STARAN SERIES E

Kenneth E. Batcher
Digital Technology
Goodyear Aerospace Corporation
Akron, Ohio 44315

Abstract. STARAN[(a)] Series E is an enhanced version of the STARAN parallel array processor. Multi-dimensional-access data storage and high speed control storage have been increased several-fold. Faster IC's and new processing algorithms improve the processing rates significantly. A new I/O unit allows faster and more flexible input and output of data. The cost impact of these changes has been minimized by using standard parts which were not available when the first STARAN was built. In this paper we discuss the enhancements and the reasons they were incorporated.

Introduction

As explained in [1] all logic and memory devices in STARAN are standard high-volume integrated circuits. The semi-conductor industry is continually improving these circuits, e.g., more bits on a chip and higher speeds. Thus, it is relatively easy to enhance STARAN by using the newer circuits as they become available. The goal in the design of the Series E systems was to enhance the capabilities of STARAN using newer circuits and better processing algorithms while preserving software compatibility with the original design.

The major enhancement in the Series E systems is a much larger multi-dimensional-access (MDA) memory in each array module [2]. Other enhancements are a larger high-speed control store, faster processing rates and a new I/O unit.

Larger MDA Memory

Need for Larger Memory

The MDA memory in the original design used 256-bit random-access-memory (RAM) devices. These were the largest bipolar RAM devices that were generally available at the time (1971-1972). The size of these devices governed the choice of 256-bit words with a simple PE per word.

After several applications were programmed on the system it became evident that larger words would be better. In most applications the size of the machine (number of array modules) is dictated by memory requirements instead of processing speed requirements. In some applications some data is off-loaded into the control store to make room in the MDA memories. In some other applications two or more words are used per item and some of the PE's are wasted.

Another indication that a larger MDA memory is desirable appears when the ratio of storage bits to processing speed (MIPS) is examined for several large computer systems. STARAN has a low ratio compared to other systems (either we have too many MIPS or too few bits). Increasing the storage by a factor of 20 would bring us in line with other systems.

MDA Memory in Series E

In the original STARAN design the number of bits per word was fixed at 256. Each array module contained 8K bytes in a 256 x 256 bit array and 256 PE's. To increase the storage capacity of a machine one added array modules and added more PE's as well. In the Series E machines one can size the processing power (number of PE's) and memory capacity independently. This is accomplished by multiplying the MDA memory address space by a large factor (256) and allowing the space to be partially populated.

Bipolar 1,024-bit RAM devices are generally available today. These devices have ECL - compatible interfaces so that they match the logic levels of STARAN better than the TTL - compatible devices of the original design and have faster access and cycle times. Thus, these devices are a natural choice for Series E.

MOS 4,096-bit RAM's are also generally available. To keep the slow MOS speed from severely affecting the processing rate of the machine, part of the storage should be bipolar and algorithms modified to move most of the memory accesses into the high-speed bipolar storage. The section on faster processing rates discusses these modifications.

A mixture of high-speed bipolar and low-speed MOS devices is allowed in the MDA storage of a Series E array module (Figure 1). The MOS MDA memory is an array of 256 mK bits where K= 1024 and m is a multiple of 4. It uses 4,096-bit MOS RAM's. The band width of the MOS memory is 256 bits in or out every 420 nanoseconds (76 megabytes/second). The bipolar MDA memory is an array of 256 by nK bits where n is an integer. It uses 1,024-bit bipolar RAM's. A read cycle in the bipolar memory requires 120 nanoseconds (267 megabytes/second) and a write cycle takes 160 nanoseconds (200 megabytes/second). For comparison, the MDA memory of the original STARAN had a read cycle of 120 nanoseconds and a write cycle of 300 nanoseconds.

The maximum MDA storage in each array module, limited by the address space, is 256 x 64K bits (2,097,152 bytes). The physical size of the array module grows with storage capacity and depends on

(a) T.M. Goodyear Aerospace Corporation, Akron, Ohio 44315

the mix of bipolar and MOS storage. In the first Series E machine, m=8 and n=1, for a capacity of 262,144 MOS bytes and 32,768 bipolar bytes per array module. Two array modules are packaged in one STARAN cabinet (array modules with greater storage capacity are packaged one per cabinet). For a cost increase of less than 50% these array modules have 36 times the storage of the original STARAN modules.

Accessing MDA Memory

In the original design each MDA memory access (fetch or store) required two parameters: an 8-bit access mode to select one of 256 stencil shapes and an 8-bit address to position the selected stencil at one of 256 positions [2].

