```
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

0000: D8 A2 FF 9A A2 C8 86 B2 20 1F 1F 20 6A 1F C5 F3
0010: F0 F6 85 F3 C9 0C D0 0F A2 1F B5 70 95 50 CA 10
0020: F9 86 DC A9 CC D0 12 C9 0E D0 07 20 B2 02 A9 EE
0030: D0 07 C9 14 D0 0B 20 A2 03 85 FB 85 FA 85 F9 D0
0040: BF C9 0F D0 06 20 4B 03 4C 9D 01 4C 96 01 10 00

0070: 03 04 00 07 02 05 01 06 10 17 11 16 12 15 14 13
0080: 73 74 70 77 72 75 71 76 60 67 61 66 62 65 64 63
0090: F0 FF 01 10 11 0F EF F1 DF E1 EE F2 12 0E 1F 21
00A0: 0B 0A 06 06 04 04 04 04 02 02 02 02 02 02 02 02

0100: A6 B5 30 5C A5 B0 F0 08 E0 08 D0 04 C5 E6 F0 2E
0110: F6 E3 C9 01 D0 02 F6 E3 50 1E A0 0F A5 B1 D9 60
0120: 00 F0 03 88 10 F8 B9 A0 00 D5 E4 90 04 94 E6 95
0130: E4 18 08 75 E5 95 E5 28 E0 04 F0 03 30 31 60 A5
0140: E8 85 DD A5 00 85 B5 20 4B 03 20 B2 02 20 00 02
0150: 20 B2 02 A9 08 85 B5 20 09 02 20 31 03 4C 80 17
0160: E0 F5 D0 0B A5 60 C5 B1 D0 04 A5 00 85 B4 60 50
0170: FD A0 07 A5 B1 D5 60 00 F0 05 88 F0 F1 10 F6 B5
0180: A0 00 D5 E2 90 02 95 E2 C6 B5 A5 FB C5 B5 F0 03
0190: 20 25 03 E6 B5 60 C9 08 B0 12 20 EA 03 A2 1F B5
01A0: 50 C5 FA F0 03 CA 10 F7 86 FB 86 B0 4C 00 00 00

0200: A2 10 A9 00 95 DE CA 10 FB A9 10 85 B0 C6 B0 10
0210: 01 60 20 1E 03 A4 B0 A2 08 86 B6 C0 08 10 41 C0
0220: 06 10 2E C0 04 10 1F C0 01 F0 09 10 0E 20 8E 02
0230: D0 FB F0 D9 20 9C 02 D0 FB F0 D2 A2 04 86 B6 20
0240: 5C 02 D0 FB F0 C7 20 9C 02 A5 B6 C9 04 D0 F7 F0
0250: BC A2 10 86 B6 20 8E 02 A5 B6 C9 08 D0 F7 F0 AD
0260: A2 06 86 B6 20 CA 02 50 05 30 03 20 00 01 20 1E
0270: 03 C6 B6 A5 B6 C9 05 F0 EB 20 CA 02 70 8F 30 8D
0280: 20 00 01 A5 B1 29 F0 C9 20 F0 EE 4C 0D 02 20 CA
0290: 02 30 03 20 00 01 20 1E 03 C6 B6 60 20 CA 02 90
02A0: 02 50 F9 30 07 08 20 00 01 28 50 F0 20 1E 03 C6
02B0: B6 60 A2 0F 38 B4 60 A9 77 F5 50 95 60 94 50 38
02C0: A5 77 F5 50 95 50 CA 10 EB 60 A5 B1 A6 B6 18 75
02D0: 8F 85 B1 29 88 D0 42 A5 B1 A2 20 CA 30 0E D5 50
02E0: D0 F9 E0 10 30 33 A9 7F 69 01 70 01 B8 A5 B5 30
02F0: 24 C9 08 10 20 48 08 A9 F9 85 B5 85 B4 20 4B 03
0300: 20 B2 02 20 09 02 20 2E 03 28 68 85 B5 A5 B4 30
0310: 04 38 A5 FF 60 18 A9 00 60 A9 FF 18 B8 60 A6 B0
0320: B5 50 85 B1 60 20 4B 03 20 B2 02 20 09 02 20 B2
0330: 02 BA 86 B3 A6 B2 9A 68 85 B6 68 85 B0 AA 68 95
0340: 50 68 AA 68 85 B1 95 50 4C 70 03 BA 86 B3 A6 B2
0350: 9A A5 B1 48 A8 A2 1F D5 50 F0 03 CA 10 F9 A9 CC
0360: 95 50 8A 48 A6 B0 B5 50 94 50 48 8A 48 A5 B6 48
0370: BA 86 B2 A6 B3 9A 60 A6 E4 E4 A0 D0 04 A9 00 F0
0380: 0A A6 E3 D0 06 A6 EE D0 02 A9 FF A2 04 86 B5 C5
0390: FA 90 0C F0 0A 85 FA A5 B0 85 FB A5 B1 85 F9 4C
03A0: 1F 1F A6 DC 10 17 A5 F9 D5 DC D0 0F CA B5 DC 85
03B0: FB CA B5 DC 85 F9 CA 86 DC D0 1A 85 DC A2 0C 86
03C0: B5 86 FA A2 14 20 02 02 A2 04 86 B5 20 00 02 A6
03D0: FA E0 0F 90 12 A6 FB B5 50 85 FA 86 B0 A5 F9 85
03E0: B1 20 4B 03 4C 00 00 A9 FF 60 A2 04 06 F9 26 FA
03F0: CA D0 F9 05 F9 85 F9 85 B1 60 00 00 00 00 00 00

1780: 18 A5 80 65 EB 65 EC 65 ED 65 E1 65 DF 38 E5 F0
1790: E5 F1 E5 E2 E5 E0 E5 DE E5 EF E5 E3 B0 02 A9 00
17A0: 4A 18 69 40 65 EC 65 ED 38 E5 E4 4A 18 69 90 65
17B0: DD 65 DD 65 DD 65 DD 65 E1 38 E5 E4 E5 E4 E5 E5
17C0: E5 E5 E5 E0 A6 B1 E0 33 F0 16 E0 34 F0 12 E0 22
17D0: F0 0E E0 25 F0 0A A6 B0 F0 09 B4 50 C0 10 10 03
17E0: 18 69 02 4C 77 03
```


MICROCHESS

Micro-Ware Ltd presents

MICROCHESS

NOW YOU CAN PLAY CHESS WITH YOUR KIM-1
6502 BASED MICROPROCESSOR SYSTEM.

MICROCHESS REQUIRES NO ADDITIONAL MEMORY.
The program and data occupy only
1118 of the 1152 bytes of available RAM.

MICROCHESS REQUIRES NO ADDITIONAL PERIPHERALS.
All moves are entered and displayed
via the KIM keyboard and LED display.

MICROCHESS PLAYS CHESS.
Although a good chessplayer will probably
beat the program, he will be surprised
again and again by challenging moves.

MICROCHESS HAS SEVERAL LEVELS OF PLAY.
You may set the program up for 3, 10,
or 100 seconds per move. Change the
speed at any time during the game!

MICROCHESS IS EXPANDABLE AND FULLY DOCUMENTED.
You receive a Player's Manual, complete
annotated source listing, and Program
Documentation describing the strategic
algorithms. Instructions are provided
for modification, expansion or system
conversion. Experiment with your own
strategies by replacing one simple subroutine.

MICROCHESS COSTS ONLY $10.00.
Send your cheque or money order today to:

MICROCHESS (KIM-1),
MICRO-WARE LIMITED,
27 Firstbrooke Road,
Toronto, Ontario,
Canada, M4E 2L2.

SAMPLE GAMES PLAYED BY MICROCHESS

Human	MICROCHESS (Ruy Lopez)	
1. P-K4	P-K4	(1)
2. N-KB3	N-QB3	(1)
3. B-N5	N-B3	(1)
4. O-O	NxP	(1)
5. P-Q4	B-K2	(1)
6. Q-K2	N-Q3	(1)
7. BxN	NPxB	(1)
8. PxP	N-N2	(1)
9. N-B3	O-O	(1)
10. R-K1	B-N5	(37)
11. B-Q2	Q-K2	(46)
12. P-QR3	B-R4	(59)
13. P-N4	B-N3	(47)
14. N-K4	P-Q4	(60)
15. PxP (e.p.)	PxP	(68)
16. P-B4	B-KB4	(65)
17. P-QB5	PxP	(82)
18. NxP	Q-B2	(121)
19. Q-K7	QxQ	(170)
20. RxQ	BxN	(58)
21. PxB	NxP	(45)
22. R-QB1	N-N6	(50)
23. RxBP	KR-Q1	(50)
24. R-B7	K-B1	(53)
25. RxBP+	K-N1	(4)
26. RxB	P-KN3	(32)
27. R(5)-B7	R-Q6	(27)
28. B-N4	R-Q8+	(25)
29. N-K1	RxN+	(30)
30. BxR	R-K1	(14)
31. R-N7+	K-R1	(2)
32. R-R7+	K-N1	(2)
33. R(B7)-N7+	K-B1	(2)
34. B-N4+	N-B5	(2)
35. BxN+	R-K2	(2)
36. BxR+	K-K1	(1)
37. B-B5	P-R3	(3)
38. R-R8+ (mate)		

MICROCHESS (Queen's Indian)		Human
1. P-Q4	(1)	N-KB3
2. P-QB4	(1)	P-K3
3. N-KB3	(1)	P-QN3
4. P-KN3	(1)	B-N2
5. B-N2	(1)	B-K2
6. O-O	(1)	O-O
7. N-B3	(1)	N-K5
8. Q-B2	(1)	NxN
9. QxN	(1)	P-Q3
10. P-K4	(58)	BxP
11. R-K1	(55)	P-Q4
12. B-B4	(65)	N-Q2
13. PxP	(87)	PxP
14. Q-B6	(91)	N-KB3
15. N-N5	(122)	R-K1
16. B-R3	(158)	B-N5
17. R-K2	(198)	R-K2
18. R-B1	(134)	Q-KB1
19. R-R1	(139)	QR-K1
20. R-B1	(119)	B-Q6
21. RxR	(187)	RxR
22. R-Q1	(98)	R-K8+
23. RxR	(13)	BxR
24. BxP	(68)	P-N3
25. QxN	(65)	B-Q7
26. N-KB3	(63)	Q-N5
27. Q-Q8+	(61)	K-N2
28. B-K5+	(60)	K-R3
29. Q-R4+ (mate)	(60)	

(60) indicates the approximate time in seconds for the computer to make its move

Please hurry! I want to play chess with my KIM.

