A CRAY RESEARCH, INC. PUBLICATION

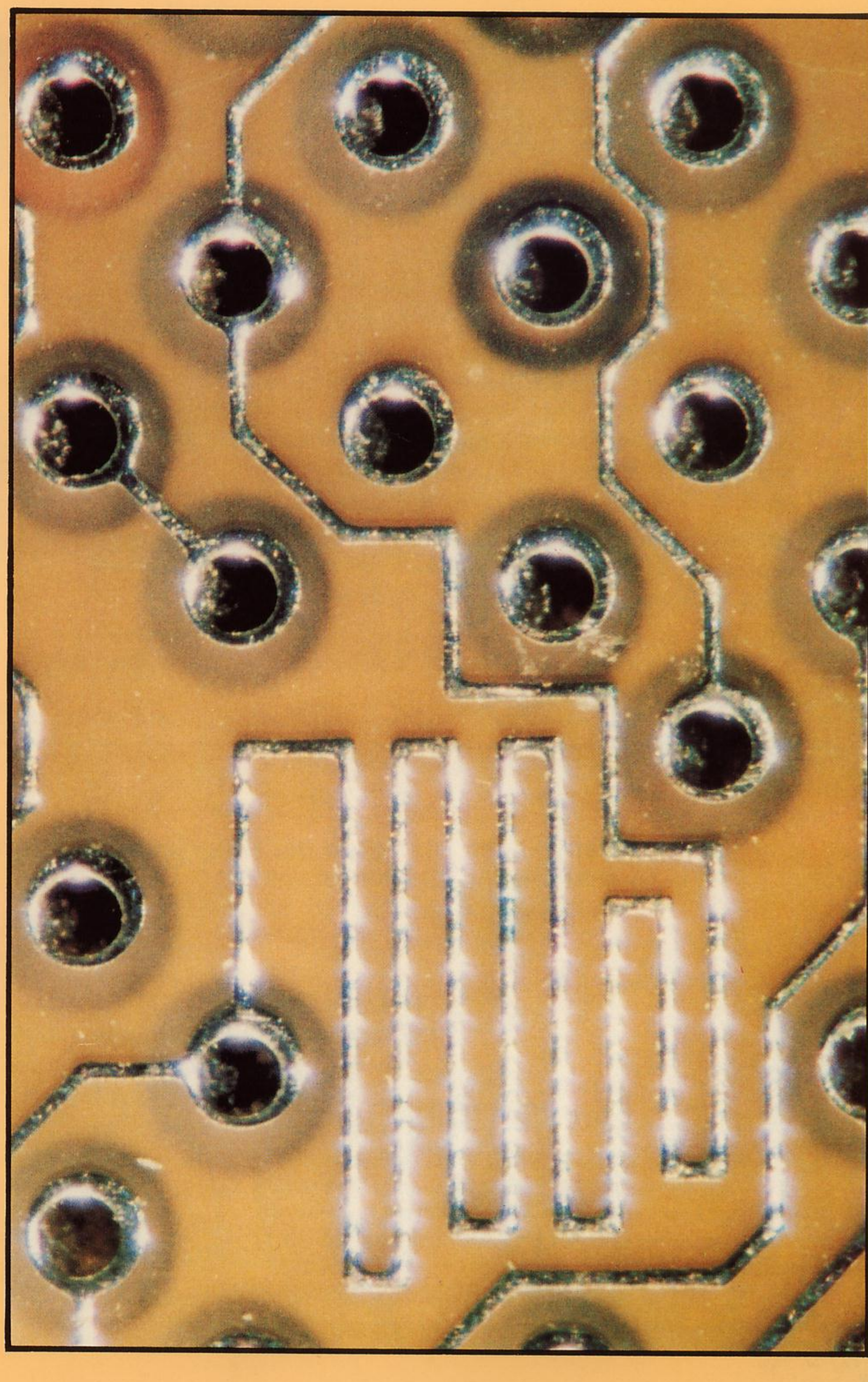
CRAY CHANNELS

Volume 3, Number 1

GNA

GNA

FEATURE ARTICLES:



AND OUR REGULAR COLUMNS:

From the editor's desk

After many months of planning, we're pleased to bring you this first issue of the new CRAY CHANNELS. Like its predecessor, CRAY CHANNELS is intended for users and potential users of Cray Research computer systems and others interested in the company and its products. The big difference is that in its new format, CRAY CHANNELS will offer users a forum for the sharing of ideas and experiences. Additionally, it will give us an opportunity to provide user information, introduce new products and discuss company activities. We also plan to feature articles giving overviews of some of the industries that currently use CRAY-1s or have the potential to become users. Naturally, we welcome your input.

In this issue, Dr. Donald A. Calahan of the University of Michigan writes about direct solution of linear equations on the CRAY-1. Dr. Calahan has special interest in vector processing and has conducted research into the influence of machine architecture on the formulation and numerical solution of large scientific and engineering problems. Each summer, Dr. Calahan chairs a course in vector processing on the Ann Arbor campus.

Also in this issue, Robert Hyatt of the University of Southern Mississippi writes about computer chess and the uphill struggle to make computers competitive with the best human chess players in the world. In Part One of this two-part series, Hyatt describes how the computer chess software keeps track of the board, searches for moves, and evaluates positions. In our next isssue of CRAY CHANNELS, we go to a computer chess tournament, and Hyatt evaluates his own computer chess program.

Several new features are being introduced in CRAY CHANNELS at this time. The "Scientific Applications Package Highlight" will feature a different major software program or package available on the CRAY-1 in each issue. In this issue, we take a look at some uses of the MacNeal-Schwendler Corporation's MSC/NASTRAN structural analysis program. Another new feature, the "Corporate Register," will keep you up to date on Cray Research, Inc. business activities and other news of interest.

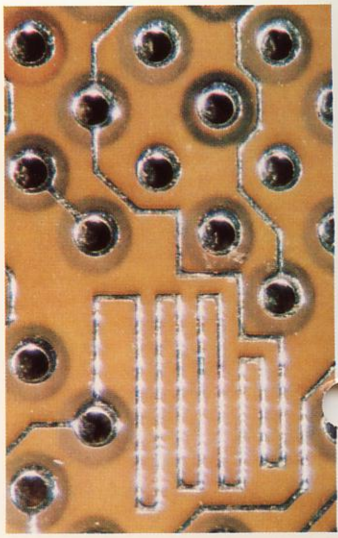
Still with us from the past are the "Software Release Summary," which describes the latest release of Cray Research, Inc. software, and "Applications In Depth," which details applications news.

All in all, we hope you'll like the new CRAY CHANNELS and that you'll look forward to future issues. We look forward to your comments!

— **T.M.B.**

About the cover

Photomicrography of a printed circuit board



A portion of a layer of a printed circuit board magnified 90 times shows the fine detail involved in the CRAY-1 computer. Copper conducting paths are laminated to a fiberglass board. Solder coating is then applied on top of the copper. Through-holes allow connection to the other side of the circuit board. Each printed circuit board in the CRAY-1 has five layers: two outer layers, one layer carrying terminator voltage, one carrying logic voltage, and a ground. (Photo credit: Jill Antolak)

CRAY CHANNELS
A Cray Research, Inc. publication Volume 3, Number 1

CONTENTS

FEATURE ARTICLES

REGULAR COLUMNS

CREDITS

Editor — Tina M. Bonetti
Art/Design — Chad Jewett

CRAY CHANNELS is a quarterly publication of Marketing Publications, Cray Research, Inc., 1440 Northland Drive, Mendota Heights, MN 55120. CRAY CHANNELS is intended for users of Cray Research computer systems and others interested in the company and its products.

Requests for copies of CRAY CHANNELS may be addressed to the Publications Department of Cray Research at the above address. If you have a feature article or news item to submit for publication, please contact the editor of CRAY CHANNELS either by mail or by telephone at (612) 452-6650.

Direct solution of linear equations on the CRAY-1

Dr. D. A. Calahan, University of Michigan, Ann Arbor

Direct (contrast iterative) methods for solving a system of linear equations (A.X = B) are the most time-consuming kernels of many scientific simulation and engineering design codes. In nearly every discipline, classes of problems exist where full- or half-implicit numerical methods yield matrices ranging from full, banded, general sparse, to tridiagonal structures.

A register-to-register machine such as the CRAY-1 is architecturally well-suited to the solution of such problems for two reasons. First, the small startup time from the vector registers permits efficient operations with short vectors arising from processing small matrices, while incurring only a modest penalty for larger matrices producing vector lengths longer than 64. Second, the relatively large number of floating-point operations usually performed on each matrix element (for example, in matrix multiplication) results in a potentially low data traffic over the critical data path from main memory.

This preference for matrix operations encourages the coding of highly-tuned CRAY-1 Assembly Language (CAL) kernels to schedule vector operations and reduce data flow, which produces a significant differential between the performance of CAL and CRAY-1 FORTRAN (CFT) codings. The performance differential, however, is far less for iterative methods.

