"Nature is pleased with simplicity, and affects not the pomp of superfluous causes" - Newton

Common practice is seldom simple and never perfect. Innovation is its natural, essential antagonist.

Green Arrays™ Architecture

FEATURES

- Many independent computers
- Computers are self contained
- Chips are complete systems
- No energy wasting clocks
- Fine grained control of energy expense
- High speed available when needed
- Simple, fast communication with automatic, energy efficient flow control
- Software Defined I/O
- I/O promotes efficient system design
- Multilevel Programming
- Wide range of chip sizes and costs

BENEFITS

- Put intelligence where it is needed at minimal added cost in BOM or energy
- Create deeply distributed systems that are otherwise impractical
- Use only the energy necessary for useful work rather than being forced to build heating elements
- Create devices that can respond to events - even complex events that require software recognition - in nanoseconds, from the deepest low power states, while others respond in microseconds and only to trivial events
- Inventory few, versatile chips
- Achieve performance for which others must pay higher costs in all respects.
- Plan for SOC or wafer scale products that amplify these advantages
- Free your intellect to invent novel, superior solutions

APPLICATIONS

- Embedded systems
- Energy harvesting applications
- Portable devices
- Smart sensors
- Complex control systems
- Cryptography
- High speed signal processing
- Simulation and synthesis
- Inexpensive, massively parallel systems for research and education
- Artificial intelligence, neural nets

OVERVIEW

Each GreenArrays chip consists of an appropriately sized array of architecturally identical, independent, complete computers, or *nodes*. The computers are self-contained, each with its own memory. Think of a computer as a function in a block diagram; context switching is unnecessary to *continuously* perform this function.

COMPLETE SYSTEMS: We refer to our chips as *Multi-Computer Systems* because they are, in fact, complete systems. Supply one of our chips with power and a reset signal, and it is up and running. All of our chips can load their software at high speed using a single wire that can be daisy chained for multiple chips; if desired, most can be bootstrapped by a simple SPI flash memory. Application software can be manufactured into a custom chip for a modest cost to further simplify overall system design. External memory is not required to run application software, but our larger chips have sufficient I/O to directly control external memory devices if desired.

Contrast this with a *Multi-Core CPU,* which is not a computing system until other devices such as crystals, memory controllers, memories, and bus controllers have been added. All of these things consume energy, occupy space, cost money, add complexity, and create bottlenecks. Most multi-core CPUs are designed to speed up conventional operating systems, which typically have hundreds or thousands of concurrent processes, by letting a handful of process execute in parallel as opposed to only one. They are not, typically, designed for significantly parallel processing, and they are even less well suited for simple applications than are their less expensive single-core progenitors.

NO CLOCKS: Most computing devices have one or more clocks that synchronize all operations. When a conventional computer is powered up and waiting to respond quickly to stimuli, clock generation and distribution are consuming energy at a huge rate by our standards, yet accomplishing nothing. This is why "starting" and "stopping" the clock is a big deal and takes much time and energy for other architectures. Our architecture explicitly omits a clock, saving energy and time among other benefits.

FINE GRAINED ENERGY CONTROL: Each computer operates asynchronously. When a computer has finished its work, it typically reads or writes an I/O address (one or more of communication ports and I/O pin.) The read or write instruction is automatically suspended in mid-operation if the address is inactive, consuming energy only due to transistor leakage currents, resuming when the address becomes active. In fact, after reset all nodes in our chips are by default suspended in this manner. For practical purposes neither time nor energy is expended in the process of suspending and resuming execution. Since energy is something that can be spent like money, consider the importance of granularity in "billing". Expenses for telephone usage, billed by the minute, can be controlled more tightly than can, for example, usage of Professional services that are billed in minimum units of an hour. When compared with other computer architectures, our systems offer much more control over energy consumption through duty cycles that "bill" only for useful work, and in small granules (the small power for individual computers and only during the short time when each is active). We refer to our control over energy consumption as *fine grained* for this reason.

IT'S WORK, NOT SPEED, THAT USES ENERGY: Our computers are very fast (our goal, with continued improvements, is one billion operations per second for each node) but because we have no clock and because we suspend when idle, this speed actually saves energy rather than burning it, and has no intrinsic cost because it simply comes with the territory of our semiconductor geometry. If the application does not need high speed, great; it can run at a low duty cycle and consume little energy.

Green Arrays™ Architecture

COMMUNICATION: Efficient communication is fundamental to our multicomputer architecture. Each computer has 5 memory-mapped port addresses. Writing to one is suspended until the node it connects to reads, and vice versa. A computer can read from multiple ports and can execute instructions directly from those ports. Computers can talk to their neighbors and their peripherals if any; some may talk to a remote node, depending on chip configuration.