Player's Manual, Programmer's Manual, Source Listing @ 10.00 $10.00

Paper Tape in MOS Technology format. @ 1.00 _____

KIM cassette. @ 3.00 _____

Ontario Residents add 7% P.S.T. _____

Total Enclosed $ _____

Send to: MICROCHESS (KIM-1)
MICRO-WARE LIMITED,
27 Firstbrooke Road,
Toronto, Ontario,
Canada, M4E 2L2.

MICROCHESS

A CHESS PLAYING PROGRAM

FOR THE 6502 MICROCOMPUTER

BY PETER JENNINGS

MICROCHESS is available on KIM cassette for $5.00.

M I C R O C H E S S

KIM Cassette Loading Instructions

1 Enter (RS) to reset KIM.
2 Enter (AD) 0 0 F 1 (DA) 0 0 to reset decimal flag.
3 Enter (AD) 1 7 F 9 (DA) C 1 to enter tape ID.
4 Enter (AD) 1 8 7 3 (GO) to begin read routine.
5 Start your cassette player.
6 When you see: 0000 D8 stop your cassette player.
7 Enter (RS) (AD) 1 8 7 3 (GO) to read block 2.
8 Start your cassette player.
9 When you see: 0000 D8 stop your cassette player.
10 Enter (RS) (GO) to start program execution.

If you wish KIM to play a specific opening, enter the ID in address 17F9 and load the opening data. Enter (RS) before and after each tape load.

Data for Openings

Microchess plays white	black	Opening
A0	A1	Four Knights
A2	A3	French Defence
A4	A5	Ruy Lopez
A6	A7	Queen's Indian
A8	A9	Guioco Piano

Remember to always press (RS) between each tape load. Otherwise, data at 0100 and 0101 may be overwritten by the stack. Verify these locations against the program listing if you have trouble executing the program.

A second copy of the two main programs can be found after the openings.

M I C R O C H E S S

MICROCHESS was originally conceived as a program which would play chess using only a minimum hobbyist microcomputer system. The program designed will run on a KIM-1, 6502 based system, using only 1.1 Kbytes of RAM. Elimination of some unnecessary features would even allow an implementation in less than 1K.

Although MICROCHESS does not play an expert level of chess, it will play a reasonable game in most instances. In addition, it can provide a useful opponent for practising checkmates, learning openings, and sharpening general playing skills.

The program has been carefully designed to allow the average user to expand or modify the basic package to suit the requirements of his particular system configuration, or to experiment with his own ideas for improvement of the playing strategy.

User documentation supplied with the MICROCHESS program consists of a Player's Manual, a complete source program listing, and a Programmer's Manual, which explains the operation of the program and includes suggestions for expansion and modifications.

Micro-Ware Ltd.
27 Firstbrooke Road,
Toronto Ontario,
Canada. M4E 2L2

TABLE OF CONTENTS

PLAYER'S MANUAL

PROGRAMMER'S MANUAL

SOURCE LISTING

MICROCHESS

PLAYER'S MANUAL

MICROCHESS was designed to play a game of chess using the KIM-1 microcomputer system with no additional memory or peripherals. The human player's moves are entered on the self contained keyboard and the computer's responses are flashed on the LED display. Slight program alterations will permit the user to run the program using a teletype, CRT terminal, or another 6502 based system, (see the Programmer's Manual for details). All references in this manual assume that the KIM keyboard and display are being used.

LOADING THE PROGRAMS

Since the KIM-1 memory is divided into two non-contiguous segments, the program must be loaded in two sections. The first section will contain the program and data for the lower 1K of available memory between addresses 0000 and 03FF. The second section will contain the program segment between locations 1780 and 17E6. In addition, short program loaders may be used to enter the data necessary to use different "canned openings", which are stored between 00C0 and 00DB. Since sections of program reside in page one, which is normally reserved for the program stack, it is advisable to reset the stack pointer using the [RS] key before each load. In addition, it is prudent to check locations 0100 and 0101 before executing the program to ensure that they have not been inadvertently altered.

MICROCHESS NOTATION

In order to keep memory requirements to a minimum, (an absolute necessity when programming chess in the 1K environment of the KIM-1), it has been necessary to use a special octal chess notation. Each square on the chess board is uniquely identified by a two digit octal number as shown below. The first digit specifies the rank (0 to 7) from the computer's end of the board. The second digit specifies the file (0 to 7) from the player's left. Moves are specified uniquely by the FROM square and the TO square using this notation.

COMPUTER

00	01	02	03	04	05	06	07
10	11	12	13	14	15	16	17
20	21	22	23	24	25	26	27
30	31	32	33	34	35	36	37
40	41	42	43	44	45	46	47
50	51	52	53	54	55	56	57
60	61	62	63	64	65	66	67
70	71	72	73	74	75	76	77

PLAYER

MICROCHESS COMMAND KEYS

The following keys are used as commands while playing chess with the MICROCHESS program.

[GO] This key is depressed immediately after loading the tape in order to start the program execution, or to restart the program after a temporary exit. No change occurs in the display after the [GO] key has been depressed. After execution begins the key has no effect on the system at all.

[ST] This key is used to leave the MICROCHESS program and enter the KIM monitor in order to examine or change memory contents while playing a game. Under no circumstances should this key be pressed when the computer is contemplating its move. Only when the system is displaying a move is it permissable to press the [ST] key.

[C] This key CLEARS the internal chessboard and resets it to begin another game. The board is set up with the computer playing white. CCCCCC is displayed to indicate that the board has been reset.

[E] This key EXCHANGES the computer's men with your men. The actual position of the board is unchanged. If [C] is pressed, followed immediately by [E], the board will be set up to begin a game with the computer playing black. By pressing [PC] followed by [E] followed by [PC] . . . the computer will play a game against itself, displaying the moves as it goes. EEEEEE is displayed immediately after the [E] key is pressed to verify operation.

[F] This key is used to move the piece on the FROM square to the TO square to register the player's move, or to move one of the computer's men if desired.

[PC] This key instructs the computer to PLAY CHESS. The computer analyses the current position and formulates its optimum move. The display will darken and flash until the move has been decided. When it relights the move is displayed.

THE COMPUTER'S MOVE

The computer moves are displayed in the format shown below:

[piece¦FROM square¦TO square]

[piece¦ The piece which the computer is indicating that it wishes to move is encoded according to the table below:

0 - KING	4 - King Bishop	8 - K R Pawn	C - K B Pawn
1 - Queen	5 - Queen Bishop	9 - Q R Pawn	D - Q B Pawn
2 - King Rook	6 - King Knight	A - K N Pawn	E - Q Pawn
3 - Queen Rook	7 - Queen Knight	B - Q N Pawn	F - K Pawn

¦FROM square¦ The FROM and TO squares are indicated using the micronotation shown above.

For example the display [0F 13 33] indicates that the King Pawn is to be moved from King Pawn 2 to King Pawn 4. (This assumes that the computer is playing white.)

ENTERING YOUR MOVE

Your moves are described to the computer using the same octal notation described above. It is not necessary to enter the type of piece being moved, just the FROM square and TO square locations.

The computer verifies the input by indicating in the left two digits the piece located on the FROM square. The first digit will be 0,1, or F. 0 indicates that the piece on the from square is one of the computer's men. 1 indicates that the piece is one of your men. F indicates that there is no piece on the FROM square.

The second digit indicates the type of piece located on the FROM square using the same hexadecimal code shown above.

If you have made an error in entering your move at this point just continue to press the appropriate keys. The numbers will scroll from right to left until the correct move is displayed.

For example, if you punch 6 3 4 3 and see the display [1F 63 43], the 1F indicates that the FROM square (63), contains the King Pawn and that you are preparing to move it to the square 43.

When you have entered and verified the move, depress the [F] key to register the move on the internal chess board. The first two digits of the display will be changed to FF to indicate that the FROM square is now unoccupied. If the TO square had been occupied, the previous occupant will have been captured automatically.

You may make as many moves in this manner as you wish, moving either your own men or the computer's. No verification of the legality of the moves is carried out. Illegal moves are accepted and executed as easily as legal moves, so care should be taken that you do not accidentally move in an illegal manner. Since the computer does not make a point of warning you if your king is in check, you must be careful not to leave this situation after your move. The computer will usually take off your king on its subsequent move if this is possible.

SPECIAL MOVES

CASTLING: You may make a castling move by making two moves in succession in the normal manner. First move the king to its new square, then move the rook. Remember to depress [F] after each move. The computer has no provision for castling during the middle game or end game, but may castle during the opening. If this occurs it will indicate a move of the king two squares over. You must complete the move for the computer by moving the rook for it. Just enter the appropriate TO and FROM square followed by [F] to make the move, then, go ahead and make your own move.

EN PASSANT: In order to capture en passant you must break the move into two separate components. First, move your pawn laterally to capture the computer's pawn. Then, move your pawn forward to its appropriate final square. Do not forget to depress [F] after each move to register it internally. Note that the computer cannot capture en passant itself and will not recognize the danger of your en passant captures in considering its double pawn moves.

QUEENING PAWNS: If you should succeed in pushing a pawn to the eighth rank (rank 7 in micronotation), it will be necessary for you to manually set up the queen on that square. Because of the internal representation of the position it is possible only to have one Queen per side at a time. Therefore, if you already have one, you will have to choose a rook, bishop, or knight instead. To replace the pawn with a Queen the following steps should be carried out.

1) Use the [ST] key to exit from the MICROCHESS program and return control to the KIM monitor.