The emphasis in this article is on the performance and applications of the CAL codes; the justification for this emphasis will also be presented by comparing selective CAL and CFT timings.

ASSEMBLY LANGUAGE SOLVERS

Dense systems

Vectors are created in linear algebra codes by exploiting either density *within* a matrix or submatrix, or patterns *across* identical submatrices. The former, being a local property, is usually more easily identified and can be utilized with less algorithm modification.

In solving large unsymmetric matrices that are dense or have locally-dense submatrices, three kernels are involved:

1. Factorization: A → LU
2. Forward and backward substitution: X → $U^{-1}L^{-1}B$
3. Multiplication and accumulation: C → C - A*B

The execution rates of CAL kernels that perform these functions are given in Table 1 [1]. These rates were prepared with the aid of a simulator [2] to achieve high short-vector performance.

Square block size	Factorization kernel	Substitution kernel	Multiplication kernel
3	6.9	5.7	17
4	8.5	12	26
6	18	21	43
8	23	30	60
12	45	46	84
16	60	58	98
32	94	89	124
64	123	117	141

Table 1. *Execution rate (MFLOPS) of three numeric kernels. All matrices are assumed square.*

Several applications of these kernels are now considered.

Full matrix factorization

The highest performance algorithm for factoring a large full matrix was proposed by Jordan [3] and consists of partitioning the matrix into at most 64-length blocks. Using the kernels of Table 1, one can obtain the execution rates of curve A in Figure 1.

For orders N≥128, the partitioned code is significantly faster than an earlier routinely-coded CAL implementation [4] (curve B in Figure 1).

Finite difference and finite element grids

The 10 to 100:1 speedup offered by the CRAY-1 over scalar processors has led researchers to consider the inclusion and coupling of more unknowns at a grid point and the coupling of more grid points by more implicit solution strategies. Such local coupling produces a naturally-blocked matrix — for example, block tridiagonal or blocked general sparse — with equally-sized blocks. If only this local blocking is exploited, Table 1 gives insight into expected execution rates. For example, for blocks of order 4, rates range from 8.5 to 26 million floating-point operations per second (MFLOPS) for the three kernels; in practice, one can solve two-dimensional grids with blocked general-sparse software [1] at 16 MFLOPS, when all blocks are 4x4. More global blocking can only increase this rate.

Band solvers

Several CAL band solvers have been proposed by Jordan [3]; achieving nearly 120 MFLOPS, these are probably near-optimal for the classes of matrices they solve. Unfortunately, they will not accommodate half-bandwidths greater than 64. A block-oriented extension has been described in [1], but this approach is inefficient for small bandwidths.

Because of the large size of typical band matrices, it is common to maintain only a part of the matrix in main memory during solution. The associated I/O scheduling tends to make such solvers dedicated to application codes and to operating system environments.

Pivoting

The pivoting process involves a sequence of vector mask operations to locate the maximum element of a vector [3].

Unfortunately, the impact of pivoting on timing is not well-documented in the literature because CAL codes being compared involve different codings of the factorization process itself. Nonetheless, Jordan's comparison [7] of different pivoting and non-pivoting CAL codes (Figure 1, curves B and C) showing a 15 to 25% slow-down due to pivoting is consistent with informal studies of the algorithms involved.

Identically-structured systems

An examination of large equation sets devoid of natural block substructure inevitably discloses another opportunity for vectorization: namely, the problem is large due to (M) replicated identical substructures. (Indeed, one may wonder how a computationally large matrix problem arises if not principally from blocking or replication.) All substructures are then *simultaneously* reduced by vector operations of length M.

Representative applications and results follow.

1. Alternating direction iteration (ADI) methods generate simultaneous systems of equations; with one unknown per grid point, 40 MFLOPS can be achieved with 64-length vectors [3]; systems of block tridiagonal equations may be solved at 70 MFLOPS with blocks as small as 3x3 [8].

2. Large sparse matrices arising from circuits and rigid body dynamics tend to have identical sparse substructures originating from the hierarchial nature of engineering design; *simultaneous* general sparse solvers have been found to execute in the range of 25-50 MFLOPS [9].

Existing scalar application programs must usually be altered to permit simultaneous formulation of the submatrix equations in vector mode. Also, simultaneous solution presumes that all matrices are available in main memory, requiring more storage than when systems can be formulated and solved one at a time. Thus, the price of vectorization may be high, although warranted for important production scientific and computer-aided design codes.

Highly-sparse systems

In the absence of both local block structure and global patterns, highly-sparse matrices, such as tridiagonal or general sparse, may be best solved by careful scheduling of scalar instructions with a simulator. For example, tridiagonal equations may be solved in scalar mode in the range of 18 MFLOPS [6], outperforming vectorized cyclic reduction CAL codes in many circumstances [8]. Similarly, the early stages of elimination of general sparse matrices can be performed in the range of 10-15 MFLOPS by scheduling and emitting appropriate scalar instructions that are executed at each numerical solution (so-called "code generation"). A polyalgorithm to reduce the remaining denser portions of the matrix is currently being studied.

Characteristically, the reduction of highly-sparse matrices does not dominate the equation formulation, which often must be performed in scalar mode utilizing the FORTRAN compiler. Thus, the overall speedup offered by CAL coding may be marginal.

FORTRAN SOLVERS

The above collections of often specialized CAL codes have arisen from interest in demonstrating ultimate CRAY-1 performance or else to service important application codes. These have been assembled into a library by Jordan [10].

As the CRAY-1 has been accepted as a high-performance general-purpose scientific processor, it has increasingly attracted the discipline-oriented user less willing to depend on or accommodate to assembly-coded kernels, often with poorly-documented algorithms. This user much prefers a (modified, if necessary) FORTRAN code that he or she can understand.

At this writing, two well-known FORTRAN linear algebra libraries — the IMSL and NAG collections — are being converted to the CRAY-1 with proper attention to the unique requirements of the CFT compiler and the CRAY-1 architecture. An earlier conversion of selective LINPACK codes was performed by Dongarra [5] and the entire library was converted by Mike Ess at Cray Research.

Two comparisons of FORTRAN and CAL solvers are presented to represent the cost of operating from a higher-level language.

First, an early version of the CFT compiler produced the (pivoting) LU factorization results of curve D in Figure 1 [7]. The equivalent CAL code (curve C) yields a 5:1 speedup for most matrix orders. Secondly, a more recent comparison of Ess's FORTRAN LINPACK (non-pivoting) full matrix solution shows a 5.7:1 speedup in factorization and a 3:1 speedup in forward and back substitution for the high-performance CAL code of curve A, for matrices of order 64 [11].

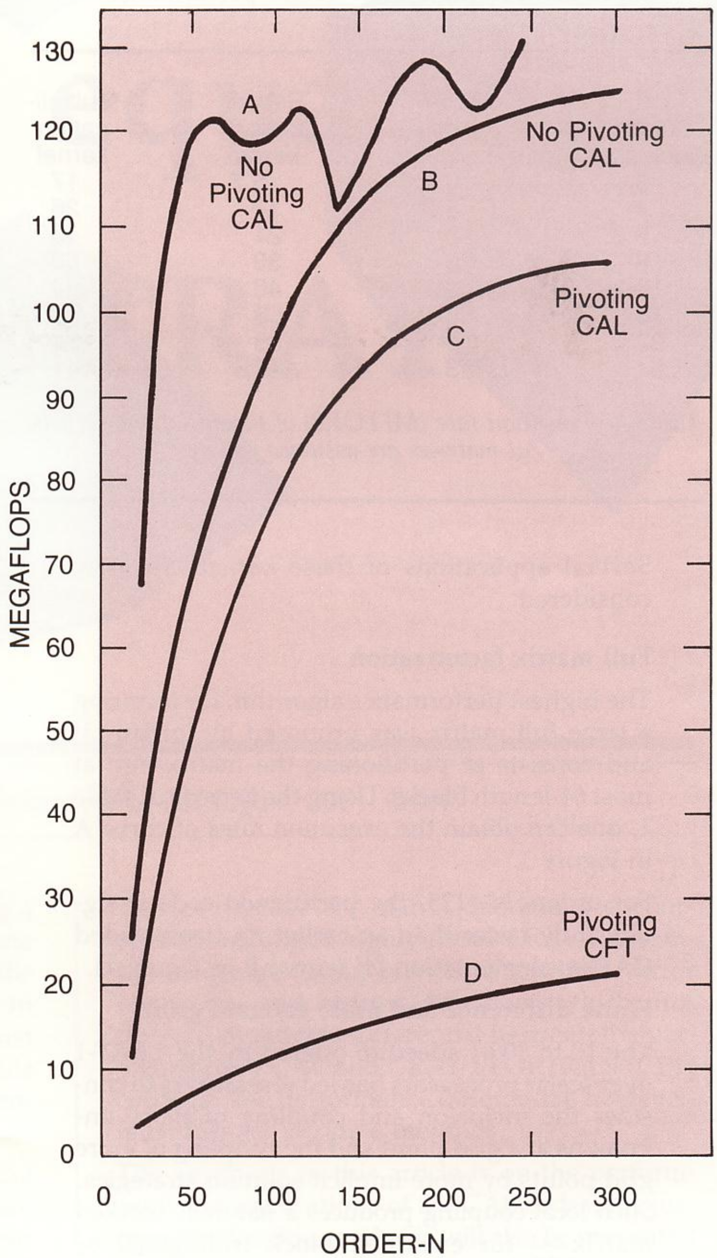


Figure 1. Execution rates of LU factorization of full matrices

Such speedups are difficult to ignore if the equation solution time is a dominant part of the total simulation time, and when general codes are accessed frequently or special codes are run in a production mode. □

ACKNOWLEDGEMENT

The University of Michigan studies reported here were supported by the Computational Mathematics program at the Air Force Office of Scientific Research.

