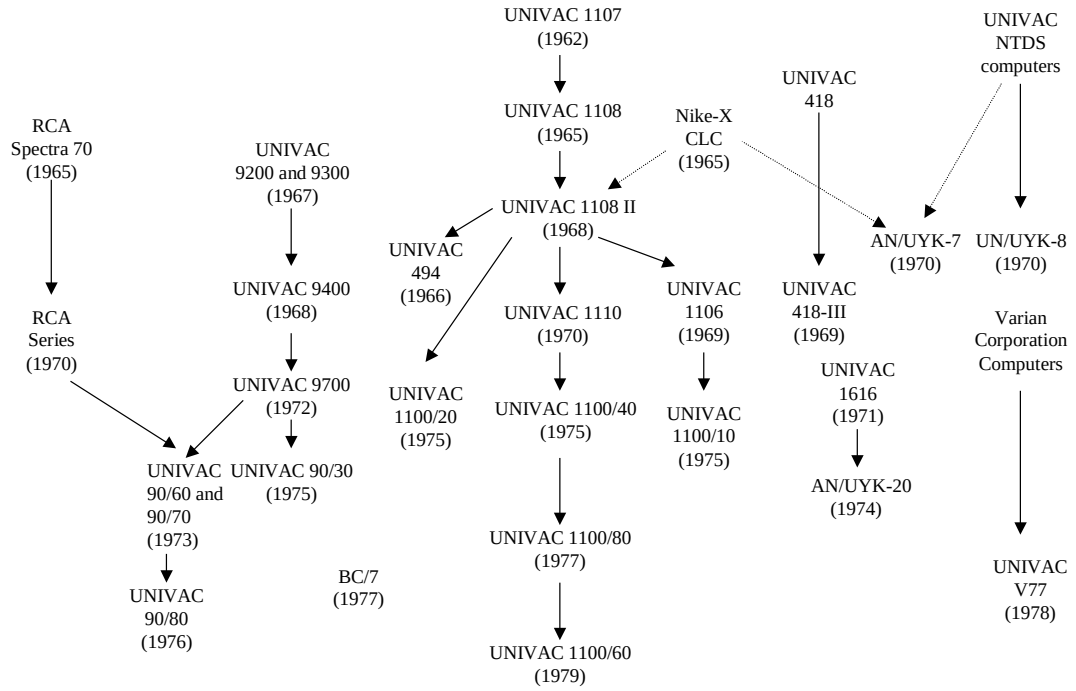
UNISYS:

HISTORY:

- 1873 E. Remington & Sons introduces first commercially viable typewriter.
- 1886 American Arithmometer Co. founded to manufacture and sell first commercially viable adding and listing machine, invented by William Seward Burroughs.
- 1905 American Arithmometer renamed Burroughs Adding Machine Co.
- 1909 Remington Typewriter Co. introduces first "noiseless" typewriter.
- 1910 Sperry Gyroscope Co. founded to manufacture and sell navigational equipment.
- 1911 Burroughs introduces first adding-subtracting machine.
- 1923 Burroughs introduces direct multiplication billing machine.
- 1925 Burroughs introduces first portable adding machine, weighing 20 pounds. Remington Typewriter introduces America's first electric typewriter.
- 1927 Remington Typewriter and Rand Kardex merge to form Remington Rand.
- 1928 Burroughs ships its one millionth adding machine.
- 1930 Working closely with Lt. James Doolittle, Sperry Gyroscope engineers developed the artificial horizon and the aircraft directional gyro – which quickly found their way aboard airmail planes and the aircraft of the fledgling commercial airlines. TWA was the first commercial buyer of these two products.
- 1933 Sperry Corp. formed.
- 1946 ENIAC, the world's first large-scale, general-purpose digital computer, developed at the University of Pennsylvania by J. Presper Eckert and John Mauchly.
- 1949 Remington Rand produces 409, the worlds first business computer. The 409 was later sold as the Univac 60 and 120 and was the first computer used by the Internal Revenue Service and the first computer installed in Japan.
- 1950 Remington Rand acquires Eckert-Mauchly Computer Corp. 1951 Remington Rand delivers UNIVAC computer to the U.S. Census Bureau.
- 1952 UNIVAC makes history by predicting the election of Dwight D. Eisenhower as U.S. president before polls close.

- 1953 Burroughs introduces first 10-key adding machine. Remington Rand introduces UNIVAC 1103, the first commercial use of random access memory (RAM).
- 1955 Sperry and Remington Rand merge to form Sperry Rand.
- 1959 Burroughs pioneers use of magnetic ink character recognition (MICR).
- 1961 Burroughs introduces the B5000 Series, the first dual-processor and virtual memory computer.
- 1965 Sperry introduces the 1108, the first multiprocessor computer.
- 1967 The current OS developed (Exec 8, OS 1100, OS 2200 has been the progression of names, but it's been upward compatible).
- 1976 Sperry introduces first cache memory disk subsystem.
- 1981 Burroughs introduces A Series, forerunner of the current ClearPath HMP NX system.
- 1986 Sperry and Burroughs merge to form Unisys Corporation. Sperry introduces 2200 Series, forerunner of the current ClearPath HMP IX system.
- 1989 Unisys introduces Micro A, the first desktop, single-chip mainframe.
  1993 Unisys introduces 2200/500, the first mainframe based on complementary metal oxide semiconductor (CMOS) technology.
- 1996 Unisys introduces ClearPath Heterogeneous Multi-Processing (HMP), enabling customers to integrate A Series and 2200 Series applications and databases with UnixWare and Windows NT applications and databases on a single platform.
- 1998 Unisys launches initiative to bring enterprise-class capabilities to Windows NT environments. As part of this plan, we announced Cellular Multi-Processing (CMP), which will bring such enterprise-class capabilities as high-speed I/O, partitioning, and cross-bar architecture to Intel-based Windows NT servers.
- 2000 Unisys begins shipping ES7000 servers – the first in the market to take advantage of Windows 2000 Datacenter Server's support for 32-processor scalability.