2) Find the pawn using the table of piece locations below. Confirm by its position that it is the correct one. Remove it from the board by entering the data 'CC', which indicates a captured piece.

3) Enter the address of the queen (0061). This memory location should now contain 'CC', assuming the queen has been lost.

4) Press [DA] and enter the new location for the Queen, which is the square the pawn moved to. (e.g. 07)

5) Press [PC] followed by [GO] to reenter the MICROCHESS program. Continue in the normal manner from this point.

If the computer should push a pawn to the eighth rank, it will be necessary for you to replace the pawn with a Queen, or the highest piece available. Use the same procedure as above. The computer's Queen should be stored at address 0051.

LEVEL OF PLAY

There are several sections of the program which can be bypassed in order to reduce the computer's response time in a given situation. This will reduce the quality of play accordingly. The strategy levels and data changes are outlined below.

LEVEL	LOCATION 02F2	LOCATION 018B	AVGE TIME PER MOVE
SUPER BLITZ	00	FF	3 seconds
BLITZ	00	FB	10 seconds
NORMAL	08	FB	100 second

POSITION VERIFICATION

Occassionally, while playing a game, you will come to the sudden realization that the computer is seeing a different board setup from the one you have. This results from your misinterpretation of one of its moves, from entering one of your moves incorrectly, or from forgetting to press [F] to register your move.

It is possible in this situation to sneak a peek at the location of each piece as it is internally stored in order to verify its location on the board. To do this press [ST] to exit the MICROCHESS program and enter the KIM monitor. Then look at the addresses shown below to determine where the computer thinks each piece is. Afterwards, return to the chess program by pressing [PC] followed by [GO].

MEMORY LOCATIONS FOR THE PIECES

COMPUTER PIECES		YOUR PIECES
0050	King	0060
0051	Queen	0061
0052	King Rook	0062
0053	Queen Rook	0063
0054	King Bishop	0064
0055	Queen Bishop	0065
0056	King Knight	0066
0057	Queen Knight	0067
0058	K R Pawn	0068
0059	Q R Pawn	0069
006A	K N Pawn	006A
005B	Q N Pawn	006B
005C	K B Pawn	006C
005D	Q B Pawn	006D
005E	Q Pawn	006E
005F	K Pawn	006F

IMPORTANT NOTE:
Never depress the [ST] key while the computer is contemplating its move. Important parameters are stored in the same area of memory used by the KIM monitor programs. Reentry after these locations have been altered will probably destroy the board position.

NOTES

As mentioned above, there are three types of moves which the current version of MICROCHESS does not play. These are castling, en passant pawn captures, and queening of pawns. In order to make the game fair some players adopt one of the two following strategies. Recognizing that the computer cannot make these moves, some players choose not to make them themselves, thus both players suffer the same restrictions. On the other hand, other players have decided to help the computer by watching for appropriate castling or en passant situations and making the moves on the computer's behalf at that time. Of course, you may always play without regard to the computer's disadvantage, allowing it to fend for itself as best it can.

If you are an above average player, you may find that the MICROCHESS program is below your level of play and hence, always loses. You can add to the challenge of the game in the same way that you might against an inexperienced human player. Remove one or more of your pieces at the start of the game and see if you can come back from a position of disadvantage. The easiest way to remove a piece is to move one of the computer's men to the square of the piece you wish to remove, and then move it back to its original square.

MICROCHESS

PROGRAMMER'S MANUAL

The program can be divided into three basic functional units.

I Control and Input/Output. This section comprises the initialization routines, the input and output routines, and the main entry into the move generation and evaluation routines.

II Move Generation and Data Collection. This program group generates the moves available to the computer, one at a time. For each of these moves, data are collected regarding available continuation moves, the threats of possible reply moves, and the gain or loss from subsequent piece exchanges.

III Strategic Analysis. The data collected by the move generation routines are analysed by a mathematical algorithm which assigns a value to each available move. The move with the highest assigned value will be the move that the computer selects.

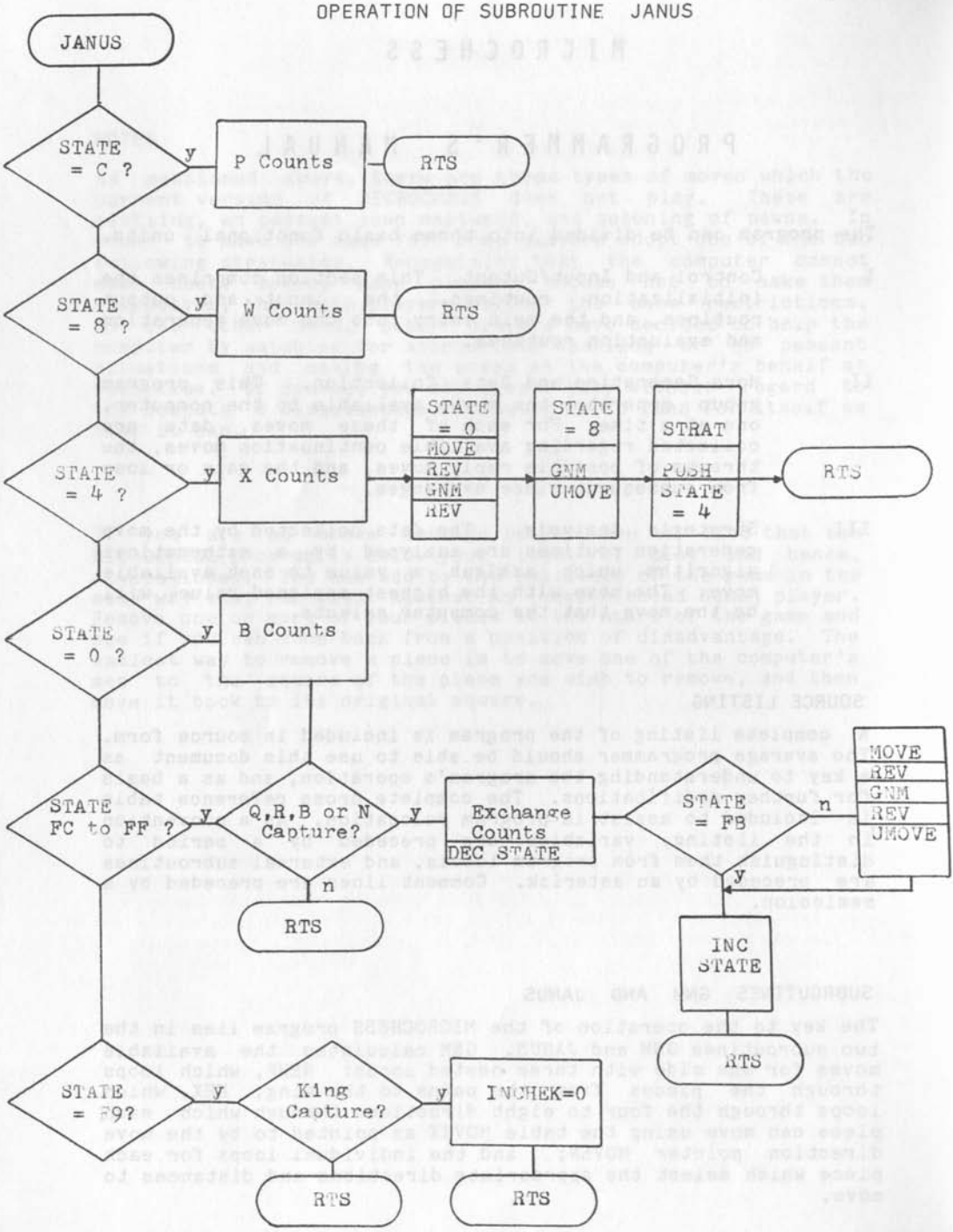
SOURCE LISTING

A complete listing of the program is included in source form. The average programmer should be able to use this document as a key to understanding the program's operation, and as a basis for further modifications. The complete cross reference table is included to assist in program relocation. As a convention in the listing, variables are preceded by a period to distinguish them from program labels, and external subroutines are preceded by an asterisk. Comment lines are preceded by a semicolon.

SUBROUTINES GNM AND JANUS

The key to the operation of the MICROCHESS program lies in the two subroutines GNM and JANUS. GNM calculates the available moves for one side with three nested loops: NEWP, which loops through the pieces from the pawns to the king; NEX, which loops through the four to eight directions through which each piece can move using the table MOVEX as pointed to by the move direction pointer MOVEN; and the individual loops for each piece which select the appropriate directions and distances to move.

OPERATION OF SUBROUTINE JANUS



After each move has been calculated by GNM, the subroutine JANUS is called. JANUS uses the value of STATE to determine which portion of the analysis the computer is working on and directs it to the appropriate continuation routines. As can be seen from the simplified flow chart of JANUS' operation, JANUS often alters the value of STATE and calls the subroutine GNM again. This series of recursive subroutine calls calculates approximately 20,000 moves per second-- over 2 million moves in a 100 second analysis. Most of these moves are repetitions generated from a slightly different board position.

PROGRAM FUNCTION FOR EACH VALUE OF .STATE

STATE	SET BY	FUNCTION
C	GO	Generate all available moves from the current position and analyse as a benchmark with which to compare the real moves, which are generated by STATE 4.
4	GO	Generate all available moves, evaluating each one and assigning a value to it as a possible selection.
8	JANUS	Having made one trial move, generate the possible second moves for analysis.
0	JANUS	Having made one trial move, generate the possible replies for analysis.
FF	JANUS	Since a reply move was a capture, reverse the board and evaluate the exchange that could result.
FE	JANUS	Stage two of the exchange evaluation started by STATE FF.
FD	JANUS	Stage three of the exchange evaluation.
FC	JANUS	Last stage of the exchange evaluation.
F9	CHKCHK	Look for a capture of the king which signifies that the move being calculated is illegal.

STRATEGY OPERATION

After each real available move is generated and the various counts have been performed, the following information is available for decision making purposes.

MOB — Mobility. The total number of moves available for a given side from a given position. Each queen move is counted as two moves.

MAXC — Maximum Capture. The number of points to be gained by capturing the most valuable piece currently under attack.