References

[1] **Calahan, D. A.,** *"A Block Oriented Equation Solver for the CRAY-1,"* Report #136, Systems Engineering Laboratory, University of Michigan, December, 1980.

[2] **Orbits, D. A.,** *"A CRAY-1 Simulator,"* Report #118, Systems Engineering Laboratory, University of Michigan, September, 1978.

[3] **Jordan, T. and R. Fong,** *"Some Linear Algebra Codes and Their Performance on the CRAY-1,"* Report LA-6774, Los Alamos Scientific Laboratory, June, 1977.

[4] **Calahan, D. A., W. N. Joy, and D. A. Orbits,** *"Preliminary Report on Results of Matrix Benchmarks on Vector Processors,"* Report #94, Systems Engineering Laboratory, University of Michigan, May, 1976.

[5] **Dongarra, J. J.,** *"LINPACK Working Note #11: LINPACK Timings on the CRAY-1,"* Report LA-7389-NS, Los Alamos Scientific Laboratory, August, 1978.

[6] **Brown, Forrest,** *"A High Performance Scalar Tridiagonal Solver for the CRAY-1,"* Dept. of Nuclear Engineering, University of Michigan, April, 1980.

[7] **Jordan, T.,** *"A Performance Evaluation of Linear Algebra Software in Parallel Architectures,"* Report LA-8078-MS, Los Alamos Scientific Laboratory, October, 1979.

[8] **Calahan, D. A., et al,** *"A Collection of Equation-Solving Codes for the CRAY-1,"* Report #133, Systems Engineering Laboratory, University of Michigan, August, 1979.

[9] **Calahan, D. A.,** *"Multilevel Vectorized Sparse Solution of LSI Circuits,"* IEEE International Conference on Circuits and Computers, October, 1980, pp. 976-979.

[10] **Jordan, T.,** *"Directory of CAL Routines for Use with CFT on the CRAY-1,"* October, 1980 (private communication).

[11] **Calahan, D. A.,** *"Performance of Linear Algebra Codes on the CRAY-1,"* (to appear, SPE Journal).

—ABOUT the AUTHOR—

Donald A. Calahan *received his B.S. degree from Notre Dame University, Notre Dame, Indiana in 1957, and his M.S. and Ph.D. degrees from the University of Illinois, Urbana, in 1958 and 1960, respectively, all in electrical engineering. He has been on the faculty at the University of Michigan, Ann Arbor, since 1966. He has authored two books and coauthored a third on circuit analysis and computer-aided design. His current research interests are in the influence of machine architecture on the formulation and numerical solution of large scientific and engineering problems, such as particle physics, aerodynamic fluid flow, oil reservoir analysis and circuit design. He was elected to fellow of the IEEE in 1973.*

CHECKMATE — TI

For the past year, Cray Research has provided computer time for computer chess program development and tournament action. The rigors of chess provide a strenuous trial for both hardware and software, giving the company an additional testing ground. The fact that CRAY BLITZ happens to be one of the top-ranked computer chess programs in the world is just frosting on the cake.

he CRAY-1 plays chess

Robert Hyatt
University of Southern Mississippi, Hattiesburg

Part one of a two-part series

"Can computers think?" This has been one of the most controversial questions provoked by the modern digital computer. It has been proven that for finite mathematics where exact solutions exist, the computer is vastly superior to man in finding the solution. However, man has always had a clear superiority in inexact problems, due to such attributes as intuition, hunches and other non-quantifiable decision-making processes. The computer is rapidly eliminating these advantages as hardware speeds improve and programming tools are redesigned.

Since chess has long been considered an "intellectual" exercise, it was natural that the computer would be applied to its solution. While there is still a significant difference in playing strength between world-champion class players and the computer, this difference is shrinking yearly. In fact, most computer scientists believe that it is only a matter of time before computers become unbeatable. As evidence of this, six years ago chess masters easily defeated computers in speed chess, simultaneous exhibitions and regular tournament chess. In a period of six years, the top programs have sharply reduced this gap. Top human players struggle to win speed chess games and simultaneous exhibition games and even world-champion class players have fallen victim to computers in speed chess. Chess masters are beginning to find that they lose regularly to the better programs, even in tournament-level chess matches. If chess programs continue to improve over the next decade as they have over the past decade, the programs could easily invade the ranks of the top 50 players in the world and have a significant chance at becoming world champions.

Most computer scientists believe that it is only a matter of time before computers become unbeatable.

INTERNAL BOARD REPRESENTATION

The logical chess board is an 8 by 8 array representing the 64 board squares. To allow rapid detection of the edge of the board during move generations, a double row of border squares is used (Figure 1). Squares outlined by the dark lines are on the board and squares outside of the dark lines are illegal squares.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120

***Figure 1.** Logical chess board*

Pieces are represented as digits 1 through 6, with pawn = 1, knight = 2, bishop = 3, rook = 4, queen = 5, and king = 6. An empty square has a value of 0, and a square off of the board has a value of 99. To distinguish program pieces from the opponent's pieces, program pieces are given positive numbers and the opponent's pieces are negative numbers.

Move generation with this representation is simple. For example, let's generate the moves of a program rook on square 22. The four legal directions for a rook are +1, -1, +10, and -10. The rook can move in the +1 direction to squares 23, 24, 25, 26, 27, 28, or 29 unless one of them is occupied. If a square is occupied (a number other than 0), the rook cannot move beyond it. Also, if the sign of the occupying piece is +, the rook cannot capture it. These steps are repeated for the other three directions to enumerate all legal rook moves. Note that for the -1 and -10 directions, a 99 will immediately terminate generation of that direction, indicating that the edge of the board has been passed.

Because, during the course of a game, each side has an average of 38 legal moves, a tree of depth 6 has 38^6 or 3,010,936,384 terminal positions!

GAME TREE SEARCH

The game tree search is the tactical component of the move selection process. The tree search will find (within reason) move sequences that will avoid losing material whenever material loss is threatened, and it will find move sequences that will win material whenever the opportunity arises.

The basic tree search is sometimes called a minimax full-width depth-first game tree search. The program is given a position for which it must try to find the best move. This position is referred to as the "root" or "base" of the tree. From this position, the program generates all of the possible moves and stores them in an array. The first move is selected and made on the internal game board. Now, the program switches sides and generates all possible moves from this new position. The first move is again selected and made, and this process continues until some fixed depth is reached.

At this point, the evaluation function is used to compute a numeric value representing the current position. The larger the value, the better for the program; the smaller the value, the better for the opponent. The side to move stores this score and selects the next move. After all moves at this level have been examined, the side to move will have chosen the score that was highest or lowest, depending on which side is to move. This score is returned to the previous level, where it is stored, and a new move is selected. Once again the program follows the process of advancing to the next level, generating all moves and making one, until the maximum depth is reached. As all moves for a level are examined, the best (worst) score is returned to the previous level. This continues until all moves at the base level are examined.

As can be seen, the game tree is quite large. Because, during the course of a game, each side has an average of 38 legal moves, a tree of depth 6 has 38^6 or 3,010,936,384 terminal positions! Even if the program is only using one microsecond of processor time per node, this is still nearly one hour of computation per move.

To speed up the process, then, there is a technique known as the alpha-beta backward pruning algorithm that drastically reduces the number of positions evaluated without affecting the ultimate outcome of the search. Suppose that the first move the program examines is N-B7 check (Figure 2). The tree search concludes that this wins a rook and returns a value of +5. When the program selects the next move to analyze, the alpha-beta algorithm may help as follows: after changing sides, generating all moves and selecting one, and doing the additional searching necessary to evaluate the resulting position, the program discovers that after it plays P-KR3, the opponent plays P-QR3, and it does not win a rook. The program could continue analyzing moves at level two and might even find that P-KR3, B-B4 wins a pawn. However, this is wasted time for two reasons: 1) N-B7+ wins a rook, and 2) P-KR3 wins nothing and may even lose something. Therefore, the program ignores P-KR3 and selects another move.