Family Tree of Sperry Rand Computers
1962-1980



The UNIVAC 1108

Sperry Rand announced the UNIVAC 1108 in the summer of 1964 and delivered the first one in late 1965. Like the IBM 360, the UNIVAC 1108 used a combination of transistors and integrated circuits. Integrated circuits took the place of the thin film memory for the general register stack, giving an access time of 125 nanoseconds, as compared with 670 nanoseconds on the 1107. The 1108's main memory used smaller and faster cores, so that its cycle time (750 nanoseconds) was five times faster than the 1107.[1] The original version of the 1108 had 65,536 words of memory organized in two banks. In addition to the faster components, the 1108 incorporated two major design improvements over the 1107: base registers and additional hardware instructions. The 1108 hardware had two base registers; so that all program addressing was done relative to the values in the base registers. This permitted dynamic relocation: over the duration of its execution, a program's instructions and its data could be positioned anywhere in memory each time it was loaded. Since the base registers had a size of 18 bits, this allowed a maximum address space of 262,144 words. The additional hardware instructions included double-precision floating-point arithmetic, double-precision fixed-point addition and subtraction, and

various double-word load, store, and comparison instructions. The 1108 processor had up to 16 input/output channels to connect to peripherals. The programming of these channels was done with specific machine instructions, and there was no capability to build multiple-step channel programs.

Just as the first UNIVAC 1108s were being delivered, Sperry Rand announced the 1108 II (also referred to as the 1108A) that had been modified to support multiprocessing. This development arose from St. Paul's experience on the Nike-X computer. The Athena missile guidance computer of 1957 had been the basis for the Target Intercept Computer (1961) that was used in the Army's Nike-Zeus antiaircraft missile. **When the Army was authorized in 1963 to develop the Nike-X anti-ballistic missile (ABM), St. Paul received the contract from Bell Telephone Laboratories to provide a computer for its guidance and control system. The Central Logic and Control (CLC) module was composed of multiple processors (a maximum of ten), two memory units, and two input/output controllers (IOCs). Unlike the 1100 Series, the CLC used twos-complement arithmetic (which was chosen by Bell Labs) and a 32-bit word size for its registers. The memory units were for program storage and data storage, each holding up to 262,000 64-bit words. The CLC could be operated as one computing environment or be dynamically partitioned into two. It was completed in 1965 and machines were delivered to the White Sands Missile Range in New Mexico, Bell Telephone Laboratories, and Kwajalein Atoll in the Pacific Ocean test range. The first missile firings were in November 1965, and the ABM program, later renamed Sentinel and then Safeguard, continued until 1975 when it was terminated because of the restrictions of the ABM treaty with the Soviet Union and the enormous cost of its radar and communications components. It was estimated that deployment at just six sites would have cost $40 billion. Various features of the CLC were adapted for use in the 1108.**

A multiprocessor 1108 could have up to three CPUs, four memory banks totaling 262,144 words, and two IOCs. (A system with four CPUs and three IOCs was installed at United Airlines, but that configuration was never again offered to customers.) The IOC was a separate processor, functionally equivalent to the I/O channel section of the CPU, which could take over the task of handling I/O. If an IOC was used, it connected to one of the channels in the CPU. Then the CPUs could load channel programs into the IOCs. Since the IOCs had their own paths to memory, once a CPU issued an I/O request, the IOC took full control of the I/O, transferring data to or from memory without further intervention by the CPU. Each IOC had up to 16 channels. Thus, an 1108 multiprocessor could be a very busy system. At the maximum configuration, five activities could be taking place at any given

moment: three programs executing instructions in CPUs and two I/O processes being performed by the IOCs. The test-and-set instruction, developed on the CLC, was added to provide synchronization between processors, giving a total of 151 instructions in the instruction set.

Happily, the UNIVAC 1108 reversed Sperry Rand's decline in the large computer market. The first 1108 was shipped to Lockheed in Sunnyvale, California, toward the end of 1965. Lockheed had already installed an 1107 as an interim machine and ultimately replaced two IBM 7094s with two 1108s. Other early 1108 orders came from the French National Railroad, the Scottish National Engineering Lab, Boeing, the **Naval Ordnance Test Station**, NASA (three in Huntsville, Alabama, two in Slidell, Louisiana, and four in Houston), the University of Utah, the U.S. Environmental Sciences Services Administration, Air France (two machines), the Census Bureau, Carnegie Mellon University, and Air Force Global Weather Central (4 machines). Choosing the 1108 over GE and IBM proposals, Clark Equipment Corporation installed two to replace its UNIVAC File Computer, leaping straight from first generation hardware to third generation. The National Bureau of Standards chose an 1108 rather than a Control Data 6600, because of the 1108's superior remote communications capabilities and lower price. The relative smoothness of many early UNIVAC 1108 installations contrasted sharply with various well-publicized delays for the IBM 360. The success of the 1108 was a wonderful surprise to Sperry Rand: in 1964 an internal study had forecast that only 43 would be sold.[2] The January 1967 issue of *Datamation* had a very favorable article by Douglass Williams of Lockheed describing its 1108 installation, and by the end of 1967 the total number of 1108 orders was over 135. Ultimately, Sperry Rand produced UNIVAC 1108 systems amounting to 296 processors.

While both EXEC I and EXEC II were provided for the unit processor 1108s, it was clear that the two should be merged to provide a true multi-programming system with the ease of use and external appearance of EXEC II. Furthermore, the multiprocessor 1108s needed an operating system. This new operating system was EXEC 8, a name sometimes written as EXEC VIII. The specifications for it were drawn up in December 1964, and work began in May 1965. The announcement of EXEC 8 in 1966 was greeted with skepticism by *Datamation*, which had seen many big software fiascoes by other computer companies: "A step towards the quicksand: Univac, which has been doing well with about the only working large-scale software, joins the mañana crowd with a new operating system for the 1108." At first, *Datamation* had it pegged correctly: the initial versions of EXEC 8 did not work very well, and in 1967 Sperry Rand had to give one of the 1108s to NASA for free as a penalty for missing contract deadlines. The situation did improve. The University of Maryland installed

its 1108 in October 1967 and by February of 1968 was controlling three remote 1004 computers and six teletype time-sharing terminals under EXEC 8. In 1969, the president of Computer Response Corporation, a service bureau with an 1108 in Washington, could say of EXEC level 23.25: "We're satisfied with the way it's handling our workload." [3] However, EXEC 8 wasn't fully settled down until 1970.