CC — Capture Count. The total points of all opposing pieces under attack.

MAXP — Maximum Capturable Piece. Identification of the opponent's piece under attack which is worth the most points.

PRIOR COUNTS (.PMOB, .PMAXC, .PCC, .PMAXP) reflect the status of the position as it exists for the computer before any move is made. This is a benchmark, against which further moves are to be compared.

CONTINUATION COUNTS (.WMOB, .WMAXC, .WCC, .WMAXP) are obtained for each move tested to determine the potential of the new position that would result if the move were made.

REPLY COUNTS (.BMOB, .BMAXC, .BCC, .BMAXP) are obtained for each move tested to determine the potential danger of the opponent's available replies.

EXCHANGE COUNTS (.WCAP0, .WCAP1, .WCAP2, .BCAP0, .BCAP1, .BCAP2) are used to analyse the effect of the potential exchange combinations. Each count reflects the maximum number of points capturable at each level of an exchange combination. Capture chains are halted by pawn captures, king captures, or by reaching a limit of three captures per side.

In addition, information regarding the moving piece and its TO and FROM squares can also be used by the STRATGY algorithm.

All information available is combined by the algorithm in the subprogram STRATGY to calculate a single strategic value for the move under analysis. The algorithm, a weighted sum of the count information, is shown below:

```
VALUE =  + 4.00 * WCAP0
         + 1.25 * WCAP1
         + 0.75 * (WMAXC + WCC)
         + 0.25 * (WMOB  + WCAP2)
         - 2.50 * BMAXC
         - 2.00 * BCC
         - 1.25 * BCAP1
         - 0.50 * BMAXC
         - 0.25 * (PMAXC + PCC + PMOB + BCAP0 + BCAP2 + BMOB)
```

VALUE = VALUE + 02, A position bonus if the move is to the centre or out of the back rank.

VALUE = 00, If the move is illegal because the king is in check.

VALUE = FF, If the move results in a checkmate.

The move with the highest value is selected by the computer as the best move available. This algorithm can easily be modified by changing the weights assigned to the various parameters. For example, the program can be made to play more aggressively by increasing the importance of BMAXC and WCAPO in the equation above. On the other hand, it can be made to play more defensively by increasing the importance of BMAXC in the equation.

Note that the algorithm above has not yet been optimized. Therefore, it may be possible to significantly improve the play of the program by empirical testing to optimize the form and weights used for the equation.

An alternative form of algorithm to the weighted average type above, which also works well, assigns a fixed number of points to the occurrence of certain conditions. For example, the condition WMOB > PMOB may be considered to be worth 3 points regardless of the difference in value between the two variables. Similarly, conditions which are unfavourable would be assigned negative points. This type of strategy can be easily implemented by keeping a running total of the value in the accumulator and using CPX and CPY instructions to control branches around the adddition and subtraction routines. In general, more memory is required to implement an equally complex strategy using this type of algorithm, but in the long run this strategy will be more flexible.

OPENING PLAY

The MICROCHESS program is designed in such a way that the opening can be played from memory, following established lines of play for up to nine moves per side. In order to conserve memory, only one opening is actually stored in the computer at a given time. The opening is stored in locations 00C0 through 00DB. By storing each of the openings provided on cassette tape with a different ID for each, it is possible to load the desired opening before beginning play. More openings can be added to the repertoire by coding them in the format shown below.

Users with expanded memory can set up all the openings in a set of tables, allowing the program to select the appropriate opening as long as its opponent is following a standard procedure.

The ability to load an opening by name and play it with the computer also provides an excellent method of rehearsing openings for a chessplayer who is attempting to memorize the standard plays.

Each move and expected reply is stored in 3 bytes. The program first checks that the expected reply TO square is the same as the one in the stored opening. If it matches, the piece and the TO square for the computer's move are loaded into the display and moved. For example, the following illustrates the GIUOCO PIANO Opening. The computer is playing white.

Address	Data	Move
00DB	CC	Expected display when computer is making its first move.
00DA	0F	King pawn.
00D9	33	To KP4.
00D8	43	Expected reply P-KP4.
00D7	06	Knight.
00D6	22	To KB3.
00D5	52	Expected reply: N-QB3.
00D4	04	Bishop.

The last line of the opening sequence must be 99, or any impossible position square, to cause the program to leave the opening routine and enter the normal strategy evaluation routines.

MODIFYING THE INPUT AND OUTPUT ROUTINES

In order to use the MICROCHESS program on 6502 microprocessor systems other than the KIM-1, the only modifications necessary are changes to the input and output subroutine calls. These subroutines appear in the program listing as *OUT and *GETKEY at locations 0008, 000B, and 039F.

*OUT is a subroutine in the KIM ROM at location 1F1F which displays, in hexadecimal format, the contents of memory locations 00FB, 00FA, and 00F9 on the 6 digit LED display. 00FB contains the coded piece identification and locations 00FA and 00F9 contain the FROM and TO squares respectively. These three locations are also used to display CCCCCC and EEEEEE as verification of the keyboard input. At address 039F, *OUT is called by CKMATE at the end of the move analysis to flash the display. This call is not necessary for operation of the program and may be eliminated by replacing the JMP instruction at that location with an RTS (60). The MICROCHESS program has been designed so that neither the X and Y registers, nor the accumulator contents need be preserved by a replacement output subroutine.

*GETKEY is a KIM subroutine which returns the value of the depressed key in the accumulator. Hexadecimal values are returned right justified (e.g. 0A). The only non-hex key used is [PC] which returns the value 14. This key is used only once, at location 0033, so is easy to replace with any other value. Once again, the X and Y registers need not be preserved by a replacement input subroutine.

EXPANDED INPUT AND OUTPUT ROUTINES

Users with CRT or teletype terminals and additional memory will probably want to customize the input and output features of the program.

A format which can be used for move entry and move display is shown by the example: N(KN1) - KB3. This format completely expresses the move, and also provides a check value in the piece descriptor. Translation from this notation to the internal octal FROM and TO square notation is easily accomplished with a simple table lookup program which contains the file descriptors and subtracts 01 from the rank value.

The board can be displayed by providing a routine which prints a layout such as the one illustrated below. Before printing each square, the program could search the piece tables to determine if the square is occupied, and by which piece. The table descriptor is then obtained from the same tables used by the I/O routines above. Users with graphic terminals will want to set up even more elaborate board display routines.

WR	WN	WB	WK	WQ	WB		WR
WP	WP	WP		WP	WP	WP	WP
	**		**		WN		**

SPECIAL MOVES

Several types of moves are not included in the basic MICROCHESS program in order reduce the memory requirements. These moves, castling, en passant capture, and queening of pawns, can be added by expanding and modifying some of the subroutines which generate and execute moves. GNM must be modified to spot the occurrence of situations in which the moves are available. The actual move calculations must be added to CMOVE, and a flag to indicate the nature of the move set to allow MOVE and UMOVE to properly interpret them. The flag could use the two spare bits in .SQUARE. Additional parameters would be required to indicate when castling, or en passant moves are legal during the game, because these moves depend upon previous play for their legality. Expansion of the piece and point tables would allow the program to keep track of more than one queen per side.

STRATEGY IMPROVEMENTS

As you will soon discover when playing against the MICROCHESS program, it has a tendency to make ridiculous moves from time to time. These moves usually result from unusual positions, which point out deficiencies in the way the move value is calculated. A major problem in the analysis is that there is only one strategy which is used for the opening, the middle game, and the end game. This involves a considerable compromise of three different types of play. Users with memory expansion may wish to write three algorithms which can be switched in and out of the analysis at various points during the game.

Similarly, allowing more than 1K of memory enables the user to add more specialized evaluation routines. For example, a separate subroutine could be used to evaluate each of the following situations from both an offensive and defensive viewpoint, enabling a much more sophisticated level of play: 1- King in check. A major flaw in the current program causes the computer to minimize attacks by placing the opponent's king in check, even at the expense of a minor piece- a very short term solution to the problem! 2- En prise capture availability for either side. 3- Pawn development value: isolated pawns, passed pawns, doubled pawns, etc. 4- Xray analysis: the value of pins, discovered attack threats, etc. 5- Mating strategies: each of the major types of mates. 6- Positional development: utilization of open files, control of the centre, king position, pawn chains, etc.

With the exception of the capture tree, the MICROCHESS program analyses in full only one move for each side beyond the move it will make. It is possible to use the same recursive technique used by TREE to carry out a full analysis to a further depth. To do this would require a routine to analyse and evaluate each intermediate position arrived at. Sequences of possible positions with positive values for computer moves and negative values for opponent's moves can be summed to give the total long term value of each currently available move. In order to be time efficient, this analysis can be performed on a subset of the available continuations selected by a quick static analysis. In addition, a system of 'tree pruning' should be implemented to prevent long excursions down low valued branches. Programmers embarking on this type of program should bear in mind that from an average position with 50 available moves per side, a total of 15.625 billion sequences are generated in three moves per side.

As can be seen, MICROCHESS is only the beginning. However, it does demonstrate the capability of a small scale hobbyist microcomputer system to tackle the game of chess. It is hoped that this program will provide an inspiration and a stepping stone that chess playing programmers will expand and build upon. Let us know what you have done to improve the system. We will attempt to publish or distribute some of your ideas. It is hoped that a tournament of chess playing microcomputers can be arranged at a future microcomputer gathering. Expanded and modified versions of MICROCHESS will then have the opportunity to prove their playing ability against other programs in the same memory utilization class.

DATA FOR OPENINGS

The data below enables the computer to play the opening specified from memory. The data is in a block from 00C0 to 00DB. W specifies that the computer will play white, B specifies that the computer is black.