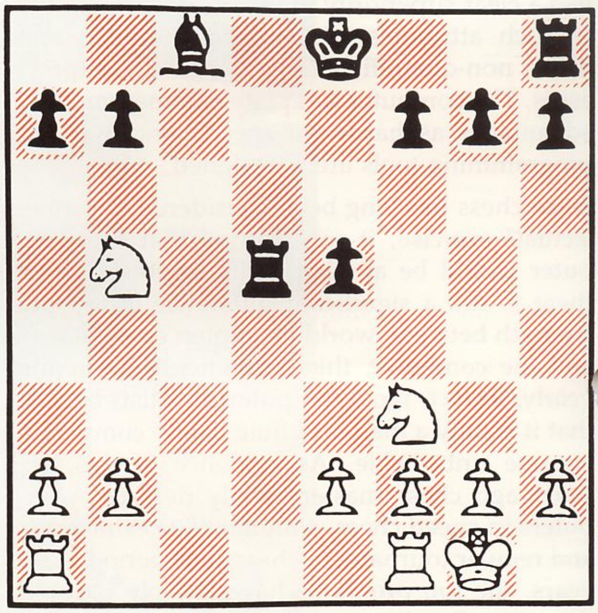


Figure 2. White to move

The above strategy is analogous to the following example: suppose you have on the table ten bowls of stew, and you must select one to eat. Some are far too hot, some are too cold, some are spoiled, and some are improperly cooked. Suppose the first bowl you select is slightly cool, but otherwise is acceptable. Examination of the next bowl finds it boiling hot. Testing it to determine if it tastes right is futile, since it can be rejected based on being hotter than something already found.

Using this algorithm can potentially reduce the number of nodes from 38^6 to $2x38^3$ or 109,744. In practice, this is difficult to achieve, but dramatic savings are possible nonetheless. On the CRAY-1, for example, a typical six ply search might examine 200,000 nodes, which is far less than the potential maximum.

There are many other useful algorithms that can help speed up the search. First, the alpha-beta algorithm depends on move ordering being quite accurate. For example, in the situation mentioned above, if the program had examined P-KR3 first, then it would also have to completely examine N-B7+ to evaluate the move, because alpha-beta would not terminate analysis early.

In Figure 3, the program must defend against the move N-B7+. However, since the tree search is responsible for tactics, the program will simply generate all moves at level one and make the first one. After changing sides and analyzing all of the replies, the program determines that the first move, P-K3, is answered by N-B7+ where the opponent wins a rook. This move, N-B7+ is remembered as a "killer" move for level two. Now, after trying P-K4, the killer move is tried first. This gives good moves a chance to be examined first, causing more alpha-beta cutoffs. In fact, until R-QB1 is examined, all of the program's moves are refuted by N-B7+.

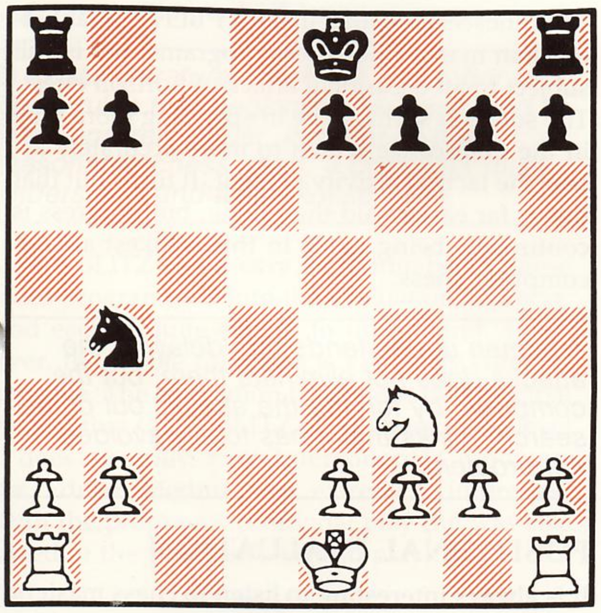


Figure 3. White to move

CRAY BLITZ maintains (remembers) up to ten killers per level in the tree. The list is carefully ordered so that the more useful killers are examined first.

Before continuing, it is worth mentioning that the killer heuristic, as well as others that improve the tree search, does not alter the move that the program ultimately will make. However, it does alter drastically the amount of time it will take to perform the search. In fact, removing the killer logic completely will increase the search time by a factor of at least ten. This is the principal advantage of the full-width or exhaustive search — that move-ordering can affect the speed of the search, but not the outcome.

Another time-saving algorithm used in CRAY BLITZ is the transposition table. When doing a six ply tree search, many different pathways may converge to the same position. For example, the move sequences N-KB3, B-QB4, N-KN5 and N-KR3, B-QB4, N-KN5 both converge to the same position. After reaching this position from one branch of the tree, and then doing an additional three ply search from this point, the result of this search is saved in the transposition table. After playing N-KR3, B-QB4, N-KN5, this position is found in the table along with the search value. This avoids analyzing the subtree that would normally be developed. This algorithm results in a speed improvement of 2 to 5 in the middle game and a speed improvement of over 100 in simple endgames.

The killer heuristic does not alter the move that the program ultimately will make. However, it does alter drastically the amount of time it will take to perform the search.

QUIESCENCE

Quiescence is the single most important consideration in a computer chess program. The problem centers around when to apply the positional evaluation, or more technically, raises the question of when a position is tactically stable enough to be accurately evaluated. For example, when performing a three ply search, the move sequence R-K1, R-K1, RXR seems to win a rook; however, suppose the next move (if the program were searching to four ply) is NXQ. If the positional evaluator were applied after these moves, it would conclude that the program is winning a rook (remember that tactics are handled by the search only).

Since terminating the search at some arbitrary fixed depth results in many blunders, most chess programs carry out the tree search in three distinct phases. Phase One is the full-width or exhaustive part of the search, and is carried out to some predetermined depth based on timing requirements. Phase Two is the quiescence part of the search and is carried out until the quiescence search encounters a position that is tactically stable. Phase Three is the positional evaluator, which can now be applied because no pieces are hanging or no threats exist.

The quiescence search differs from the full-width search in that only a small subset of the legal moves are examined — notably, captures to stabilize the material balance and checks to find/avoid checkmates. Unfortunately, this area is key to why a computer is not the current world champion. It is extremely difficult for a chess program to determine if a position is tactically quiescent.

For example, if the position in Figure 4 were encountered during the quiescence search, it might be judged as a terminal position, since no material is hanging. However, white can attack the black knight, which is pinned, more times that black can defend it, winning at least a piece.

While a human easily sees this and avoids it, a computer has a great deal of difficulty with it. CRAY BLITZ examines winning captures and checks in the quiescence search, as do most other top-flight programs, and all would improperly evaluate this position if it occurs in the quiescence search.

Another problem of quiescence is known as the "horizon effect". Simply, if a program can delay something so that it is not discerned within the search, it does not exist! In Figure 4, if black continually attacks white pieces with its pawns and pieces, white doesn't have time to continue attacking the knight to win it. A human understands that delaying the attacks does not eliminate them, but the computer, by forcing the attacks out of the search, thinks that it has totally avoided the problem.

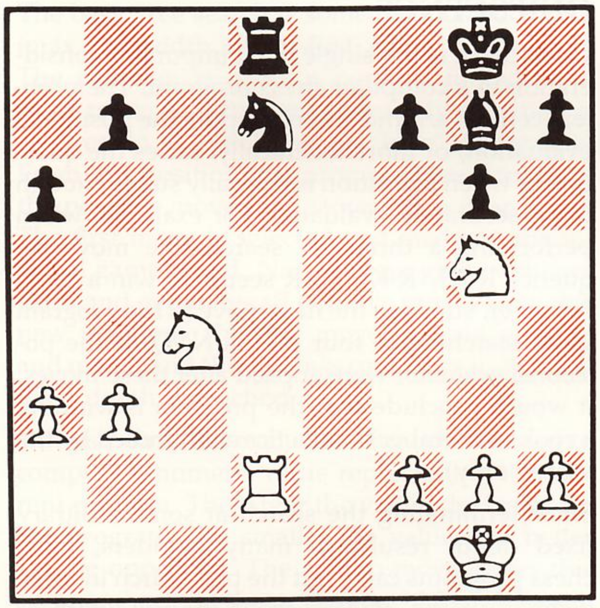


Figure 4. *White to move*

An additional item of interest concerns the program's desire to play RXQ, winning the queen (Figure 5). Assume that the program plays RXQ while performing a one ply search. The quiescence search tries R-K8 and finds that white is checkmated. The program then tries NXRP, PXN, RXQ. This position is reached in the quiescence search at a point where checks are not considered (considering all checks could result in an infinite loop). This is because there are stringent controls on what is included in the quiescence search in order to conserve time. The program happily evaluates this as good for white. After playing this, and receiving black's response, the program is surprised to find that once more it cannot play RXQ.