Sperry Rand was also fortunate to have a good FORTRAN compiler and some program conversion tools. The FORTRAN V compiler for the 1108, written by Computer Sciences Corporation, produced very efficient programs. At a meeting of Burroughs engineers discussing their competitors, Robert Barton referred to it as "a polished masterpiece" and another participant said: "You sit there and watch the code that thing cranks out and just try to *imagine* assembly code that would be written that well." Lockheed developed a "decompiler" which translated IBM 7094 machine language programs into NELIAC. One of these decompiled programs comprised 500,000 instructions. There already was a NELIAC compiler for the 1107 (and later for the 1108). At Air Force Global Weather Central these doubly translated programs ran much faster on the 1108 than IBM's 7094 emulation did on the 360. Sperry Rand had a program that translated 7094 assembler programs to 1108 assembler, and Boeing developed another program that converted IBM 7080 Autocoder programs to the 1107 and 1108. The 1108 did well in competitions: a single processor 1108 outperformed an IBM 360/65 and a GE 635 on benchmarks done for the University Computing Company in 1968.[4]

Many programmers who came to the 1108 after working on the machines of other computer companies were struck by how easy it was to work with EXEC 8. Steve Seaquist, now a self-employed programmer, started out on a Control Data Corporation (CDC) 6600 at the University of Texas in 1967. The 6600 was fast, but the students were taught to use an assembly language simulator, written in FORTRAN. In 1969, he transferred to the University of Maryland and used the 1108. Seaquist's first class was 1100 assembly language, and he was amazed that students were allowed to write real assembler programs. Seaquist said he "fell in love with the 1108." He liked the fact that batch and time-sharing (which was called "demand" processing on the 1108) used the same commands in the Executive Control Language (ECL), and that the consistency of ECL made it easy to compile and test programs. He got a job as the computer center librarian, but quit the next summer to work as a lifeguard, because that had higher pay.[5]

Sperry Rand implemented the CLC and 1108 multiprocessor architecture in its line of military computers. In response to a 1962 Navy specification for a computer with a 36-bit word, St. Paul developed the single processor CP-667, which was delivered

in March 1964. Like the 1108, it used a mixture of transistors and integrated circuits. It had a compatibility mode which allowed it to run programs written for the 30-bit NTDS computers. The CP-667 fell victim to bureaucratic politics within the Department of Defense, and none of the U.S. military services ordered any. However, in December 1967, the company was awarded a contract by the U.S. Navy to develop a multiprocessor successor to the NTDS family of computers. This was the AN/UYK-7, which was frequently configured with three CPUs, two input/output controllers and 262,144 words of memory. The design was a blend of the CLC and the 1108, with a word-size of 32 bits, and instructions in both 16 and 32 bit lengths. The first AN/UYK-7 was delivered in April 1969, and it became the basis of the Navy's AEGIS ship defense system. An airborne version, designated the 1832, was used in Navy anti-submarine aircraft. By January 1974, the AN/UYK-7 was being used in 32 different Navy Projects. Sperry Rand also produced the AN/UYK-8, which was a compatible (30 bit word) dual processor replacement for the NTDS machines.[6]

The 1106 and the 1110

The 1108 was not a series or family of computers. A customer could get a unit processor machine or, in the multiprocessor version, expand up to a maximum of three CPUs and two IOCs, but that was it. There was no small model. The unit processor 1108 for Carnegie Mellon University, as an example, cost $1.8 million, $1 million of which was covered by a grant from the Richard King Mellon Charitable and Educational Trust. Not everyone had a charitable and educational trust, so it was clear that there needed to be a less expensive entry into the 1100 world. Sperry Rand announced the 1106 in May 1969 to meet this need. The first few machines shipped as 1106s were really 1108s with a jumper wire added to the back panel to introduce an additional clock cycle into every instruction. Astute customers soon learned which wire they had to clip to speed up their 1106s. The real 1106 used a slower and less expensive memory that had a cycle time of 1500 nanoseconds, half the speed of the 1108, and was packaged in 131,072-word modules referred to as unitized storage. On a system with just one memory module, it was not possible to overlap the operand access of one instruction with the fetch of the next instruction so the basic add time was 3000 nanoseconds. Systems with two modules could do the overlap and achieve faster operation. Later on, a faster memory unit was built in 32,768-word modules, and systems that used that memory were called the 1106 II. A single processor 1106 sold for around $800,000 which still was not entry level, but was considerably less expensive than the 1108. Sperry Rand sold 1106 systems amounting to 338 processors.

These were prosperous times for Sperry Rand, given the success of the 1108 and the 1106. One key exception to this bright picture was the failure of the airline reservations project at United Airlines. Sperry Rand and United had embarked on a joint venture at the beginning of 1966 to develop an airline reservations system based on the 1108, and a three-CPU 1108 II was installed at United's Elk Grove Center outside of Chicago two years later. By the summer of 1968, the problems with EXEC 8 had already put the project six months behind schedule and trouble continued. The original specifications were overly ambitious, and they kept changing as the project went along. The project also got bogged down in making extensive modifications to EXEC 8 that eventually amounted to half the code. Even with additional processor power (a fourth CPU), the system was unable to meet the goal of 39 transactions per second, reaching only around 10. United terminated the project in the spring of 1970, and purchased the IBM-based PARS software which had been developed at Eastern Airlines. United decided to keep the 1108 and use it for message switching, materials control, and flight information; this system, called UNIMATIC, is still running today. Although UNIVAC did regroup and get a reservations system based on the United project going successfully at Air Canada two years later, the failure at United was a significant lost opportunity. (It should be noted that, independently, Air France wrote its own operating system and reservations software for the 1108 and put its reservations system into operation in September 1969.) On the bright side, 1970 saw the implementation on two 1108s of the automated stock market quotation system for the National Association of Securities Dealers (NASD). This project, coordinated by Bunker-Ramo, went into operation in January 1971 and has run on 1100 (and its successors, the 2200 and ClearPath IX) series computers ever since.