ADDR	W	FRENCH DEFENCE	B	W	GIUOCO PIANO	B	W	RUY LOPEZ	B	W	QUEEN'S INDIAN	B	W	FOUR KNIGHTS	B
DB	CC		44	CC		44	CC		44	CC		43	CC		44
DA	0F	P-K4	0F	0F	P-K4	0F	0F	P-K4	0F	0E	P-Q4	06	0F	P-K4	0F
D9	33	P-K3	24	33	P-K4	34	33	P-K4	34	34	N-KB3	25	33	P-K4	34
D8	53		43	43		55	43		55	52		42	43		55
D7	0E	P-Q4	0E	06	N-KB3	07	06	N-KB3	07	0D	P-QB4	0F	06	N-KB3	07
D6	34	P-Q4	33	22	N-QB3	22	22	N-QB3	22	35	P-K3	24	22	N-QB3	22
D5	44		52	55		42	55		31	53		55	55		52
D4	07	N-QB3	06	04	B-B4	04	04	B-N5	06	06	N-KB3	0B	07	N-B3	06
D3	25	N-KB3	25	35	B-B4	32	46	N-B3	25	22	P-QN3	21	25	N-B3	25
D2	52		36	45		52	52		75	56		56	52		31
D1	05	B-N5	04	0D	P-B3	06	00	O-O	06	0A	P-KN3	05	04	B-N5	04
D0	41	B-K2	14	25	N-B3	25	01	NxP	44	21	B-N2	11	46	B-N5	41
CF	63		34	52		43	33		43	66		66	36		75
CE	0F	P-K5	06	0E	P-Q4	0F	0E	P-Q4	04	04	B-N2	04	00	O-O	00
CD	43	KN-Q2	13	34	PxP	43	34	B-K2	14	11	B-K2	14	01	O-O	06
CC	64		14	34		43	63		64	63		75	72		53
CB	05	BxB	01	0D	PxP	04	01	Q-K2	06	00	O-O	00	0E	P-Q3	0E
CA	63	QxB	14	34	B-N5	41	13	N-Q3	23	01	O-O	06	24	P-Q3	23
C9	63		63	36		52	54		22	72		52	54		36
C8	01	Q-Q2	00	07	N-B3	06	04	BxN	0B	07	N-B3	06	05	B-N5	04
C7	14	O-O	06	25	NxKP	44	55	NPxB	22	25	N-K5	44	41	BxN	52
C6	72		45	33		75	55		34	33		62	25		52
C5	0C	P-B4	0D	00	O-O	06	0E	PxP	06	01	Q-B2	06	0B	PxB	01
C4	32	P-QB4	32	01	NxN	52	43	N-N2	11	15	NxN	52	25	Q-K2	14
C3	45		55	25		52	66		52	25		52	63		74
C2	06	N-B3	07	0B	PxN	04	07	N-B3	00	01	QxN	0C	02	R-K1	07
C1	22	N-QB3	22	25	BxP	52	25	O-O	06	25	P-KB4	35	03	N-Q1	03
C0	99		99	99		99	99		99	99		99	99		99

EXPLANATION OF SYMBOLS

ADDR	SYMBOL	EXPLANATION
0050	.BOARD	: LOCATION OF PIECES
0060	.BK	: OPPONENT'S PIECES
0070	.SETW	: INITIAL PIECE LOCATIONS
008F	.MOVEX	: TABLE OF MOVE DIRECTIONS
00A0	.POINTS	: TABLE OF PIECE VALUES
00B0	.PIECE	: CURRENT PIECE UNDER ANALYSIS
00B1	.SQUARE	: TO SQUARE OF PIECE
00B2	.SP2	: STACK POINTER FOR STACK 2
00B3	.SP1	: STACK POINTER FOR STACK 1
00B4	.INCHEK	: MOVE INTO CHECK FLAG
00B5	.STATE	: STATE OF ANALYSIS
00B6	.MOVEN	: MOVE TABLE POINTER
00DC	.OMOVE	: OPENING POINTER
00DC	.OPNING	: OPENING MOVE TABLE
00DD	.WCAP0	: COMPUTER CAPTURE 0
00DE	.COUNT	: START OF COUNT TABLE
00DE	.BCAP2	: OPPONENT CAPTURE 2
00DF	.WCAP2	: COMPUTER CAPTURE 2
00E0	.BCAP1	: OPPONENT CAPTURE 1
00E1	.WCAP1	: COMPUTER CAPTURE 1
00E2	.BCAP0	: OPPONENT CAPTURE 0
00E3	.MOB	: MOBILITY
00E4	.MAXC	: MAXIMUM CAPTURE
00E5	.CC	: CAPTURE COUNT
00E6	.PCAP	: PIECE ID OF MAXC
00E3	.BMOB	: OPPONENT MOBILITY
00E4	.BMAXC	: OPPONENT MAXIMUM CAPTURE
00E5	.BCC	: OPPONENT CAPTURE COUNT
00E6	.BMAXP	: OPPONENT MAXP
00E8	.XMAXC	: CURRENT MAXIMUM CAPTURE
00EB	.WMOB	: COMPUTER MOBILITY
00EC	.WMAXC	: COMPUTER MAXIMUM CAPTURE
00ED	.WCC	: COMPUTER CAPTURE COUNT
00EE	.WMAXP	: COMPUTER MAXP
00EF	.PMOB	: PREVIOUS COMPUTER MOB
00F0	.PMAXC	: PREVIOUS COMPUTER MAXC
00F1	.PCC	: PREVIOUS COMPUTER CC
00F2	.PCP	: PREVIOUS COMPUTER MAXP
00F3	.OLDKY	: KEY INPUT TEMPORARY
00FB	.BESTP	: PIECE OF BEST MOVE FOUND
00FA	.BESTV	: VALUE OF BEST MOVE FOUND
00F9	.BESTM	: TO SQUARE OF BEST MOVE
00FB	.DIS1	: DISPLAY POINT 1
00FA	.DIS2	: DISPLAY POINT 2
00F9	.DIS3	: DISPLAY POINT 3

```
      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

0000: D8 A2 FF 9A A2 C8 86 B2 20 1F 1F 20 6A 1F C5 F3
0010: F0 F6 85 F3 C9 0C D0 0F A2 1F B5 70 95 50 CA 10
0020: F9 86 DC A9 CC D0 12 C9 0E D0 07 20 B2 02 A9 EE
0030: D0 07 C9 14 D0 0B 20 A2 03 85 FB 85 FA 85 F9 D0
0040: BF C9 0F D0 06 20 4B 03 4C 9D 01 4C 96 01 10 00

0070: 03 04 00 07 02 05 01 06 10 17 11 16 12 15 14 13
0080: 73 74 70 77 72 75 71 76 60 67 61 66 62 65 64 63
0090: F0 FF 01 10 11 0F EF F1 DF E1 EE F2 12 0E 1F 21
00A0: 0B 0A 06 06 04 04 04 04 02 02 02 02 02 02 02 02

0100: A6 B5 30 5C A5 B0 F0 08 E0 08 D0 04 C5 E6 F0 2E
0110: F6 E3 C9 01 D0 02 F6 E3 50 1E A0 0F A5 B1 D9 60
0120: 00 F0 03 88 10 F8 B9 A0 00 D5 E4 90 04 94 E6 95
0130: E4 18 08 75 E5 95 E5 28 E0 04 F0 03 30 31 60 A5
0140: E8 85 DD A9 00 85 B5 20 4B 03 20 B2 02 20 00 02
0150: 20 B2 02 A9 08 85 B5 20 09 02 20 31 03 4C 80 17
0160: E0 F9 D0 0B A5 60 C5 B1 D0 04 A9 00 85 B4 60 50
0170: FD A0 07 A5 B1 D9 60 00 F0 05 88 F0 F1 10 F6 B9
0180: A0 00 D5 E2 90 02 95 E2 C6 B5 A9 FB C5 B5 F0 03
0190: 20 25 03 E6 B5 60 C9 08 B0 12 20 EA 03 A2 1F B5
01A0: 50 C5 FA F0 03 CA 10 F7 86 FB 86 B0 4C 00 00 C0

0200: A2 10 A9 00 95 DE CA 10 FB A9 10 85 B0 C6 B0 10
0210: 01 60 20 1E 03 A4 B0 A2 08 86 B6 C0 08 10 41 C0
0220: 06 10 2E C0 04 10 1F C0 01 F0 09 10 0E 20 8E 02
0230: D0 FB F0 D9 20 9C 02 D0 FB F0 D2 A2 04 86 B6 20
0240: 9C 02 D0 FB F0 C7 20 9C 02 A5 B6 C9 04 D0 F7 F0
0250: BC A2 10 86 B6 20 8E 02 A5 B6 C9 08 D0 F7 F0 AD
0260: A2 06 86 B6 20 CA 02 50 05 30 03 20 00 01 20 1E
0270: 03 C6 B6 A5 B6 C9 05 F0 EB 20 CA 02 70 8F 30 8D
0280: 20 00 01 A5 B1 29 F0 C9 20 F0 EE 4C 0D 02 20 CA
0290: 02 30 03 20 00 01 20 1E 03 C6 B6 60 20 CA 02 90
02A0: 02 50 F9 30 07 08 20 00 01 28 50 F0 20 1E 03 C6
02B0: B6 60 A2 0F 38 B4 60 A9 77 F5 50 95 60 94 50 38
02C0: A9 77 F5 50 95 50 CA 10 EB 60 A5 B1 A6 B6 18 75
02D0: 8F 85 B1 29 88 D0 42 A5 B1 A2 20 CA 30 0E D5 50
02E0: D0 F9 E0 10 30 33 A9 7F 69 01 70 01 B8 A5 B5 30
02F0: 24 C9 08 10 20 48 08 A9 F9 85 B5 85 B4 20 4B 03
0300: 20 B2 02 20 09 02 20 2E 03 28 68 85 B5 A5 B4 30
0310: 04 38 A9 FF 60 18 A9 00 60 A9 FF 18 B8 60 A6 B0
0320: B5 50 85 B1 60 20 4B 03 20 B2 02 20 09 02 20 B2
0330: 02 BA 86 B3 A6 B2 9A 68 85 B6 68 85 B0 AA 68 95
0340: 50 68 AA 68 85 B1 95 50 4C 70 03 BA 86 B3 A6 B2
0350: 9A A5 B1 48 A8 A2 1F D5 50 F0 03 CA 10 F9 A9 CC
0360: 95 50 8A 48 A6 B0 B5 50 94 50 48 8A 48 A5 B6 48
0370: BA 86 B2 A6 B3 9A 60 A6 E4 E4 A0 D0 04 A9 00 F0
0380: 0A A6 E3 D0 06 A6 EE D0 02 A9 FF A2 04 86 B5 C5
0390: FA 90 0C F0 0A 85 FA A5 B0 85 FB A5 B1 85 F9 4C
03A0: 1F 1F A6 DC 10 17 A5 F9 D5 DC D0 0F CA B5 DC 85
03B0: FB CA B5 DC 85 F9 CA 86 DC D0 1A 85 DC A2 0C 86
03C0: B5 86 FA A2 14 20 02 02 A2 04 86 B5 20 00 02 A6
03D0: FA E0 0F 90 12 A6 FB B5 50 85 FA 86 B0 A5 F9 85
03E0: B1 20 4B 03 4C 00 00 A9 FF 60 A2 04 06 F9 26 FA
03F0: CA D0 F9 05 F9 85 F9 85 B1 60 00 00 00 00 00 00

1780: 18 A5 80 65 EB 65 EC 65 ED 65 E1 65 DF 38 E5 F0
1790: E5 F1 E5 E2 E5 E0 E5 DE E5 EF E5 E3 B0 02 A9 00
17A0: 4A 18 69 40 65 EC 65 ED 38 E5 E4 4A 18 69 90 65
17B0: DD 65 DD 65 DD 65 DD 65 E1 38 E5 E4 E5 E4 E5 E5
17C0: E5 E5 E5 E0 A6 B1 E0 33 F0 16 E0 34 F0 12 E0 22
17D0: F0 0E E0 25 F0 0A A6 B0 F0 09 B4 50 C0 10 10 03
17E0: 18 69 02 4C 77 03
```