In computer chess tournaments, there have been many amusing examples of this problem. Suppose that black is faced with the loss of a trapped rook. There have been actual games played where black would force white to capture a series of pawns and pieces merely to delay (eliminate) the loss of the rook.

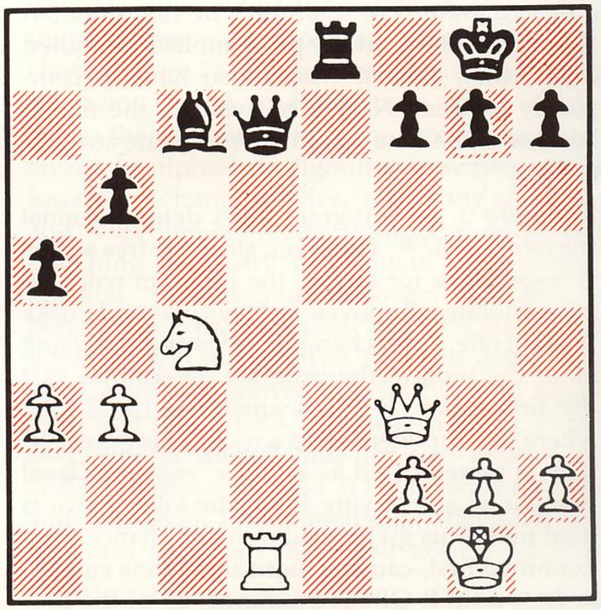


Figure 5. *White to move*

The quiescence search in CRAY BLITZ is far better than many other chess programs, but is still subject to an occasional attack of horizon effect. The solution seems to be in spending more time in the quiescence search to more accurately assess the tactical activity present. It turns out that this is far easier said than done, but progress is continually being made in this weakest area of computer chess.

A human understands that delaying the attacks does not eliminate them, but the computer, by forcing the attacks out of the search, thinks that it has totally avoided the problem.

POSITIONAL EVALUATION

It is always interesting to listen to chess masters comment on computer chess programs. The common assessment is that they are tactically brilliant but positionally weak. However, positional chess is really just long-range tactical chess. If the program can win material, the tree search is responsible for finding out how; or, if the program is threatened with losing material, the tree search is responsible for finding a defense. Positional chess is simply determining what to do when there is nothing to do.

In chess literature, there are a number of basic rules that beginners are cautioned to learn and remember, such as "control and occupy the center of the board", "occupy open files with rooks", and "keep the king safely tucked away behind pawns." In order to play reasonable chess, then, a computer must understand and follow these rules.

The dominating term in the evaluation function is material. That is, the program will almost always prefer capturing a tangible piece to win material rather than winning some positional advantage while giving up a material advantage. As can be seen, the positional judgement of the program is only used when material balance cannot be altered.

The positional evaluators are really quite simple, although they are relatively long. For example, the chess principle "a knight on the rim is dim" implies knights should avoid squares on the edge of the board. If you imagine the chess board as an 8 x 8 array, the necessary loops to check each square on the outer edge of the array are quite simple. Each square must be scanned looking for a +/- 2 (white/black knight). For each one found, add a penalty for the correct side to the positional score (assuming + scores are good for white, a white knight on the edge might get -100 added to the positional score). If the tree search encounters this identical position, except that the knight has moved off of the edge, the positional score would be 100 points better, causing this position to be favored.

In chess literature, there are a number of basic rules that beginners are cautioned to learn and remember. In order to play reasonable chess, then, a computer must understand and follow these rules.

CRAY BLITZ has a very large number of these rules programmed into the evaluation function, and each is quite simple to understand. However, the whole collection is extremely comprehensive. The evaluation function is the principal reason that CRAY BLITZ plays chess at the level it does currently. Even International Grandmaster Edmar Mednis was extremely impressed with the program's positional play (he was also awed at the program's tactical play). □

—ABOUT the AUTHOR—

Robert Hyatt *is an Instructor and Chief of Systems at the University of Southern Mississippi in Hattiesburg. He received his B.S. in Computer Science from USM in 1970 and has remained there to teach and do research. Bob has been competing in computer chess tournaments with BLITZ since 1976. He has had CRAY-1 support from Cray Research since April of 1980. Recently, Bob completed work on a microprocessor-based electronic chess board which he uses in tournament play.*

In the next issue of CRAY CHANNELS, Bob describes a computer chess tournament and evaluates CRAY BLITZ.

Scientific applications package highlight:

MSC/NASTRAN

MSC/NASTRAN is a large-scale general-purpose digital computer program used for solving a variety of engineering analysis problems by the finite element method. Developed and maintained by the MacNeal-Schwendler Corporation, MSC/NASTRAN is an enhanced version of the original *NA*SA *STR*uctural *AN*alysis program. The capabilities of MSC/NASTRAN include static and dynamic structural analysis, material and geometric nonlinearity, heat transfer, aeroelasticity, acoustics and electromagnetism analysis. MSC/NASTRAN has been successfully used worldwide by large and small companies engaged in the automotive, aerospace, civil engineering, shipbuilding, offshore oil, industrial equipment and chemical engineering industries and in government research.

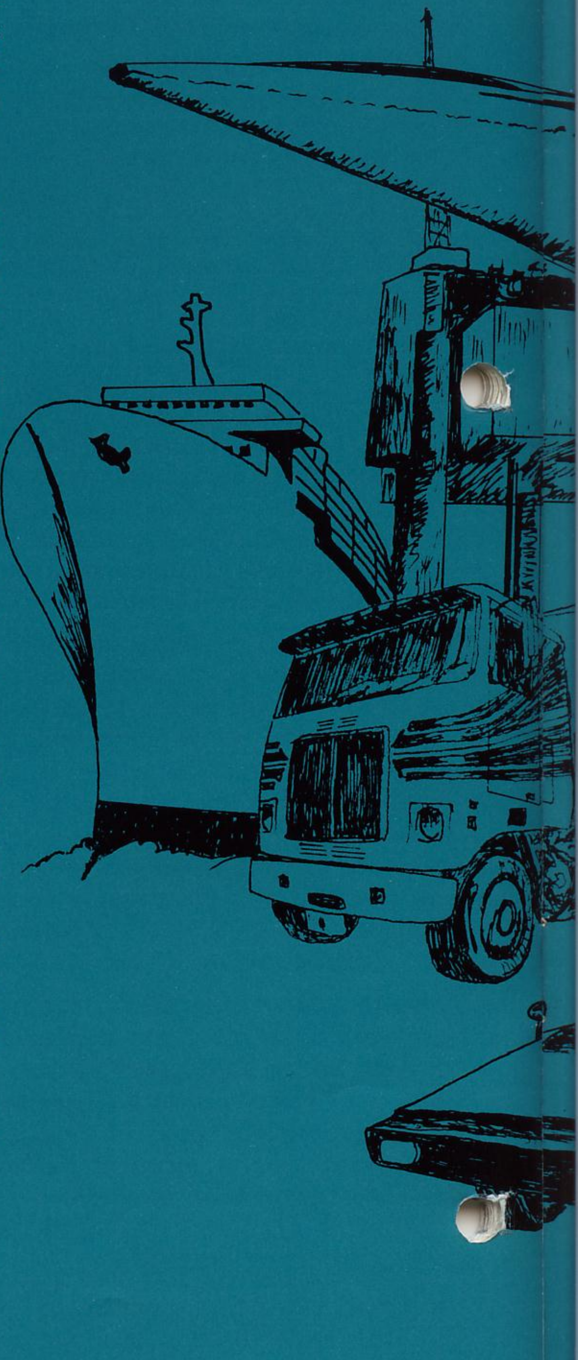
Initially, NASTRAN was developed for use as a problem solving tool for the aerospace industry. One interesting application of MSC/NASTRAN was its use in modal analysis of the *S*pace *T*ransportation *S*ystem (STS) and its payload for both lift-off and landing configurations. Modal analyses of launch vehicle/payload coupled systems have proven to be useful in providing structural data during the design and development of a spacecraft. A substructure technique can be used to combine two initially incompatible substructure models, and the resultant combination can then be analyzed. Thus, the fact that the launch vehicle and payload may have been developed by different contractors on different computer systems no longer need be an issue. The results can provide spacecraft contractors with timely data for assessing launch vehicle/payload dynamic interaction and for performing transient analyses to determine member loads.

MSC/NASTRAN is the accepted standard for structural analysis in the transportation industry. MSC/NASTRAN has been found useful in the analysis of trucks, tractors, trains and ships. It also enables American, Japanese and European automakers to analyze various candidate designs. For example, the need for automotive downsizing has offered finite element analysis techniques new opportunities to make a significant impact on the development and manufacture of highly efficient engine-transmission designs. Efficient use of material in this type of design calls for a thorough understanding of the dynamic behavior of the system.