A set of UNIVAC 1100 account profiles from the early 1970s makes it possible to take a closer look at this time when companies and government agencies were actually switching from IBM and other vendors to UNIVAC computers. The account profiles cover thirty-five 1106 and four 1108 sites where a new computer had been acquired in the early 1970s. There were eleven government agencies (local, state/provincial, and national), three universities, seven utility or communications companies, a savings and loan, and a newspaper. The other sixteen were various manufacturing and business enterprises. The UNIVAC replaced an IBM 360 or 370 at fourteen of these customers, Honeywell/GE computers at five, RCA at four, and Burroughs at two. The RCA replacements arose from Sperry Rand's purchase of the RCA customer base in 1971. They are still significant since those four companies could well have chosen IBM to stay with that architecture instead of switching to the 1100. In two of these sites, the customer was converting from a UNIVAC III to an 1106. Conversions to the 1100

continued through the 1970s, and information on some of them can be found in various papers presented at the semi-annual conference of the USE user group for 1100 sites.

Various strengths of the 1100 helped make these sales. EXEC 8 was superior to IBM's OS and DOS in several areas, including scheduling, the ability to handle a mix of batch and demand runs, time-sharing capabilities, and the simplicity of Executive Control Language (ECL) as compared with IBM's Job Control Language (JCL). Programmers who had worked only on IBM computers sometimes thought they were being tricked when they were first shown an ECL runstream: it *had* to be more complicated than that. Greg Schweizer, a programmer at the Portland *Oregonian*, started out on the IBM 360 as a student at Washington State University and at his first job. The first time he used a UNIVAC 1100 was in the mid-1970s when he started work at the State of Washington, which was converting from an RCA Spectra 70. He was impressed by how much easier it was to work on the 1100; his first reaction to ECL was: "This is fantastic; why couldn't IBM do this?" On IBM the complexity of JCL led to the frequent embarrassment of having to re-run jobs because of JCL errors. At his previous company, Greg had been struggling for weeks to get an IBM CICS transaction program to work, and it still wasn't working when he left. On the 1100 he found that "UNIVAC knew how to do transactions." It was easy to write transaction programs with UNIVAC's Transaction Interface Package (TIP), a generalization of the routines used at Air Canada.[7] The existence of two operating systems (DOS and OS) was another disadvantage for IBM. Customers who wanted to move up to larger models in the IBM 370 hardware line were faced with a laborious conversion from DOS to OS, and some chose to convert to other vendors. By this time, Sperry Rand was finished with its move from EXEC II to EXEC 8, and EXEC 8 had settled down to be a stable operating system.

UNIVAC computers had an advantage in their multiprocessor architecture, an area in which Burroughs was the only other serious contender. This permitted easier, incremental hardware upgrades and was the beginning of the road toward today's fully redundant systems. At this point, IBM was still several years away from delivering effective multiprocessor machines. This, combined with the scheduling flexibility of EXEC 8, meant that the 1106 outperformed IBM 370/135 and 370/145 computers in benchmarks done for several of these customers. Another area of advantage for the UNIVAC was remote job entry (RJE) capabilities. In 1964 the 1107 at Case Institute of Technology in Cleveland had been linked to a 1004 at a hospital ten miles away, and the following year another 1004 one hundred miles away was also connected. By the end of the 1960s this capability was widely used, although 9200 and 9300 computers had begun to displace the 1004 as the preferred remote device. One of these new customers

tied its 1106 in Missouri to remote 9200/9300s in Houston, Fort Worth, and Kansas, while the State of Georgia implemented a network of an 1106 connected to fourteen 9200s spread across the state. Sperry Rand did not have the advantage in every area, as IBM was clearly ahead in disk drive technology. The 1100 series had just started using disks (as opposed to drums) in 1970, and the 8414 disk was a slow performer compared with IBM's 3330s. One of these customers had severe problems with its 8414s.

In the area of software, the availability of Sperry Rand's DMS-1100 database system was a factor in twelve of these sales. While still in a very rudimentary form, it provided greater data handling capability than IBM's IMS. General Electric (and then Honeywell, after it acquired GE's computer business) was a more serious contender with its Integrated Data Store (IDS) developed by C.W. Bachman and others in the mid-1960s. Both IDS and DMS-1100 had the additional glamour of complying with the database standard of the Committee on Data Systems Languages (CODASYL), while IMS did not. Demonstrations of time-sharing programs accessing DMS-1100 databases impressed several of these customers and they became early users of it. At two other companies, an older data management tool, FMS-8, was a key factor in the choice of the 1100. Since so many of these sales involved conversions, it is not surprising that conversion software, such as a 1401 simulator, a COBOL translator, and an IBM assembly language (BAL) to COBOL translator, played an important role. At the time of these sales, few computer users had ventured far into transaction processing and screen formatting. This meant that most COBOL or FORTRAN programs were batch-oriented and thus relatively easy to convert. Sperry Rand's edge over IBM in easy time-sharing access also facilitated program conversions: program card decks could be read into disk files and changed with the ED processor, which seemed very powerful at the time, particularly on the Uniscope 100 and 200 screen terminals. In 1975, the General Atomic Company converted approximately 800 COBOL programs from IBM to the 1100 over a nine-month period with a staff of eight programmers.

Situations like the one at United Airlines showed that something more powerful than the 1108 would be needed for very large applications. Work on a bigger system began in the late 1960s, but it was delayed by various engineering design problems as well as difficulties in establishing business relationships with integrated circuit manufacturers. Sperry Rand did not have the resources to build its own integrated circuits in the quantities needed and had to buy them from Raytheon, Fairchild, Motorola, and Texas Instruments.[8] These problems were worked out, and the UNIVAC 1110 was announced on November 10, 1970. The announcement had been delayed for several weeks so that it could happen on the date, 11-10, which matched its name. The 1110 processor was

constructed entirely of integrated circuits, but they were only about 25 percent faster than the transistors used in the 1108. The design of the 1110 incorporated several features to give it greater throughput than the 1108.