MICROCHESS

MICROCHESS


```
 2                      ;       EXECUTION BEGINS AT ADDRESS 0000
 3                      ;
 4                              +++
 5  0000 D8             CHESS   CLD                     INITIALIZE
 6  0001 A2 FF                  LDXIM   FF              TWO STACKS
 7  0003 9A                     TXS
 8  0004 A2 C8                  LDXIM   C8
 9  0006 86 B2                  STXZ    .SP2
10                      ;
11                      ;       ROUTINES TO LIGHT LED
12                      ;       DISPLAY AND GET KEY
13                      ;       FROM KEYBOARD.
14                      ;
15  0008 20 1F 1F       OUT     JSR     *OUT            DISPLAY AND
16  000B 20 6A 1F               JSR     *GETKEY         GET INPUT
17  000E C5 F3                  CMPZ    .OLDKY          KEY IN ACC
18  0010 F0 F6                  BEQ     OUT             (DEBOUNCE)
19  0012 85 F3                  STAZ    .OLDKY
20                      ;
21  0014 C9 0C                  CMPIM   0C              [C]
22  0016 D0 0F                  BNE     NOSET           SET UP
23  0018 A2 1F                  LDXIM   1F              BOARD
24  001A B5 70          WHSET   LDAZX   .SETW           FROM
25  001C 95 50                  STAZX   .BOARD          SETW
26  001E CA                     DEX
27  001F 10 F9                  BPL     WHSET
28  0021 86 DC                  STXZ    .OMOVE
29  0023 A9 CC                  LDAIM   CC
30  0025 D0 12                  BNE     CLDSP
31                      ;
32  0027 C9 0E          NOSET   CMPIM   0E              [E]
33  0029 D0 07                  BNE     NOREV           REVERSE
34  002B 20 B2 02               JSR     REVERSE         BOARD AS
35  002E A9 EE                  LDAIM   EE                 IS
36  0030 D0 07                  BNE     CLDSP
37                      ;
38  0032 C9 14          NOREV   CMPIM   14              [PC]
39  0034 D0 0B                  BNE     NOGO            PLAY CHESS
40  0036 20 A2 03               JSR     GO
41                      ;
42  0039 85 FB          CLDSP   STA     .DIS1           DISPLAY
43  003B 85 FA                  STAZ    .DIS2           ACROSS
44  003D 85 F9                  STAZ    .DIS3           DISPLAY
45  003F D0 BF                  BNE     CHESS
46                      ;
47  0041 C9 0F          NOGO    CMPIM   0F              [F]
48  0043 D0 06                  BNE     NOMV            MOVE MAN
49  0045 20 4B 03               JSR     MOVE            AS ENTERED
50  0048 4C 9D 01               JMP     DISP
```

```
 51  004B 4C 96 01      NOMV      JMP       INPUT
 52                     ;
 53                     ;         THE ROUTINE JANUS DIRECTS THE
 54                     ;         ANALYSIS BY DETERMINING WHAT
 55                     ;         SHOULD OCCUR AFTER EACH MOVE
 56                     ;         GENERATED BY GNM
 57                     ;
 58                     ;
 59                               +++
 60  0100 A6 B5         JANUS     LDXZ      .STATE
 61  0102 30 5C                   BMI       NOCOUNT
 62                     ;
 63                     ;         THIS ROUTINE COUNTS OCCURRENCES
 64                     ;         IT DEPENDS UPON STATE TO INDEX
 65                     ;         THE CORRECT COUNTERS
 66                     ;
 67  0104 A5 B0         COUNTS    LDAZ      .PIECE
 68  0106 F0 08                   BEQ       OVER           IF STATE=8
 69  0108 E0 08                   CPXIM     08             DO NOT COUNT
 70  010A D0 04                   BNE       OVER           BLK MAX CAP
 71  010C C5 E6                   CMPZ      .BMAXP         MOVES FOR
 72  010E F0 2E                   BEQ       XRT            WHITE
 73                     ;
 74  0110 F6 E3         OVER      INCZX     .MOB           MOBILITY
 75  0112 C9 01                   CMPIM     01             + QUEEN
 76  0114 D0 02                   BNE       NOQ            FOR TWO
 77  0116 F6 E3                   INCZX     .MOB
 78                     ;
 79  0118 50 1E         NOQ       BVC       NOCAP
 80  011A A0 0F                   LDYIM     0F             CALCULATE
 81  011C A5 B1                   LDAZ      .SQUARE        POINTS
 82  011E D9 60 00      ELOOP     CMPAY     .BK            CAPTURED
 83  0121 F0 03                   BEQ       FOUN           BY THIS
 84  0123 88                      DEY                      MOVE
 85  0124 10 F8                   BPL       ELOOP
 86  0126 B9 A0 00      FOUN      LDAAY     .POINTS
 87  0129 D5 E4                   CMPZX     .MAXC
 88  012B 90 04                   BCC       LESS           SAVE IF
 89  012D 94 E6                   STYZX     .PCAP          BEST THIS
 90  012F 95 E4                   STAZX     .MAXC          STATE
 91                     ;
 92  0131 18            LESS      CLC
 93  0132 08                      PHP                      ADD TO
 94  0133 75 E5                   ADCZX     .CC            CAPTURE
 95  0135 95 E5                   STAZX     .CC            COUNTS
 96  0137 28                      PLP
 97                     ;
 98  0138 E0 04         NOCAP     CPXIM     04
 99  013A F0 03                   BEQ       ON4
100  013C 30 31                   BMI       TREE           (=00 ONLY)
```

```
                                XRT     RTS
                                ;
                                ;       GENERATE FURTHER MOVES FOR COUNT
                                ;       AND ANALYSIS
                                ;
013F A5 E8                      ON4       LDAZ    .XMAXC          SAVE ACTUAL
0141 85 DD                                STAZ    .WCAPO           CAPTURE
0143 A9 00                                LDAIM   00              STATE=0
0145 85 B5                                STAZ    .STATE
0147 20 4B 03                             JSR     MOVE            GENERATE
014A 20 B2 02                             JSR     REVERSE         IMMEDIATE
014D 20 00 02                             JSR     GNMZ            REPLY MOVES
0150 20 B2 02                             JSR     REVERSE
                                ;
0153 A9 08                                LDAIM   08              STATE=8
0155 85 B5                                STAZ    .STATE          GENERATE
0157 20 09 02                             JSR     GNM             CONTINUATION
015A 20 31 03                             JSR     UMOVE           MOVES
                                ;
015D 4C 80 17                             JMP     STRATGY         FINAL EVALUATION
0160 E0 F9                      NOCOUNT   CPXIM   F9
0162 D0 0B                                BNE     TREE
                                ;
                                ;       DETERMINE IF THE KING CAN BE
                                ;       TAKEN, USED BY CHKCHK
                                ;
0164 A5 60                                LDAZ    .BK             IS KING
0166 C5 B1                                CMPZ    .SQUARE         IN CHECK?
0168 D0 04                                BNE     RETJ            SET INCHEK=0
016A A9 00                                LDAIM   00              IF IT IS
016C 85 B4                                STAZ    .INCHEK
016E 60                         RETJ      RTS
                                ;
                                ;       IF A PIECE HAS BEEN CAPTURED BY
                                ;       A TRIAL MOVE, GENERATE REPLIES &
                                ;       EVALUATE THE EXCHANGE GAIN/LOSS
                                ;
016F 50 FD                      TREE      BVC     RETJ            NO CAP
0171 A0 07                                LDYIM   07              (PIECES)
0173 A5 B1                                LDAZ    .SQUARE
0175 D9 60 00                   LOOPX     CMPAY   .BK
0178 F0 05                                BEQ     FOUNX
017A 88                                   DEY
017B F0 F1                                BEQ     RETJ            (KING)
017D 10 F6                                BPL     LOOPX           SAVE
017F B9 A0 00                   FOUNX     LDAAY   .POINTS         BEST CAP
0182 D5 E2                                CMPZX   .BCAPO          AT THIS
0184 90 02                                BCC     NOMAX           LEVEL
0186 95 E2                                STAZX   .BCAPO
0188 C6 B5                      NOMAX     DEC     .STATE
```

```
                                        LDAIM   FB              IF STATE=FB
                                        CMPZ    .STATE          TIME TO TURN
                                        BEQ     UPTREE          AROUND
                                        JSR     GENRM           GENERATE FURTHER
                            UPTREE      INC     .STATE          CAPTURES
                                        RTS
                            ;
                            ;       THE PLAYER'S MOVE IS INPUT
                            ;
                            INPUT       CMPIM   08              NOT A LEGAL
                                        BCS     ERROR           SQUARE #
                                        JSR     DISMV
                            DISP        LDXIM   1F
                            SEARCH      LDAZX   .BOARD
                                        CMPZ    .DIS2
                                        BEQ     HERE            DISPLAY
                                        DEX                     PIECE AT
                                        BPL     SEARCH          FROM
                            HERE        STXZ    .DIS1           SQUARE
                                        STXZ    .PIECE
                            ERROR       JMP     CHESS
                            ;
                            ;       GENERATE ALL MOVES FOR ONE
                            ;       SIDE, CALL JANUS AFTER EACH
                            ;       ONE FOR NEXT STEP
                            ;
                                        +++
