a single system image at the parallel control flow and data name space level. It incorporates automatic resource management employing a diffusion policy for acquiring new memory and garbage collecting. Vectors of fontons can be easily programmed to operate like a vector pipeline, each stage performing a different function on a vector of data carried by the parcel. Alternatively, it can store the data in a contiguous sequence of fontons to form a vector and pass a parcel with a stream of operations across the data vector. In either case, a software controlled vector structure performs at hardware speeds, but without the need for reconfigurability. CCA dramatically increases the amount of arithmetic units and the memory bandwidth on a semiconductor die improving the peak performance and memory bandwidth of between 1

and 2 orders of magnitude per chip. It exploits fine grain parallelism thus providing superior scalability and sustained performance. It enables efficient manipulation of complex data structures and facilitates complex in-memory operations including sophisticated synchronization mechanisms like futures, empty/full bits, queues, and stacks. Because of its uniform structure of highly replicated fontons and its virtual name space, it permits high reliability through graceful degradation by allowing failed fontons to be isolated from the rest of the system, which continues to operate.