The first was the use of plated wire memory. Plated wire had already been used in the 9000 series, but it was too expensive to use for all the memory needed in the 1110. Therefore, the 1110 was designed to have a relatively small amount of "primary" memory using plated wire and a larger amount of "extended" memory using core. The plated wire memory had a read cycle time of 300 nanoseconds and a write cycle time of 500 nanoseconds; it came in cabinets of 65,536 words and up to four cabinets could be used in a full system. The core memory had a cycle time of 1500 nanoseconds and came in 131,072 word cabinets, with a maximum of eight cabinets (1,048,576 words) on a full system. Elaborate algorithms were added to EXEC 8 to move programs between primary and extended memory depending on their relative compute to I/O ratios. The processor base addressing registers were expanded to handle 24-bit addresses, and the number of registers was increased from two to four so that a program could have four banks based at one time.

The 1110 also increased throughput by having separate input/output processors and more instruction processors. Following the method used on the 1108 II, all 1110s had separate input/output processors called IOAUs (input/output access units) to handle I/O operations. The CPUs, which no longer had any I/O capability of their own, were called CAUs (command-arithmetic units). As originally announced 1110s ranged from a minimum of one CAU and one IOAU (a 1x1 system) up to four of each (a 4x4). Later, the capability to go up to six CAUs was added. Each IOAU contained up to 24 channels. Another feature was increased instruction overlap: on the 1110 instruction overlap was increased to a depth of four instructions. This made the design more complex because of the need to check for conflicts: if one instruction changed the value in a register which would be used to index a memory access in the next instruction, then that instruction's operand fetch had to be delayed until the register value was established. The 1110 also added 24 byte-handling instructions to the instruction set to improve the execution speed of COBOL programs.[9]

Sperry Rand planned for the UNIVAC 1110 to be a competitor for the high end of the IBM 370 series. **Accordingly, a 2x1 1110 rented for about $60,000 to $65,000 per month, which was about $10,000 less than an IBM 370/165.** At first, the response was disappointing: there were only six orders during the first year.[10] The pace began to pick up as 1110s replaced or supplemented 1108s at existing customers (Lockheed, Air Force

Global Weather Central, Shell Oil, and the University of Wisconsin) and added some new customers as well. The Environmental Protection Agency's center in North Carolina, which had been an all-IBM site, got a 2x1, as did Arizona State University, where the 1110 replaced a Honeywell (General Electric) 255 and some smaller computers. Both Arizona State and the University of Wisconsin installed 1110s in 1973. Arizona State was trying to do really large scale time-sharing using several new software packages, and it encountered severe problems during the first few months while the bugs were being worked out, while Wisconsin fared better using older software.[11] Shell Oil made extensive use of the 1110, having three 4x2 systems in place by the end of 1975. In all, 290 1110 processors were produced.

The 1100 Series During the 1970s

During the early 1970s metal oxide semiconductor (MOS) memory chips became available as a replacement for core memory. IBM used them on its 370/145, which was introduced in September 1970. Sperry Rand followed suit in 1975 and 1976 by bringing out semiconductor memory replacements for the 1106, 1108, and 1110 called the 1100/10, 1100/20, and 1100/40. The maximum memory on the 1106 and 1108 had been limited to 262,144 words, but changes to the size of fields in the addressing structure increased the maximum to 524,288 words on the 1100/10 and 1100/20. Cable length considerations had confined the 1110 to 262,144 words of primary storage. The use of bipolar memory chips made the memory more compact than plated wire, so in the 1100/40 the maximum was increased to 524,288. The 1100/40, like the 1110, could have 1,048,576 words of extended storage, and the use of semiconductor chips made it faster, cutting the access time from 1500 nanoseconds to 800. In this new naming convention for these models, using the slash, the last digit represented the number of instruction processors on a particular machine, so that an 1100/40 with three CAUs would be referred to as an 1100/43.

Sperry Rand had started work on a follow-on system even before the 1110 was shipped to customers. It was originally referred to internally as the 1112, but then things got complicated. In Chapter 7, we saw that St. Paul had taken its NTDS computer and produced commercial versions of it as the 490, 491, and 492. After the introduction of the 1108, this series was upgraded with 1108 components to become the 494 which was introduced in 1966. The 494 sold well, with 125 machines having been shipped by 1976. They were used to run airline reservations systems at Eastern, Northwest, British European Airways, Iberia, and Lufthansa. However, the company was faced with the burden of supporting yet another line of software (operating system, compilers, and utilities) for it. An internal report in 1969 said: "Although we seem compelled to continue to invest millions of dollars in 494

software, this will not contribute to UNIVAC growth. UNIVAC must solve the problem of the 494. In the meantime, its proliferation especially in the software area is robbing limited resources from other efforts."[12] St. Paul decided to have the 1112 provide a 494 emulation mode, and it combined 1112 and 495 (a better 494) to come up with 1195. Fortunately, by the time it was announced the name had changed to the 1100/80 to fit into the new numbering scheme. The 1100/80 used a new circuit technology known as emitter coupled logic (ECL), which brought about a considerable increase in speed. The considerations of chip placement on boards and the wiring connections among them had become so complex that the engineers developed new software programs to do the circuit designs.

The 1100/80 was the first 1100 to use cache memory. (IBM had introduced cache memory on its 360/85, announced in 1968). This was a relatively small amount (maximum of 16,384 words) of very fast (45 nanosecond access time) memory in a separate module that could be accessed by any processor in the system. On any reference to memory, the hardware would first check to see if the request could be satisfied from cache; if not, eight words at the main memory address would be read into cache and then the requested item passed on to the processor. The use of cache memory and faster components made the 1100/80 about twice as fast as the 1110. The original 1100/80 could have one or two instruction processors, one or two input-output processors, and four million words (4 MW) of memory. The later version (called the 1100/80A) could have up to four of each type of processor. The 1100/80 introduced industry standard byte- and block-multiplexor input/output channels. The 1100/80 was first delivered in 1977 and was a very successful product: over 1,000 processors were eventually delivered.