In energy-related industries, MSC/NASTRAN has proven to be a useful tool for the analysis of structures such as containment vessels for nuclear and conventional power plants. Offshore drilling rigs, piping and valves can also be analyzed thoroughly for strength and the ability to withstand severe temperature extremes.

Architectural and structural engineering firms represent areas where the use of MSC/NASTRAN is rapidly growing. Its accuracy and efficiency in analyzing a wide variety of structures makes MSC/NASTRAN a highly useful tool. In fact, through MSC/NASTRAN's sophisticated analytical tools, engineers can achieve dramatic reductions in structural weight and costs of design, construction, maintenance and operation.

The MSC/NASTRAN heat transfer option has been used to demonstrate finite element analysis of biological heat transfer. The heat transfer option has allowed researchers to determine temperature gradients and heat fluxes for a general three-dimensional inhomogeneous biological medium, the human arm. The complex geometric and material properties of biological media have made thermal calculations very difficult. Such calculation is further complicated when it is attempted on living organisms. Such research provides valuable medical understanding of the exceptionally high fever found in heat stroke. Using MSC/NASTRAN, the various tissue types in the arm can be modeled using elements that conform to their geometries. Temperature distribution in the arm can then be calculated and studied.



MSC/NASTRAN embodies a finite element approach wherein the distributed physical properties of the problem are represented by a model consisting of a finite number of idealized elements interconnected at a finite number of grid points. DMAP (*D*irect *M*atrix *A*bstraction *P*rogram), a user-oriented macroinstruction language, provides maximum flexibility and allows the analyst to solve problems by authoring tailor-made analysis routines. Preformatted solution sequences are available for the most commonly required solutions.

MSC/NASTRAN has experienced wide acceptance due to its efficient performance, generality of applications and strong customer support. It has been described as being user-friendly, versatile, and unequaled for solving a variety of design problems.

A version of the program has been developed for use specifically on the CRAY-1. Thus, MSC/NASTRAN may be applied to problems with significantly increased degrees of freedom by employing the fast computation section of the CRAY-1 and up to 4 million 64-bit words of high-speed memory.

A number of current CRAY-1 sites use MSC/NASTRAN. Additionally, the program is offered on the CRAY-1 through several service bureau operations, including United Computing Systems, Inc., who assisted in the conversion process. For more information, contact either your nearest Cray Research sales office or the MacNeal-Schwendler Corporation, 7442 N. Figueroa St., Los Angeles, CA 90041, telephone (213) 254-3456 and TWX (910) 321-2492.

APPLICATIONS IN DEPTH

New release of SAPH

A new release and update of the Cray Research Scientific Application Package Handbook (SAPH) is now available. The latest SAPH is completely revised, enlarged and updated. It appears in a new, looseleaf format facilitating regular updates and copying of individual pages.

The primary goal of the Handbook is to provide an information source on the nature, current status, and availability of the most significant scientific applications programs and packages in use today. Where CRAY-1 versions of such software exist, the Handbook indicates from what source, if any, they are available. The SAPH also offers information on users of particular software packages.

Currently, the SAPH contains over 250 entries, with one page devoted to each program or package. The software is divided into application areas such as structural analysis, nuclear engineering and safety, chemistry, math/statistical libraries, graphics, languages, and so on.

The Handbook is supported by the Applications Development and Technical Advisory Department of Cray Research. Further information can be obtained from: Dr. David Darling, Cray Research, Inc., 1440 Northland Drive, Mendota Heights, MN 55120.

Cray Research to attend finite element congress

This fall, Cray Research will be represented at the Third World Congress and Exhibition on Finite Element Methods. In its past two meetings, the World Congress and Exhibition has served as both forum and marketplace for those involved in all facets of finite element technology. The World Congress, which is held every three years, was the first major conference in finite element technology to incorporate an exhibition. At this year's exhibition, Cray Research will have a booth featuring information on the finite element packages currently running on the CRAY-1. The Third World Congress and Exhibition will be held October 12-16, 1981 at the Beverly Hilton in Beverly Hills, California.

Cray Research Applications Program Library underway

A considerable amount of publicly available applications software has been implemented on the CRAY-1. As the Cray Research marketplace has expanded, the need for a centralized source of quality public-domain applications software has grown. Thus, Cray Research has organized an Applications Program Library for CRAY-1 users. The company plans to provide a responsive service for the verification, documentation and distribution of public-domain CRAY-1 software.

The Library will contain a variety of applications software. Some of the first software to be offered through the new service will be that currently available through the Cray User Group software exchange. In addition, significant codes in specific fields, such as computational aerodynamics, structural analysis and nuclear systems analysis, will be incorporated into the Library. Availability of certain software may be restricted according to the licensing regulations of the original code developer or distributor. Where such restrictions are in effect, they will be noted in a document describing the software contained in the Library.

The scope and contents of Cray Research's new Program Library service will grow with time. An important function of the Library will be the coordination and benchmark demonstration of third-party vendor CRAY-1 software. As an example, Cray Research currently has the ability to demonstrate both MSC/NASTRAN and ANSYS, two major finite element programs, on the CRAY-1 in Mendota Heights, Minnesota.

All software within the Cray Research Program Library will be verified with test cases and fully documented prior to being offered for distribution. Although the company will not accept responsibility for the programs or for the consequences of their use, every effort will be made to ensure quality and reliability prior to distribution.

CRAY BLITZ lives up to its name

As detailed elsewhere in this issue, Robert Hyatt of the University of Southern Mississippi has developed CRAY BLITZ, a computer chess program for competitive chess play. Last fall, CRAY BLITZ participated in the Eleventh ACM North American Computer Chess Championship held in Nashville, Tennessee. Two distinct competitions were held: Speed Chess and Tournament Chess. Ten programs in all participated, including Bell Laboratories' BELLE, which is the current world champion, and the University of Michigan's CHAOS, which was runner-up to BELLE in the world competition.

Hyatt and CRAY BLITZ experienced spectacular success in the Speed Chess competition. BLITZ took first place by defeating each of the other nine participants comprehensively. In addition, a speed game played against ex-world champion CHESS 4.9 running on a CDC CYBER 176 was won decisively by BLITZ. Two international Chess Masters were dispatched in a similar manner, along with several other Expert-rated human players. All in all, BLITZ handily defeated all other players, both human and machine.

In the Tournament Chess competition, BLITZ won the first round against the other program competing on a CRAY-1, CUBE 2.0, entered by United Computing Systems of Kansas City, Missouri. In the second round, BLITZ wrestled with BELLE for 4½ hours in what was described by tournament organizer David Levy as "the best game ever played by two computers." CRAY BLITZ fought a spectacular battle before succumbing. The third game saw BLITZ defeat BEBE, a special chess hardware system currently ranked fifth in the world. In the final game, BLITZ lost to second-ranked CHAOS. Thus, in Tournament Chess action, CRAY BLITZ finished with a record of two wins and two losses, good for a fourth-place finish.

Since the championships, Hyatt has continued work on CRAY BLITZ. He has concentrated on performance improvements, including tuning the evaluation function to deep ply searches in tournament mode and rewriting often-used routines for enhanced speed. His work continues with an eye towards the Mississippi state tournament to be held over Labor Day and, beyond that, the next ACM championship to be held in November in Los Angeles. In his own words, "CRAY BLITZ looks like it will be hard to beat next time."

NISA now available on the CRAY-1

NISA, *N*umerically *I*ntegrated Elements for *S*ystems *A*nalysis, is now available for use on the CRAY-1 computer. Developed by Engineering Mechanics Research Corporation, NISA provides linear analysis of static, dynamic, steady-state and transient heat transfer and field problems. NISA uses a large library of high-order isoparametric elements. Other features of NISA include free-form data input, a preprocessor for finite element model generation and a variety of output options. For more information about NISA, contact Engineering Mechanics Research Corporation, P.O. Box 696, Troy, MI 48099, telephone (313) 968-1606.

CRI to participate in vector processing seminar

Dr. D. A. Calahan, who wrote the article on linear equation solution appearing in this issue, chairs a course on vector processing each summer. Entitled "High Speed Computation: Vector Processing," the course is held on the University of Michigan campus in Ann Arbor. This year, the course will run August 10-14 and will include lectures and a laboratory segment.

Following are some of the lectures and speakers scheduled for the week:

Vector Algorithm Organization and Evaluation — D. A. Calahan, University of Michigan

Organization and Programming of the CRAY-1 — Derek Robb, Cray Research, Inc.

Applications and Performance of the CRAY-1 — Derek Robb, D. A. Calahan

Performance of Finite Element Programs on the CRAY-1/S — Kash Kasturi, United Computing Systems, Inc.

Other lecture topics will include user experience in multiprocessing, the impact of vector languages, state-of-the-art multiprocessing hardware, and overviews and performance information on several other vector computers.