SOFTWARE DEFINED I/O: The GreenArrays architecture incorporates dedicated interface hardware only where direct software "bit banging" is impractical. The argument that hardware is needed to offload a burden on the CPU evaporates with numerous fast CPUs. Versatile I/O pins mean fewer chip models to inventory.

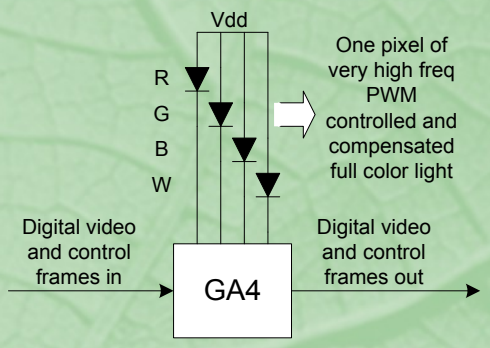
Any computer on an edge of one of our arrays may be built with a selected class of I/O pin(s), such as general purpose analog or digital, the actual function of which is defined by the software running in that node. For example, our general purpose digital I/O nodes can be programmed to communicate using a variety of protocols including USB or Ethernet, or to excite and monitor a crystal or a resonator.

I/O DESIGNED FOR LOW-ENERGY INTERFACES: When an I/O pin is configured for input, its impedance can be high and therefore represent little load for other components to drive. When configured for output, we optimally drive small capacitive loads. These practices minimize energy consumption at the system level.

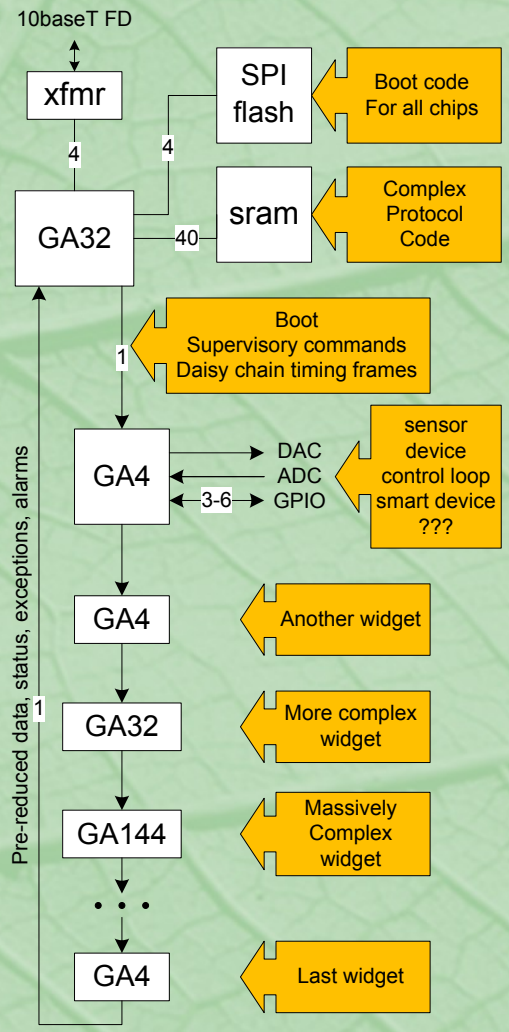
MULTILEVEL PROGRAMMING: The software running in a node might be likened to very high speed microcode running in a small memory. If an application needs a complicated communications protocol such as for example TCP/IP, which is too large to implement in a small number of nodes with "microcode", our architecture supports at least two higher level methods of programming: Code streaming, and interpretation of any high level language from an external memory. We supply eForth as an example of the latter. By adding a small SRAM device and committing a handful of nodes to running the interpreter, things like TCP/IP can be supported at a very respectable ratio of performance to both cost and energy consumption.

WIDE RANGE ARCHITECTURE: Most computer architectures have hard limits at both the top and bottom ends of the scales of performance, cost, and size. Because our computers are as small and as inexpensive as they are, we have been able to make chips that are truly suitable for use in large quantity as smart sensors, or trivial controllers, yet still with enough intelligence to allow for economical communications. At the other end, our larger systems are designed for building massive arrays of chips, such as for example a credit card sized circuit board containing nine chips and 1,296 computers. For customers building wafer scale devices, we are prepared to place assorted computing systems, sized appropriately to the need, at the sites where the work is needed. This wide range of capability allows us to propose distributed processing systems that cover a far greater depth than has been attempted heretofore, and using our modular architecture we can quickly and simply create an optimal chip for a particular role in a system.

For more information, visit www.GreenArrayChips.com



Use in LED Video Displays



Distributed Data Acquisition and Control System

GreenArrays, Inc.
774 Mays Blvd #10 PMB 320
Incline Village, NV 89451

(775) 298-4748 voice
(775) 548-8547 fax
sales@GreenArrayChips.com