                            GNMZ        LDXIM   10              CLEAR
                            GNMX        LDAIM   00              COUNTERS
                            CLEAR       STAZX   .COUNT
                                        DEX
                                        BPL     CLEAR
                            ;
                            GNM         LDAIM   10              SET UP
                                        STAZ    .PIECE          PIECE
                            NEWP        DECZ    .PIECE          NEW PIECE
                                        BPL     NEX             ALL DONE?
                                        RTS                       -YES
                            ;
                            NEX         JSR     RESET         READY
                                        LDYZ    .PIECE          GET PIECE
                                        LDXIM   08
                                        STXZ    .MOVEN          COMMON START
                                        CPYIM   08              WHAT IS IT?
                                        BPL     PAWN            PAWN
                                        CPYIM   06
                                        BPL     KNIGHT          KNIGHT
                                        CPYIM   04
                                        BPL     BISHOP          BISHOP
                                        CPYIM   01
```

```
0229 F0 09                BEQ     QUEEN       QUEEN
022B 10 0E                BPL     ROOK        ROOK
                ;
022D 20 8E 02   KING      JSR     SNGMV       MUST BE KING!
0230 D0 FB                BNE     KING        MOVES
0232 F0 D9                BEQ     NEWP        8 TO 1
0234 20 9C 02   QUEEN     JSR     LINE
0237 D0 FB                BNE     QUEEN       MOVES
0239 F0 D2                BEQ     NEWP        8 TO 1
                ;
023B A2 04      ROOK      LDXIM   04
023D 86 B6                STXZ    .MOVEN      MOVES
023F 20 9C 02   AGNR      JSR     LINE        4 TO 1
0242 D0 FB                BNE     AGNR
0244 F0 C7                BEQ     NEWP
                ;
0246 20 9C 02   BISHOP    JSR     LINE
0249 A5 B6                LDAZ    .MOVEN      MOVES
024B C9 04                CMPIM   04          8 TO 5
024D D0 F7                BNE     BISHOP
024F F0 BC                BEQ     NEWP
                ;
0251 A2 10      KNIGHT    LDXIM   10
0253 86 B6                STXZ    .MOVEN      MOVES
0255 20 8E 02   AGNN      JSR     SNGMV       16 TO 9
0258 A5 B6                LDAZ    .MOVEN
025A C9 08                CMPIM   08
025C D0 F7                BNE     AGNN
025E F0 AD                BEQ     NEWP
                ;
0260 A2 06      PAWN      LDXIM   06
0262 86 B6                STXZ    .MOVEN
0264 20 CA 02   P1        JSR     CMOVE       RIGHT CAP?
0267 50 05                BVC     P2
0269 30 03                BMI     P2
026B 20 00 01             JSR     JANUS       YES
026E 20 1E 03   P2        JSR     RESET
0271 C6 B6                DECZ    .MOVEN       LEFT CAP?
0273 A5 B6                LDAZ    .MOVEN
0275 C9 05                CMPIM   05
0277 F0 EB                BEQ     P1
0279 20 CA 02   P3        JSR     CMOVE       AHEAD
027C 70 8F                BVS     NEWP         ILLEGAL
027E 30 8D                BMI     NEWP
0280 20 00 01             JSR     JANUS
0283 A5 B1                LDAZ    .SQUARE      GETS TO
0285 29 F0                ANDIM   F0           3RD RANK?
0287 C9 20                CMPIM   20
0289 F0 EE                BEQ     P3           DO DOUBLE
028B 4C 0D 02             JMP     NEWP
```

```
                        ;
                        ;       CALCULATE SINGLE STEP MOVES
                        ;       FOR K, N
                        ;
028E 20 CA 02   SNGMV   JSR     CMOVE           CALC MOVE
0291 30 03              BMI     ILL1             -IF LEGAL
0293 20 00 01           JSR     JANUS            -EVALUATE
0296 20 1E 03   ILL1    JSR     RESET
0299 C6 B6              DECZ    .MOVEN
029B 60                 RTS
                        ;
                        ;       CALCULATE ALL MOVES DOWN A
                        ;       STRAIGHT LINE FOR Q,B,R
                        ;
029C 20 CA 02   LINE    JSR     CMOVE           CALC MOVE
029F 90 02              BCC     OVL              NO CHK
02A1 50 F9              BVC     LINE             CH,NOCAP
02A3 30 07      OVL     BMI     ILL              RETURN
02A5 08                 PHP
02A6 20 00 01           JSR     JANUS           EVALUATE POSN
02A9 28                 PLP
02AA 50 F0              BVC     LINE            NOT A CAP
02AC 20 1E 03   ILL     JSR     RESET           LINE STOPPED
02AF C6 B6              DECZ    .MOVEN           NEXT DIR
02B1 60                 RTS
                        ;
                        ;       EXCHANGE SIDES FOR REPLY
                        ;       ANALYSIS
                        ;
02B2 A2 0F      REVERSE LDXIM   0F
02B4 38         ETC     SEC
02B5 B4 60              LDYZX   .BK             SUBTRACT
02B7 A9 77              LDAIM   77              POSITION
02B9 F5 50              SBCZX   .BOARD          FROM 77
02BB 95 60              STAZX   .BK
02BD 94 50              STYZX   .BOARD            AND
02BF 38                 SEC
02C0 A9 77              LDAIM   77              EXCHANGE
02C2 F5 50              SBCZX   .BOARD          PIECES
02C4 95 50              STAZX   .BOARD
02C6 CA                 DEX
02C7 10 EB              BPL     ETC
02C9 60                 RTS
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
```

```
                        ;       CMOVE CALCULATES THE TO SQUARE
                        ;       USING .SQUARE AND THE MOVE
                        ;     TABLE.  FLAGS SET AS FOLLOWS:
                        ;     N - ILLEGAL MOVE
                        ;     V - CAPTURE (LEGAL UNLESS IN CH)
                        ;     C - ILLEGAL BECAUSE OF CHECK
                        ;     [MY THANKS TO JIM BUTTERFIELD
                        ;      WHO WROTE THIS MORE EFFICIENT
                        ;      VERSION OF CMOVE]
                        ;
02CA A5 B1      CMOVE     LDAZ      .SQUARE       GET SQUARE
02CC A6 B6                LDXZ      .MOVEN        MOVE POINTER
02CE 18                   CLC
02CF 75 8F                ADCZX     .MOVEX        MOVE LIST
02D1 85 B1                STAZ      .SQUARE       NEW POS'N
02D3 29 88                ANDIM     88
02D5 D0 42                BNE       ILLEGAL       OFF BOARD
02D7 A5 B1                LDAZ      .SQUARE
                        ;
02D9 A2 20                LDXIM     20
02DB CA         LOOP      DEX                     IS TO
02DC 30 0E                BMI       NO            SQUARE
02DE D5 50                CMPZX     .BOARD        OCCUPIED?
02E0 D0 F9                BNE       LOOP
                        ;
02E2 E0 10                CPXIM     10            BY SELF?
02E4 30 33                BMI       ILLEGAL
                        ;
02E6 A9 7F                LDAIM     7F            MUST BE CAP!
02E8 69 01                ADCIM     01            SET V FLAG
02EA 70 01                BVS       SPX           (JMP)
                        ;
02EC B8         NO        CLV                     NO CAPTURE
                        ;
02ED A5 B5      SPX       LDAZ      .STATE        SHOULD WE
02EF 30 24                BMI       RETL          DO THE
02F1 C9 08                CMPIM     08            CHECK CHECK?
02F3 10 20                BPL       RETL
                        ;
                        ;       CHKCHK REVERSES SIDES
                        ;     AND LOOKS FOR A KING
                        ;     CAPTURE TO INDICATE
                        ;     ILLEGAL MOVE BECAUSE OF
                        ;     CHECK.  SINCE THIS IS
                        ;     TIME CONSUMING, IT IS NOT
                        ;     ALWAYS DONE.
                        ;
02F5 48         CHKCHK    PHA                     STATE
02F6 08                   PHP
02F7 A9 F9                LDAIM     F9
```

```
351  02F9 85 B5          STAZ    .STATE      GENERATE
352  02FB 85 B4          STAZ    .INCHEK     ALL REPLY
353  02FD 20 4B 03       JSR     MOVE        MOVES TO
354  0300 20 B2 02       JSR     REVERSE     SEE IF KING
355  0303 20 09 02       JSR     GNM         IS IN
356  0306 20 2E 03       JSR     RUM         CHECK
357  0309 28             PLP
358  030A 68             PLA
359  030B 85 B5          STAZ    .STATE
360  030D A5 B4          LDAZ    .INCHEK
361  030F 30 04          BMI     RETL        NO - SAFE
362  0311 38             SEC                 YES - IN CHK
363  0312 A9 FF          LDAIM   FF
364  0314 60             RTS
365                      ;
366  0315 18             RETL    CLC         LEGAL
367  0316 A9 00          LDAIM   00          RETURN
368  0318 60             RTS
369                      ;
370  0319 A9 FF          ILLEGAL LDAIM   FF
371  031B 18             CLC                 ILLEGAL
372  031C B8             CLV                 RETURN
373  031D 60             RTS
374                      ;
375                      ;       REPLACE .PIECE ON CORRECT .SQUARE
376                      ;
377  031E A6 B0          RESET   LDXZ    .PIECE      GET LOCAT.
378  0320 B5 50          LDAZX   .BOARD      FOR PIECE
379  0322 85 B1          STAZ    .SQUARE     FROM BOARD
380  0324 60             RTS
381                      ;
382                      ;
383                      ;
384  0325 20 4B 03       GENRM   JSR     MOVE        MAKE MOVE
385  0328 20 B2 02       GENR2   JSR     REVERSE     REVERSE BOARD
386  032B 20 09 02               JSR     GNM         GENERATE MOVES
387  032E 20 B2 02       RUM     JSR     REVERSE     REVERSE BACK
388                      ;
389                      ;       ROUTINE TO UNMAKE A MOVE MADE BY
390                      ;               MOVE
391                      ;
392  0331 BA             UMOVE   TSX                 UNMAKE MOVE