Remote access to the CRAY-1 will be provided during the laboratory segment of the course. Participants may bring a small (300-statement) FORTRAN program to run. Program counseling will be available.

For more information about the conference, write: Engineering Summer Conferences, Chrysler Center, University of Michigan, Ann Arbor, MI 48109. □

Software Release Summary

This article summarizes major changes made in the 1.10 version of Cray Research software.

COS

With this release, the COS job control language was significantly enhanced by such features as procedure definition, conditional processing and iterative control statement processing capabilities.

The **procedure definition block control statements** allow the user to copy a series of control statements and/or data to a library and call them at a later time. When a procedure is called, character strings within the procedure can be substituted with either default values or values from the procedure call statement. The COS control statements provided for this capability are PROC, a prototype statement, ENDPROC, and &DATA.

Conditional processing can now be specified through the use of four new COS control statements. The conditional control statements IF, ELSEIF, ELSE and ENDIF allow the user to specify a control statement sequence to be processed only if certain conditions are true.

Iterative control statement processing is now possible. An iterative block contains a control statement sequence that is to be processed more than once during the processing of a job. The COS iterative block control statements are LOOP, EXITLOOP and ENDLOOP. The EXITLOOP statement indicates the normal exit condition for the loop. If its expression is true, the loop is exited; if it is false, loop execution continues with the subsequent statements. Control returns to the beginning of the loop when the ENDLOOP statement is encountered.

A number of new control statements were added with this release. The new **ROLLJOB control statement** allows the user to force a job to be rolled out so that it can be restarted in case of a system failure. **SET** allows the user to change the value of a job control language variable. The **LIBRARY statement** allows the user to specify which JCL library datasets are to be searched during job processing and in what order. It can also be used to print the current or new searchlist to the logfile.

The new CNS parameter on the CALL control statement specifies that the control statement following is a procedure calling statement containing parameters for procedure string substitution. The **ECHO control statement** or **CSECHO macro** enables user control of message classes to be written to the job logfile. The **PRINT control statement** writes the value of an expression on the logfile in three different formats: 16-digit decimal integer, 22-digit octal value, and as an ASCII string.

The **FLODUMP control statement** recovers and dumps flowtrace tables when a program aborts with flow tracing active. The flow trace tables are dumped in FORTRAN flow trace format.

The **SYSREF statement** generates a global cross-reference listing for a group of CAL or APML assembly language programs.

CAL

Three new pseudo instructions were added to the CRAY-1 Assembly Language (CAL) with this release. The **TEXT pseudo instruction** allows the user to declare the beginning of a text source, and **ENDTEXT** terminates the text source. Text source is not normally listed, and symbols defined within text source are not listed in the cross-reference unless referred to elsewhere in the program. The **MODULE pseudo instruction** defines the contents of the Program Descriptor Table module type field.

CFT

The major change in CFT this time is that it now adheres to the new ANSI standard. CFT is an extended version of the American National Standards Institute programming language, **ANSI X3.9-1978,** better known as FORTRAN 77. The compiler also accepts most of the older ANSI X3.9-1966 syntax. The major newly-implemented FORTRAN 77 features include the character data type and generic functions.

The **character data type** can be used to replace Hollerith constants in a machine-independent way. It allows for concatenation, comparison and substringing of character constants or variables.

The **generic function feature** allows CFT to select a specific intrinsic function appropriate to the form of the function's arguments. For example, SIN may be used with a real, complex, or double precision argument and CFT will automatically select SIN, CSIN, or DSIN, respectively. This feature is particularly useful when moving programs between machines using different word lengths. Variables can thus be declared double precision on a small machine and real on the large system, without requiring any changes to the program's function references.

With this release, several **I/O enhancements** were made. The OPEN, CLOSE and INQUIRE statements now accept character arguments and return more information about a dataset's status. **List-directed I/O** is now supported and several small-scale features have been added to the other I/O statements.

In addition, several CDIR options have been added or expanded and the DO processor has been optimized. This allows somewhat greater programmer control over loop vectorization and enhanced efficiency. □

An update on 1981 marketing efforts, installations to date, new corporate facilities, and people and product news

1981 installations to date

The first few months of 1981 have proven to be the busiest months ever for the Cray Research manufacturing division. Four new computer systems have been installed and have completed acceptance testing since January 1. In addition, four systems were re-installed. Among the new installations so far this year are the first CRAY-1 to be installed at a petroleum company, the company's first shipment to France, and the first installation of a CRAY-1/S with an I/O Subsystem outside of the company.

In late February, a CRAY-1 S/1300 was installed at ARCO Oil and Gas Company. A field upgrade of this system to two million words is scheduled for later in 1981. ARCO, a division of Atlantic Richfield Company, explores for, produces and sells oil, natural gas and natural gas liquids in North America.

A one-million word CRAY-1/S was installed for the Groupement D'exploitation de Techniques Informatiques Avancees of France.

A two-million word CRAY-1/S was installed at Los Alamos Scientific Laboratory in Los Alamos, New Mexico as an upgrade to their one-million word memory system.

A CRAY-1 S/500 was installed in the U.K. at the Science Research Council's Daresbury Laboratory. This system replaced a CRAY-1B system, which was moved to AERE, Harwell, also in the U.K. Another re-installed system was that completed for the Shell Company outside the United States. Yet another re-installation took place at the Lawrence Livermore Laboratory.

The CRAY-1/S at Cray Research headquarters was field upgraded to an S/1300 during the first quarter of 1981. Originally installed in December of 1980, this system is used by the Software Development division of the company. Additionally, the CRAY-1 at Century Research Center Corporation, Tokyo, Japan was upgraded to a million words.

P. A. Gregory

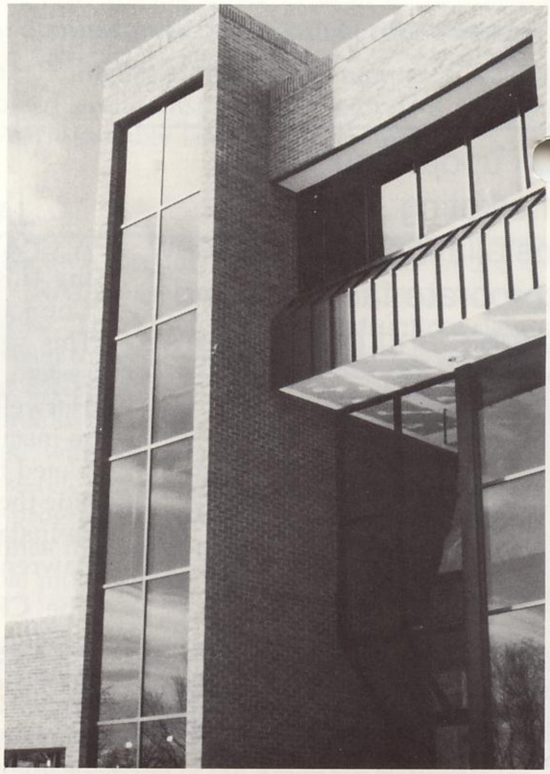
Peter A. Gregory named director of product management

Peter A. Gregory has been appointed director of product management of Cray Research. Mr. Gregory is responsible for all aspects of product management for the company, including market research, forecasting, business management and announcement activities. Prior to joining Cray Research, Mr. Gregory held positions in marketing and product management for International Business Machines Corporation, including director of systems marketing in Europe.

Cray Labs moves to new facility in Boulder

The 42 employees of Cray Laboratories recently moved to a new lab building in Boulder, Colorado. The move to the 30,000 square foot building consolidated workers previously occupying temporary offices in three different locations in Boulder. The new facility contains a Class 100 clean room, circuit test and assembly areas, and a computer-aided design laboratory.

The new Cray Labs facility in Boulder, Colorado



The clean room is equipped for integrated circuit development and customization work. The air filtration system in the room is quite sophisticated, allowing no more than 100 particles five-tenths of a micron in size per cubic foot. Room air is filtered several hundred times an hour.

The lab has also purchased an electron beam microfabricator and a scanning electron microscope for their integrated circuit work.

Cray Laboratories was established in May of 1979 as the company's wholly-owned research and development subsidiary. Its efforts are concentrated on designing advanced hardware and software systems. The geographical separation between the labs and the rest of the company has allowed the Boulder employees to work on future products and technologies without involvement in the parent company's day-to-day operations.

New product announcement

In April, Cray Research announced a major product enhancement that expands the capabilities of the CRAY-1 S Series of Computer Systems, or more specifically, the capabilities of the I/O Subsystem. Introduced with the CRAY-1 S Series of Computer Systems in 1979, the I/O Subsystem was itself an architectural extension of the original CRAY-1 system. This latest product enhancement provides up to eight million words of Buffer Memory, dual high performance channels for streaming data to Central Memory, and support for on-line magnetic tape. This enhancement demonstrates Cray Research's continuing commitment to further development of the CRAY-1 system, with the ultimate goal of providing the best computer system possible for engineering and scientific users.