The larger MDA address space in Series E requires a 16-bit address. Some thought was given to also increasing the access mode parameter and allowing some stencils to access every ith bit slice of a word (i=2,4,8,etc.). This idea was rejected because such stencils do not appear to be generally useful and they are hard to implement in a memory which is partially populated with both fast and slow memory devices. Because of the way data is scrambled in memory it is important that all 256 memory bits accessed at one time actually exist and either be all "fast" bits or all "slow" bits. The basic memory increment is 1,024 bit-slices so the maximum allowable access mode parameter is 10 bits--the increase from 8 bits to 10 bits does not add any significant MDA capability so the access mode parameter was left at 8 bits.

In Series E, one may view the MDA memory of an array module as a number of 256 x 256-bit planes (Figure 2). The leftmost 8 bits of the 16-bit address select one of the planes. The 8-bit access mode selects a stencil shape and the rightmost 8 bits of the address positions the stencil within the selected plane. All 256 bits covered by the stencil are fetched or stored in one memory cycle. The shapes of the 256 possible stencils are discussed in (2).

Bipolar MDA memory occupies the first 4n planes (n=1,2,3,...) and MOS MDA memory the next 4m planes (m=0,4,8,12,...).

Base Registers

In the original design the 8-bit MDA memory address came from one of five sources: an address field in the instruction, the resolver through the link pointer or one of three field pointers. The instruction address field is usually used to reference flag bits in fixed locations. The resolver and link pointer are used to reference particular words in the MDA memory, e.g., a word satisfying an associative search operation. The three field pointers are used to step through the bit-slices of fields in arithmetic and search operation; e.g., to add field A to field B with the result put in field C the three field pointers reference corresponding bit-slices of fields A, B and C.

In the original design the 8-bit MDA access mode came from one of two access mode registers (AMR0 and AMR1) depending on the state of a mode bit in the instructions referencing MDA memory.

In Series E, five base registers are included in the control unit; one for each of the five sources of MDA memory addresses. The final 16-bit MDA memory address is formed by adding the 8-bit source to a 16-bit base address in the associated base register. The 8-bit access mode comes from a field in the base register. This arrangement allows addressing of: 1. flags in a flag-bit region, 2. words satisfying a search, and 3. fields in scattered memory regions without modifying the base registers.

The speed of arithmetic instructions with long sequences of micro-steps (multiply, divide, square root and floating-point) would be severely affected if all micro-steps addressed fields in the MOS MDA memory. To speed up these instructions, a sixth base register is used as a pointer to the base of a vector stack in the fast bipolar MDA memory. The long arithmetic instructions move vector operands from MOS memory to the bipolar stack, operate on the stacked vectors and then return the results to the MOS memory. Guard bits are added to the vectors when moved onto the stack and rounded-off when results are unstacked.

The mode bit of the instruction (used in the original design to select AMR0 or AMR1) is used in Series E to select vector stack addressing or the five-base-register addressing. In vector stack addressing, the 16-bit stack base address in the sixth base register is added to the 8-bit source regardless of its source so system micro-programs operating on stacked vectors can use all address sources.

A set of sixteen 32-bit registers is added to the control unit in Series E. Six of the registers are the MDA base registers just discussed. Another eight registers are the return-jump registers (R0 - R7) of the original design. The other two registers can be used as general-purpose registers. The Series E instruction set is augmented with instructions to manipulate these registers.

Control Memory

The control memory of the original design had an address space of 65,536 32-bit words populated with three high-speed 512-word page memories, one high-speed 512-word data buffer and a 16,384-word magnetic core memory. The remainder of the address space could be used to address the memory of a host computer if such an interface exists or to double the capacity of the pages, the high-speed data buffer and/or the core memory.

The page memories hold the micro-program instructions of the system subroutines and user-generated micro code. For some applications page memory space was tight and measures such as executing some micro-code in core memory or swapping

micro-code in the pages were necessary. With the larger memory devices available now it is easy to expand the page memory capacity without increasing their physical size. In Series E, each of the three page memories holds 4,096 words and can be doubled to 8,192 words if necessary. The memory devices are faster so 100-nanosecond instruction fetch rates can be supported (compared to 120 nanoseconds in the original design).

Magnetic-core memory space was also tight in some applications of the original design. In these applications a significant amount of core memory space was used to unload the small MDA memories and/or buffer data on the I/O channels. In Series E the MDA memories are much larger and the I/O channels communicate with the MDA memories directly so the core memory is relieved of this burden. The core memory capacity in Series E is the same as the original design (32,768 words).