While the 1100/80 was being developed, researchers at Sperry Rand's Corporate Research Center in Sudbury, Massachusetts established the feasibility of using multiple microprocessors to build a mainframe computer processor. St. Paul started a project under the code name Vanguard to design a new 1100 processor using Motorola 10800 microprocessors. This design turned out to have a significantly lower cost, and the designers decided to enhance the reliability of the system by totally duplicating each instruction processor and having the two halves check each other. This was a return to the concept used in the BINAC and the UNIVAC I. The lower cost made it possible to bring out a smaller, less expensive machine intended to broaden the user base of the 1100, and this is exactly what happened. The Vanguard was announced on June 5, 1979 as the 1100/60, and its first delivery was later that year. Its availability coincided with the first widespread use of the MAPPER software, which provided a simple database and reporting capability. The combination of MAPPER and the lower

price brought in many new customers. Sperry Rand exceeded its sales target in the first year, shipping systems amounting to 528 processors. The original model provided for a maximum of two instruction processors and two input/output processors (2x2), but this was subsequently increased to a maximum of four of each (4x4). The switch to a denser main memory in 1981 was the occasion for changing the name to the 1100/70. Between the 1100/60 and the 1100/70, nearly 4,000 processors were delivered.

The MAPPER software package originated on a UNIVAC 418 computer being used to keep track of the Sperry Rand factory production line in Minnesota. The software, called RPS, made it possible for anyone connected to the 418 to monitor the status of production and to print status reports. In the early 1970's, a new corporate policy required that internal use of the 418 be discontinued. A software development group started working on a new version of RPS for the 1100, but they took a very ambitious approach, basing their product on use of the DMS-1100 database software. The factory users of RPS, fearing that RPS-1100 would be slow, difficult to learn and present a difference appearance on the terminal, decided to do their own rewrite. Since the name RPS had already been given to the new product, they called their version MAPPER for Maintaining and Preparing Executive Reports. RPS-1100 was released as a product in December 1974, but it never caught on in the customer base. The factory used MAPPER and was happy with it. Over the next few years, several existing and prospective customers who were touring the factory saw MAPPER and wanted it for themselves. For a time, Sperry Rand resisted these requests, but when the Santa Fe Railroad asked for MAPPER to keep track of its freight cars (and proposed to make a large 1100 purchase) the company gave in. Santa Fe started using MAPPER in 1976 on an exception basis, and MAPPER was announced as a product in the fall of 1979.

Subaru of America was another one of the major early MAPPER customers. In 1979, Subaru selected a UNIVAC 1100/60 to replace its 90/30. Sperry Rand narrowly beat out IBM in the competition, because IBM's 4300 computers (replacements for the smaller 360s and 370s) weren't quite ready. Subaru, however, took the plunge into the 1100 world, interested in the potential of MAPPER. Sperry Rand delivered an 1100/61 in 1980, and Subaru started out as an all-MAPPER environment. Bill Krewson, Subaru’s database administrator, was impressed with MAPPER: "It was so much easier to deal with than the IBM and 90/30 environment. We wondered: why doesn't everybody do it this way?" MAPPER was so easy to use and such a big consumer of machine resources that the 1100/61 was swamped within six months, and an 1100/62 had to be installed at Christmas of 1980. Subaru continued to be a major MAPPER user, integrating it with the 1100 relational database software (RDMS), and kept on moving up into bigger 1100 computers over the

following years.[13] MAPPER was used extensively at both large and small 1100 customers. A survey of 224 customers done in 1989 found that 140 of them were using MAPPER.[14]

In 1979, Sperry Rand changed its name to Sperry Corporation, but the computer division continued to be called Sperry UNIVAC and the computers still used the name UNIVAC. Throughout the 1970s the company had maintained its position as the second-place producer of large-scale computers. Particularly during the first half of the decade, it was able to attract customers who converted from IBM machines, and in both the 1100 and 90 series it offered products which many customers believed were superior to IBM in ease of use. While the 90/30 provided a good mid-size business computer, Sperry Rand's failure to establish a large market share in minicomputers and small business computers left it in a vulnerable market position at the beginning of the 1980s.

DESIGN GOALS:

- The design of OS 2200 can be attributed to:

  1. A commercial OS designed to support systems that would exist for many years.

- The design goals where:

  1. To support a mixture of the most demanding real time processing (it flew missiles for example and ran the Air Force message switch) in a mixture with batch and timesharing processes.

  2. To support multi-activity (multi-threaded) user applications (from the beginning even the FORTRAN and COBOL languages had built-in functions and libraries to permit multi-threaded applications).

  3. To be "B1" security certified.

OS DESIGN:

The Exec (also known as the Executive System) is a software supervisor that controls the OS 2200 system operating environment. Largely resident in memory, the Exec processes user runs, controls files, manages system resources, and performs input/output operations for users.

The Executive, or Exec, is the principal interface between the user and the system as a whole. It is responsible for such functions as time and space allocation of system resources;

first-level I/O control and interrupt answering; logging of system accounting data; first-level debugging assistance; and protection against undesired interaction of users with other users of the system. By presenting a relatively simple interface, it allows the programmer to use the system easily, while relieving concern for the internal interaction between the program and other coexistent programs.

CPU SCHEDULING:

The earliest systems had a single dispatcher queue split into multiple priority Types. Each Type was then split into Levels and the overall priority of a process was referred to as TAL (Type and Level). The TAL could be arithmetically compared to the value in another activity's state to determine if it should be allowed to pre-empt the running process.

Current systems use a priority-based scheme (a single priority number), but with special handling of certain ranges (of priorities). In addition several "fairness schemes" are included:

- To ensure that compute-bound batch programs make some progress even if highly-interactive programs were capable of using up all the CPU time.

- To allow sites to define the percentage of resources available to each of the job classes.

- To optimize cache miss rates on the very large SMP's of today. With 32 processors, megabytes of on-chip cache and more megabytes shared by sets of 4 processors, it's really important to keep threads in the same cache even if they've been interrupted by other processing. The performance benefit can run to 15% or more. Of course, you then have to have other algorithms to be sure that real time work doesn't have to wait (Microsoft has some similar capabilities in Windows DataCenter, but expects the site administrator to statically manage the allocation of each executable to a subset of the processors. OS 2200 does it dynamically and also handles re-allocation if a processor fails and is dropped by the system (may not even take a reboot)).