Marketing update for the petroleum industry

Recently, Cray Research received two CRAY-1 S Computer System orders from the petroleum industry, the company's second and third orders from major petroleum companies.

On March 24, Exxon Production Research Company ordered a CRAY-1 S/4400 valued at approximately $17 million. This system, which will be purchased, is scheduled for installation during the fourth quarter of 1981. It will be the largest single system Cray Research has installed to date, with four million words of memory, an I/O Subsystem and Buffer Memory in excess of 19 billion bytes.

The company also received an order for a CRAY-1B Computer System from the Shell Company. The system will be leased and installation has been made outside the United States.

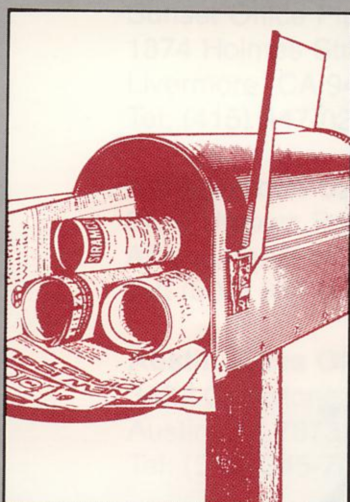
Work progresses on new printed circuit building

Construction of a new printed circuit board facility in Chippewa Falls, Wisconsin began several months ago. The decision to build this facility was based on the successes of the company's initial printed circuit facility near the manufacturing building.

The original PCB facility was brought on-line in 1980, and by year-end was meeting one-fourth of production's total needs. The operation has provided a cost advantage to the company and has also given engineering a significant advantage in their development work.

The facility currently under construction will be 17,000 square feet in size, and the company expects that it will be operational by early 1982. □

USER NEWS



University of Minnesota orders CRAY-1

Because the Minneapolis area is one of the top areas in the country for supercomputer development, it seems only fitting that the local university should have a large-scale computer on campus. Therefore, when the University of Minnesota ordered a computer from Cray Research on June 12, there was much cause for excitement. Said John Rollwagen, president of Cray Research, "This will be the first Cray system installed in Minnesota other than the one at our Mendota Heights data center. We are delighted to be a part of the state's scientific research community."

The nation's universities are looking at investing in large-scale scientific computers because complex research in areas such as weather, energy, agriculture and oil exploration require a great deal of computing power. Until now, the University of Minnesota has been purchasing a number of smaller computers to replace its older equipment. However, the CRAY-1 purchase is especially significant in that it makes Minnesota the first university in the country to have a supercomputer installed on its campus. Frank Verbrugge, director of the University's computer services, thinks the CRAY-1 can handle so much research that it will prove more practical than smaller computers in the long run.

The University's new acquisition is a CRAY-1/B system, and installation is scheduled at the Lauderdale Computer Center in the third quarter of 1981. The system has a value of approximately $4.5 million and will be purchased. The CRAY-1 will take on batch processing, replacing an older large-scale computer. It also will be used in research work, including theoretical studies on molecular orbital calculations in quantum chemistry and experimental and theoretical studies in high-energy physics, said Verbrugge.

In commenting about acquisition of the system, Verbrugge said, "We look forward to a mutually beneficial program of systems development by which the CRAY-1 computer becomes an integral part of the University's general purpose computer network. Software developments can include a variety of applications programs and the implementation of additional higher-level languages on the CRAY-1 system, and additional software developments by which the CRAY-1 can communicate with the other systems in the University's computer complex." □

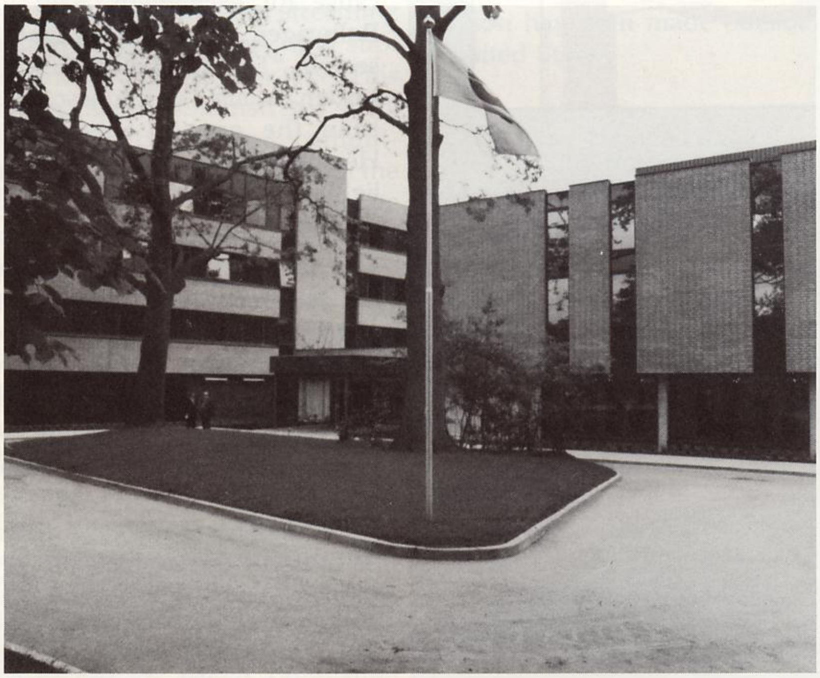
ECMWF accurately forecasts freeze in Florida

The European Centre for Medium Range Weather Forecasts (ECMWF) has long contended that reliable forecasts for a week ahead are of great economic value to industries such as agriculture, transportation, construction, shipping and energy. Early this year, representatives of ECMWF attending a conference in Tallahassee, Florida were able to follow their predictions of a killer freeze.

The first economically destructive freeze in four years struck Florida on January 12-13, 1981. Temperatures in mid-January broke state record lows set as long ago as 1886, destroying close to 20 percent of central Florida's huge orange crop. Utility companies reported record electricity demand, forcing rotating blackouts and requests for conservation. Fishing along the coast was brought to a standstill at the height of the oyster and shrimp season.

ECMWF had been predicting a sharp temperature drop for Florida a week in advance. Dr. L. Bengtsson, Head of Research at ECMWF, brought a copy of this forecast with him to the International Conference on Early Results of FGGE, held in Tallahassee. Conference delegates followed the progress of the prediction as it verified day by day.

Producing a ten-day forecast within a few hours requires incrementing nearly two million elements at each time step. The CRAY-1 at ECMWF has made feasible highly complex ten-day forecast models, enabling accurate forecasting as witnessed in Tallahassee a few months ago.☐



The headquarters of ECMWF in Reading, England

CORPORATE ADDRESSES

Sales Offices

Central Regional Sales Office
5330 Manhattan Circle, Suite F
Boulder, CO 80303
Tel: (303) 499-3055

Eastern Regional Sales Office
11120 New Hampshire Avenue
Silver Spring, MD 20904
Tel: (301) 681-9514

Western Regional Sales Office
Sunset Office Plaza
1874 Holmes Street
Livermore, CA 94550
Tel: (415) 447-0201

Albuquerque Sales Office
6400 Uptown Boulevard N.E., Suite 361 W
Albuquerque, NM 87110
Tel: (505) 881-8278

Austin Sales Office
3432 Greystone, Suite 108
Austin, TX 78731
Tel: (512) 345-7034

Chicago Sales Office
625 N. Michigan, Suite 500
Chicago, IL 60611
Tel: (312) 944-1183

Houston Sales Office
5858 Westheimer, Suite 602
Houston, TX 77057
Tel: (713) 975-8998

Los Angeles Sales Office
2134 Main Street, Suite 260
Huntington Beach, CA 92648
Tel: (714) 960-7611

Pittsburgh Sales Office
One Northgate Square
Greensburg, PA 15601
Tel: (412) 832-8120

Seattle Sales Office
1750 112th Avenue N.E., Suite C-237
Bellevue, WA 98004
Tel: (206) 454-4438

Washington Special Systems
312 Marshall Avenue
Laurel, MD 20810
Tel: (301) 490-1800

International Subsidiaries

Cray Research France, S.A.
3, Avenue de Madrid
Neuilly Sur Seine 92200
France
Tel: 01-747-9440

Cray Research GmbH
Wartburgplatz 7
8000 Munich 40
West Germany
Tel: 089-363076
TLX: 8415213211

Cray Research Japan, Limited
New Aoyama Bldg, West 1661
1-1 Minami-Aoyama 1-chome
Minato-Ku, Tokyo 107
Japan
Tel: 03-403-3471
TLX: T8128893

Cray Research (UK) Limited
Seymour House, The Courtyard
Wokingham, Berkshire
England
Tel: 0734-791180
TLX: 848841

CRAY RESEARCH, INC.

Corporate Headquarters
Cray Research, Inc.
1440 Northland Drive
Mendota Heights, MN 55120
Tel: (612) 452-6650
TLX: 298444