Faster Processing Rates

There are about two MDA memory read steps, and one array register transfer for every MDA memory write step in the typical application program. The following table uses this ratio and the read and write cycle times of the original MDA memory and the MDA memories of Series E to show the effect of MDA memory times on the processing rate.

	Original MDA Memory	Series E Bipolar Memory	Series E MOS Memory
Array Register Move Time(nsec)	120	100	100
Read Time (nsec)	120	120	420
Write Time (nsec)	300	160	420
1 Reg.Move + 2 read + 1 write (nsec)	660	500	1360
Relative Processing Rate	1	1.32	0.49

With no changes in the system micro routines, the processing rate of Series E would be close to that of the original design.

The small MDA memory in the original design limited the arithmetic micro-routines to little or no temporary space for their calculations. This had a severe impact on the execution times of the multiply, divide, square root and floating-point operations. With the much larger MDA memory of Series E some of the memory space can now be given to these operations for temporary storage. The micro-routines for these operations were rewritten to use the vector stack in bipolar MDA memory for temporary storage. Some examples of the speed improvement are shown in the following table.

Operation	Series E speed/ original speed
32-bit floating-point add	3.0
32-bit floating-point multiply	4.0
32-bit floating-point divide	2.0
16-bit fixed-point multiply	1.8
16-bit fixed-point divide	1.5

Since the floating-point micro-routines had to be recoded, it was decided to allow other precisions besides single-and double-length. The precision of these operations can then be tailored to match the precision of an attached host computer or to match problem requirements. A maximum precision of 100 bits was selected -- this is large enough to cover most applications and small enough to be handled conveniently in the MDA memory vector stack. No special problems arise if the precision is two or more bits so a minimum precision of 2 bits was selected. Users can adjust the precisions of floating-point operands anywhere in this large range. Operands with different precisions can be combined and results stored with another precision in the four basic floating-point operations: add, subtract, multiply and divide. The execution time and vector stack space used depends on the operand precisions.

To accomodate host computers with different exponent lengths, floating-point operands can have a base-2 exponent with 7 to 11-bits. In the basic operations all operand exponent lengths must agree.

The format of floating-poing numbers in Series E was selected to maximize performance. The format is one sign bit followed by 7 to 11 base-2 exponent bits followed by 2 to 100 mantissa bits. Exponents are biased by 64, 128, 256, 512, or 1024 depending on exponent length. Non-zero numbers have normalized mantissas (the most-significant mantissa bit is always 1). All bits of a floating-point-zero are 0.

The format of fixed-point numbers is the same as in the original design--a two's-complement representation with three or more bits, and any scale factor.

Input-Output

The array modules of the original design had two ways of inputting and outputting data: through the 32-bit common register or through an optional parallel input-output (PIO) unit. The PIO unit had wide ports (256 bits) into each array module and allowed transfer of data at 80 megabyte/second rates. Each array port had 1024 wires (256 twisted-pair inputs and 256 twisted-pair outputs) so the PIO unit was relatively expensive. In some applications the common register path was too slow, inconvenient to use, or required large buffer space in the control memory.

In Series E the array I/O was redesigned. We found that data can be reliably transferred over a 32-bit-wide path at 80 megabytes/second so the high I/O rates supported by the original PIO unit can be accomplished with busses only 1/8 as

wide. The 8-to-1 reduction of bus width through the I/O unit reduces its cost dramatically.

Each array module has a multiplexer-demultiplexer (MPX/DEMPX) to pack and unpack data between the 256-bit-wide internal busses and the 32-bit-wide I/O busses (see Figure 1). A control unit associated with the MPX/DEMPX steals an MDA memory cycle to fetch or store I/O data -- both the access mode and the address come from registers in the MPX/DEMPX control unit.

The I/O busses of the array modules are coupled to a cross-bar to permit data transfers between array modules and I/O to external devices.

[1] J. D. Feldman and L. C. Fulmer, RADCAP - An operational parallel processing facility, AFIPS Conf. Proc. Vol. 43, pp.7-15 (1974 Nat'l. Computer Conference).

[2] K. E. Batcher, The Multidimensional Access Memory in STARAN, IEEE Trans. on Computers, Vol. C-26, no.2, pp. 174-177, Feb. 1977.