MEMORY MANAGEMENT:

The 1100/2200 series was late to true virtual addressing. The earliest systems used a mechanism called banks which had evolved from simple memory performance desires. In the earliest machines memories consisted of multiple banks which could be accessed in

parallel and it was desirable to have instructions in one bank and data in another so that a single instruction could be executed in one machine cycle. Because memory was the most expensive and limited part of the system, UNIVAC software (e.g., compilers) and many customer applications were written with re-entrant instruction banks that could be shared by multiple jobs.

Later systems introduced instructions to switch banks and the idea of shared data banks (e.g., for database managers). Thus, the system "paged" (swapped) whole banks into and out of memory. Real paging didn't come until that late 1980s.

The bank is still used as the primary application-visible entity. Thus, we have a segmented virtual address space that is in some ways similar to that of the current Intel x86 chips. However, our segments are mapped to a 54-bit "absolute" address space and it is that absolute space that is actually paged (16K bytes pages). With the large memories on today's systems (64GB) real page fault interrupts are very infrequent but paging remains the most convenient way to manage the large memories.

We have a large number of acceleration techniques for handling memory loading. The Exec will try to determine if a whole bank should be loaded on the first page fault, or if it should be brought in a bit at a time. The site can influence this decision with various linking commands.

The system supports full dynamic linking to libraries and modules of user applications, but real time and transaction programs are usually statically linked internally to avoid the overhead. Modules, like libraries, and user-written shared segments, that are dynamically linked may also be replaced in a running system. There are mechanisms to hold job scheduling until a shared segment is not in use, load the replacement, and then allow scheduling.

We currently try to spread pages equally across all available memory modules to improve performance. If we need to remove a memory module, the Exec will move all pages out of that module to some other memory module with no impact on running programs.

The system is fully dynamically partitionable. Processors, channels, I/O processors, and memories or even memory address ranges may be added or removed from a running system without a reboot. This may be as a result of error handling or just partitioning commands from the operator.

The customer may set aside a portion of the 54-bit absolute address space to hold "memory-resident" files. That space is

paged along with all the other users of real memory, but will tend to stick if in use. The two primary uses are for scratch files (e.g., compiler temporary space) and for high-access-rate transaction files. Transaction files are usually duplexed with a disk copy so that while reads all come from memory, writes go to both the memory and the disk copy. The disk copy may then itself be mirrored by the disk subsystem. If files use up more than the set-aside space, they automatically use disk for any remaining requirements.

FILE SYSTEM:

The OS 2200 file system is at heart a very simple flat-file system (fast and simple).

Mass storage is generally treated as a pool of space, much as main memory is. When you create a file, you specify the name and how much space you want initially allocated and to what size it may grow dynamically. **OS 2200 attempts to find contiguous space on a single device at least able to hold the initial space.** If it cannot find such space, it is able to allocate space across any number of devices. **When a file expands, typically by just writing to the next higher address, OS 2200 will attempt to place the expansion adjacent to the previous address but is able to place it anywhere on any device.** Space is managed in units of approximately 8K bytes although larger chunks are usually allocated for better performance.

Just as main memory is paged, so to some extent is the file system. The file system itself is a virtual pool of space. Our file backup utility records information in the file directory with the tape number(s), time of backup, physical start position on the first tape, etc. If mass storage starts to get used up, OS 2200 will look through the file directory and find files that have current backups and haven't been used recently. They are then marked as unloaded and their space made available for re-use. If anyone does access those files, their "Open" request is held while the file is brought back from tape and placed, probably somewhere different than before, on mass storage.

The file directory itself is spread across all the disks. It contains the file names, security information, list of locations of the file contents, backup information, etc.

There is one level of hierarchy. Some files are what are known as "Program files" contain elements which are usually program modules. Elements are distinguished by name and type. For example, it is quite possible to have at least 3 elements in a program file with the same name: Mainprog a COBOL text module, Mainprog a compiler output module, and Mainprog an executable. Text modules are even sub-typed by the language so that build routines know which compiler or other program to call.

All files, including program files, are treated the same by the low-level I/O system. You specify the file name, file-relative offset of the start of the area, and the length of the area. All access methods are provided by libraries, not the OS, for most standard files.

The OS does provide one set of built-in access methods. These are all variants of fixed record size, random access files. They include record locking and full journaling support. This is used mostly by the most demanding transaction environments for the highest performance.

The original native character set of Exec 8 is something called FIELDATA. FIELDATA was invented by the Army Signal Corps as a 6-bit character set to replace the older 5-bit teletype character sets (like BAUDOT) and gets its name from its function - sending data back from the field. Virtually all computer systems of the late 1950s through the mid 1960s supported FIELDATA as you couldn't otherwise sell to the Department of Defense. IBM supported FIELDATA on their 7000 Series (7090, 7040, 7094) but also supported BCD as being more appropriate to business processing.

OS 2200 also supports many other character sets, with 8-bit extended ASCII being the one used most frequently today. However, you'll still see references to FIELDATA in a lot of places. Since the 2200 systems use a 36-bit word (also mandated by the DOD), the internal representation uses 9 bits for 8-bit codes (and 18 bits for 16-bit codes). There is some user software that's found a use for the extra bit, but that's discouraged as it impacts data portability.

OS 2200 has a disk scheduling algorithm for boom movement optimization. When we initially implemented the algorithm, we did some analysis of patterns. We determined that we couldn't simply order requests by closest position as that would lock some requests potentially forever. So at first, we ordered requests so that we always kept the boom moving in the same direction and didn't permit more than 2 same-position requests before moving on. However, even that turned out to not provide the service patterns that our customers required. So we finally adopted a saw-tooth pattern. Requests are ordered so that we keep the boom moving from the edge towards the center. Once it reaches the center, we jump back to the edge again. Real time and High Exec requests override boom optimization and sites can turn it off in general if they find it to be counter-productive.

Very few sites are using the boom position optimization anymore as they all have some form of cache disk. With a cache boom optimization doesn't work. The computer can't even tell which requests are going to cause cache misses and hence result in potential boom movement. However, the high-end cache subsystems do boom optimization in their own algorithms.