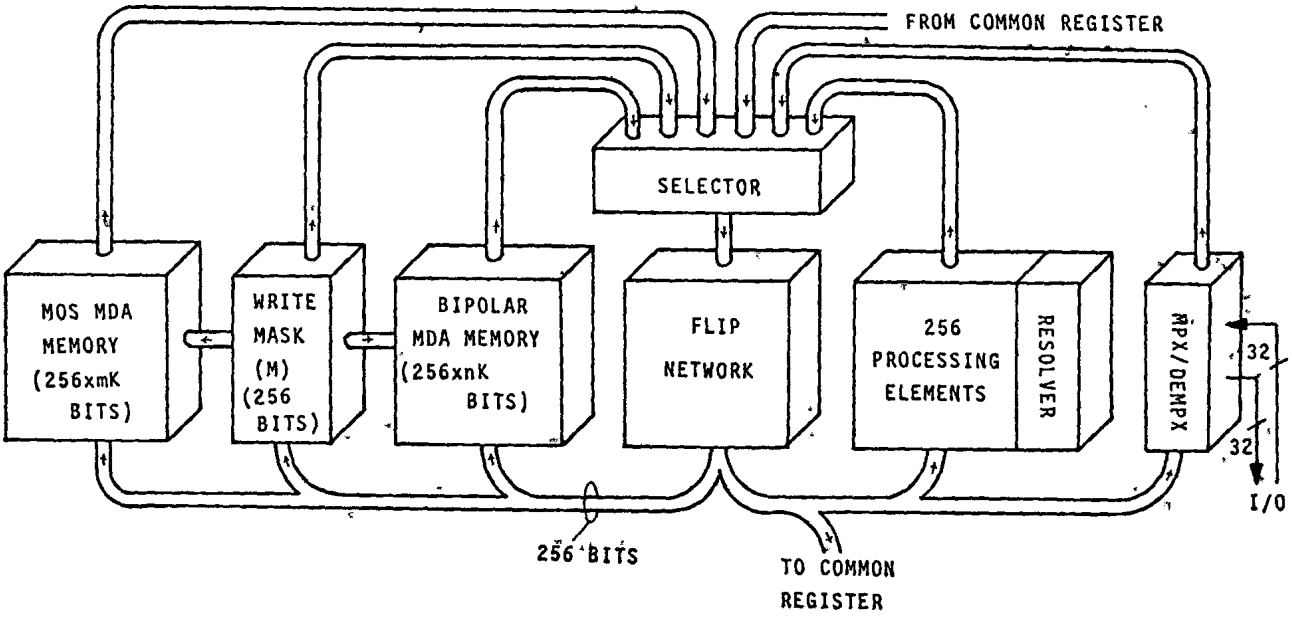

Figure 1 - Series E Array Module

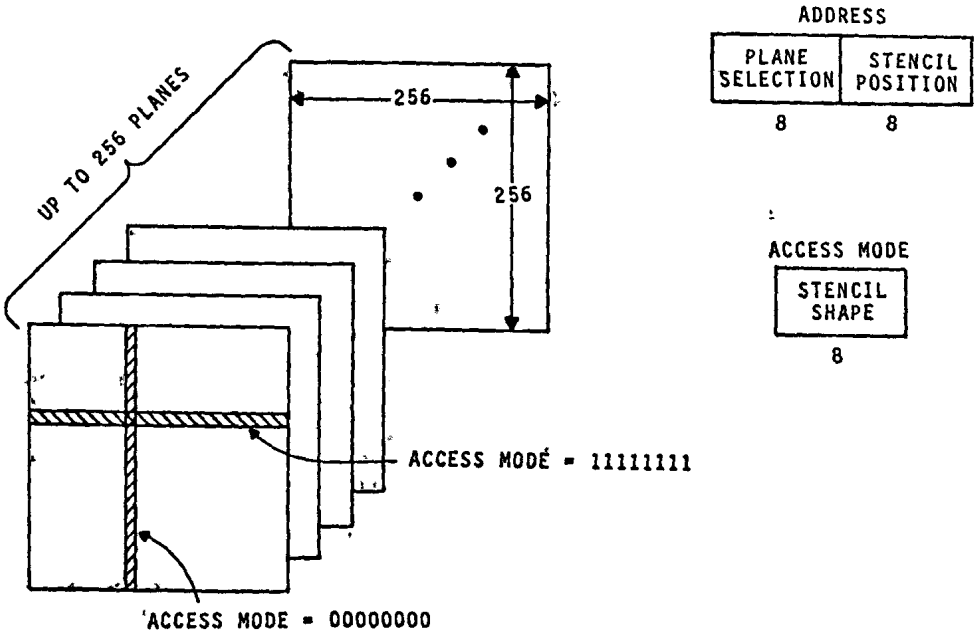

Figure 2 - Accessing Series E MDA Memory