What we do have is optimization knowing that there is a cache. We will typically have at least two channels to each disk subsystem and may have as many as eight channels. Knowing that a request to a single disk drive may result in a cache miss, we always queue requests to the channel with the least outstanding operations. We do check to see that the request is for an area of the disk not being written by any preceding request. This allows us to take advantage of potential cache hits while one operation is delayed for a physical disk access. The payoff is very large. An I/O operation that results in a cache hit is typically serviced in about 1 millisecond. A miss may take 4 to 10 milliseconds. By allowing non-dependent I/O operations to go on in parallel, we dramatically speed up average service time.

In a high-volume transaction environment the average I/O size is 2K - 4K bytes (a database page) and about 80% reads to 20% writes. The channel of choice today is the Ultra-Fibre channel. While it is theoretically capable of 2Gb/s, the transfer rate is really not an issue given the small block sizes. Instead it is the fast set-up time of the channel. Earlier channels such as ESCON, SCSI, and Fibre often took several hundred microseconds to do their own bus negotiation and this would occur twice per request (once to send the function to the control unit and once to do the data transfer). The newest channels do this in substantially less than 100 microseconds. The result is an average service time for hits of about 700 microseconds which

includes the data transfer. With the new 15,000 RPM disks in use the overall average access time comes out to about 1.3 milliseconds if we can keep everything going in parallel. From the system perspective it's much better than that. With eight channels we average 8 operations in that 1.3 milliseconds for a throughput of over 6000 I/O operations per second. Large mainframes will have several disk subsystems and many more than the 8 channels and may achieve over 20,000 I/Os per second.

A typical PC or low-end server disk is 7200 RPM with an average access time of about 9 milliseconds. In a PC there's no point in boom optimization as you don't have requests from multiple programs queued at the same time often enough to matter. In a server it may help to reduce the average access time from about 9ms to 5-6ms. However most PCs and small servers don't have smart enough channels and drivers to take full advantage of it. For example, OS 2200 and IBM mainframes have channels with 256 sub-channels each capable of handling an I/O operation concurrently. This means all the disks in the subsystem may be positioning at the same time. Overlapped seeks, we call it. When the data transfer starts, the data blocks are multiplexed on the channel bus. Prior to the advent of all disks having at least track buffers, one disk would grab the channel for the duration of the actual transfer and any others that were ready would have to slip another revolution. Some Unix OS device drivers have this capability also.

IN A NUTSHELL:

From Ronald Smith of Unisys, "The following is a true story that shows the difference between the best theories and the "overly" complex algorithms in use in virtually all commercial operating systems.

In the early 1970s we hired a Ph.D. from the University of Minnesota whose thesis had been on CPU dispatching. We were hoping he could help us make it a lot better. In the end, he did, but only after about six months of real frustration on both sides.

When he arrived, he studied the code for a while and then told us that he could clearly see that whoever designed the dispatcher had no understanding of queuing theory. The original designer was a little miffed but had to admit it was true. The design was all empirical. When our new designer wrote up his first paper on the design approach, it started off pretty much as "...assuming an exponentially distributed arrival rate of activities at each priority level..." That didn't last through the first design

review as other designers pointed out that arrival rates were not smooth or well distributed in any really predictable sense. Instead they would show up in bursts of very different sizes and durations at different times of the day. Huge spikes often occurred first thing in the morning, just after lunch, and at the end of the day but they would be different sets of transactions at each point in many cases. The patterns were also very different at different customers. Some ran batch all day and some only overnight. Others had timesharing which showed usage patterns related to the work day but others didn't allow timesharing except for minor administrative work.

Anyway, he really was an excellent designer. After throwing out all the queuing theory analysis which he just couldn't fit to the measured facts, he studied the algorithm and came up with several heuristic improvements."

---

[1]Richard J. Petschauer, "History and Evolution of 1100/2200 Mainframe Technology" , *USE Conference Proceedings*, Fall 1990.

[2]Draft chapter by Gregory E. Mellen on noncommercial systems and Robert E. McDonald Deposition (July 16-July 18, 1974), both in the Sperry UNIVAC Archives, Hagley Library, Wilmington, Delaware; Michael J. Muolo, *A War Fighter's Guide to Space* (Maxwell Air Force Base: Air University Press, 1993); Nike-X DPS Machine-Oriented Programming Manual (1966).

[3]*Datamation*: September 1966, 19; October 1967, 135; March 1968, 131; June 1969, 149.

[4]Burroughs Indian Wells Conference, typewritten transcript, n.d. in the Burroughs Corporation Records (CBI 90), Charles Babbage Institute, University of Minnesota; *Datamation*, August 1968, 12.

[5]George Gray, "The First Time Ever I Saw an 1108", *Unisphere*, January 1995, 45.

[6] David L. Boslaugh, *When Computers Went to Sea* (Los Alamitos, CA: IEEE Computer Society Press, 1999) 358-361; AN/UYK-7(v) The Next Generation, n.d., Sperry UNIVAC Archives, Hagley Library.

[7]Gray, "The First Time Ever I Saw an 1108", 44.

[8]*Datamation*, October 15, 1970, p.78; Memo on Vendor Selection, June 19, 1968, R.E. McDonald Files, Sperry UNIVAC Archives, Hagley Library.

[9]Barry R. Borgerson, M.L. Hanson, and Philip A. Hartley, "The Evolution of the Sperry Univac 1100 Series: A History, Analysis, and Projection" *Communications of the ACM*, 21 (January 1978) 32-33 and Richard J. Petschauer, "History and Evolution of 1100/2200 Mainframe Technology" *USE Conference Proceedings*, Fall 1990.

[10]*Datamation*, January 1, 1971, 58 and January 1972, 7.

[11]Bruce Alper, "Experience with CTS" *USE Technical Papers*, Spring Conference, New Orleans, Louisiana, March 18-22, 1974.

[12]Opportunities and Threats Through 1975 and Beyond (S.W.O.T. Report), May 1, 1969, Sperry UNIVAC Archives, Hagley Library.

[13]Gray, "The First Time Ever I Saw an 1108", 44.

[14]USE inc., *Site Status Reports*, February 1990.