393  0332 86 B3          STXZ    .SP1
394  0334 A6 B2          LDXZ    .SP2        EXCHANGE
395  0336 9A             TXS                 STACKS
396  0337 68             PLA                 MOVEN
397  0338 85 B6          STAZ    .MOVEN
398  033A 68             PLA                 CAPTURED
399  033B 85 B0          STAZ    .PIECE      PIECE
400  033D AA             TAX
```

```
401  033E 68                    PLA                    FROM SQUARE
402  033F 95 50                 STAZX     .BOARD
403  0341 68                    PLA                    PIECE
404  0342 AA                    TAX
405  0343 68                    PLA                    TO SQUARE
406  0344 85 B1                 STAZ      .SQUARE
407  0346 95 50                 STAZX     .BOARD
408  0348 4C 70 03              JMP       STRV
409                       ;
410                       ;    THIS ROUTINE MOVES .PIECE
411                       ;    TO .SQUARE,  PARAMETERS
412                       ;    ARE SAVED IN A STACK TO UNMAKE
413                       ;    THE MOVE LATER
414                       ;
415  034B BA              MOVE  TSX
416  034C 86 B3                 STXZ      .SP1           SWITCH
417  034E A6 B2                 LDXZ      .SP2           STACKS
418  0350 9A                    TXS
419  0351 A5 B1                 LDAZ      .SQUARE
420  0353 48                    PHA                    TO SQUARE
421  0354 A8                    TAY
422  0355 A2 1F                 LDXIM     1F
423  0357 D5 50           CHECK CMPZX     .BOARD         CHECK FOR
424  0359 F0 03                 BEQ       TAKE           CAPTURE
425  035B CA                    DEX
426  035C 10 F9                 BPL       CHECK
427  035E A9 CC           TAKE  LDAIM     CC
428  0360 95 50                 STAZX     .BOARD
429  0362 8A                    TXA                    CAPTURED
430  0363 48                    PHA                    PIECE
431  0364 A6 B0                 LDXZ      .PIECE
432  0366 B5 50                 LDAZX     .BOARD
433  0368 94 50                 STYZX     .BOARD         FROM
434  036A 48                    PHA                        SQUARE
435  036B 8A                    TXA
436  036C 48                    PHA                    PIECE
437  036D A5 B6                 LDAZ      .MOVEN
438  036F 48                    PHA                    MOVEN
439  0370 BA              STRV  TSX
440  0371 86 B2                 STXZ      .SP2           SWITCH
441  0373 A6 B3                 LDXZ      .SP1           STACKS
442  0375 9A                    TXS                      BACK
443  0376 60                    RTS
444                       ;
445                       ;    CONTINUATION OF SUB STRATGY
446                       ;    -CHECKS FOR CHECK OR CHECKMATE
447                       ;    AND ASSIGNS VALUE TO MOVE
448                       ;
449  0377 A6 E4           CKMATE LDXZ     .BMAXC         CAN BLK CAP
450  0379 E4 A0                 CPXZ      .POINTS        MY KING?
```

```
                                BNE     NOCHEK
037B D0 04                      LDAIM   00              GULP!
037D A9 00                      BEQ     RETV            DUMB MOVE!
037F F0 0A
                        ;
0381 A6 E3              NOCHEK  LDXZ    .BMOB           IS BLACK
0383 D0 06                      BNE     RETV            UNABLE TO
0385 A6 EE                      LDXZ    .WMAXP          MOVE AND
0387 D0 02                      BNE     RETV            KING IN CH?
0389 A9 FF                      LDAIM   FF              YES! MATE
                        ;
038B A2 04              RETV    LDXIM   04              RESTORE
038D 86 B5                      STXZ    .STATE          STATE=4
                        ;
                        ;       THE VALUE OF THE MOVE (IN ACC)
                        ;       IS COMPARED TO THE BEST MOVE AND
                        ;       REPLACES IT IF IT IS BETTER
                        ;
038F C5 FA              PUSH    CMPZ    .BESTV          IS THIS BEST
0391 90 0C                      BCC     RETP            MOVE SO FAR?
0393 F0 0A                      BEQ     RETP
0395 85 FA                      STAZ    .BESTV          YES!
0397 A5 B0                      LDAZ    .PIECE          SAVE IT
0399 85 FB                      STAZ    .BESTP
039B A5 B1                      LDAZ    .SQUARE
039D 85 F9                      STAZ    .BESTM          FLASH DISPLAY
039F 4C 1F 1F           RETP    JMP     *OUT            AND RTS
                        ;
                        ;       MAIN PROGRAM TO PLAY CHESS
                        ;       PLAY FROM OPENING OR THINK
                        ;
03A2 A6 DC              GO      LDXZ    .OMOVE          OPENING?
03A4 10 17                      BPL     NOOPEN              -NO
03A6 A5 F9                      LDAZ    .DIS3           -YES WAS
03A8 D5 DC                      CMPZX   .OPNING          OPPONENT'S
03AA D0 0F                      BNE     END              MOVE OK?
03AC CA                         DEX
03AD B5 DC                      LDAZX   .OPNING         GET NEXT
03AF 85 FB                      STAZ    .DIS1           CANNED
03B1 CA                         DEX                     OPENING MOVE
03B2 B5 DC                      LDAZX   .OPNING
03B4 85 F9                      STAZ    .DIS3           DISPLAY IT
03B6 CA                         DEX
03B7 86 DC                      STXZ    .OMOVE          MOVE IT
03B9 D0 1A                      BNE     MV2             (JMP)
                        ;
03BB 85 DC              END     STAZ    .OMOVE          FLAG OPENING
03BD A2 0C              NOOPEN  LDXIM   0C              FINISHED
03BF 86 B5                      STXZ    .STATE          STATE=C
03C1 86 FA                      STXZ    .BESTV          CLEAR BESTV
03C3 A2 14                      LDXIM   14              GENERATE P
```

```
                                        JSR     GNMX            MOVES
                                ;
                                        LDXIM   04              STATE=4
                                        STXZ    .STATE          GENERATE AND
                                        JSR     GNMZ            TEST AVAILABLE
                                ;                               MOVES
                                ;
                                        LDXZ    .BESTV          GET BEST MOVE
                                        CPXIM   0F              IF NONE
                                        BCC     MATE            OH OH!
                                ;
                                MV2     LDXZ    .BESTP          MOVE
                                        LDAZX   .BOARD           THE
                                        STAZ    .BESTV          BEST
                                        STXZ    .PIECE          MOVE
                                        LDAZ    .BESTM
                                        STAZ    .SQUARE         AND DISPLAY
                                        JSR     MOVE             IT
                                        JMP     CHESS
                                ;
                                MATE    LDAIM   FF              RESIGN
                                        RTS                     OR STALEMATE
                                ;
                                ;       SUBROUTINE TO ENTER THE
                                ;       PLAYER'S MOVE
                                ;
                                DISMV     LDXIM   04            ROTATE
                                ROL       ASLZ    .DIS3          KEY
                                          ROLZ    .DIS2         INTO
                                          DEX                   DISPLAY
                                          BNE     ROL
                                          ORAZ    .DIS3
                                          STAZ    .DIS3
                                          STAZ    .SQUARE
                                          RTS
                                ;
                                ;       THE FOLLOWING SUBROUTINE ASSIGNS
                                ;       A VALUE TO THE MOVE UNDER
                                ;       CONSIDERATION AND RETURNS IT IN
                                ;         THE ACCUMULATOR
                                ;
                                          +++
1780 18                         STRATGY   CLC
1781 A9 80                                LDAIM   80
1783 65 EB                                ADCZ    .WMOB         PARAMETERS
1785 65 EC                                ADCZ    .WMAXC        WITH WEIGHT
1787 65 ED                                ADCZ    .WCC          OF 0.25
1789 65 E1                                ADCZ    .WCAP1
178B 65 DF                                ADCZ    .WCAP2
178D 38                                   SEC
```

```
551  178E E5 F0              SBCZ    .PMAXC
552  1790 E5 F1              SBCZ    .PCC
553  1792 E5 E2              SBCZ    .BCAP0
554  1794 E5 E0              SBCZ    .BCAP1
555  1796 E5 DE              SBCZ    .BCAP2
556  1798 E5 EF              SBCZ    .PMOB
557  179A E5 E3              SBCZ    .BMOB
558  179C B0 02              BCS     POS             UNDERFLOW
559  179E A9 00              LDAIM   00              PREVENTION
560  17A0 4A          POS    LSRA
561  17A1 18                 CLC                     **************
562  17A2 69 40              ADCIM   40
563  17A4 65 EC              ADCZ    .WMAXC          PARAMETERS
564  17A6 65 ED              ADCZ    .WCC            WITH WEIGHT
565  17A8 38                 SEC                     OF 0.5
566  17A9 E5 E4              SBCZ    .BMAXC
567  17AB 4A                 LSRA                    ************
568  17AC 18                 CLC
569  17AD 69 90              ADCIM   90
570  17AF 65 DD              ADCZ    .WCAP0          PARAMETERS
571  17B1 65 DD              ADCZ    .WCAP0          WITH WEIGHT
572  17B3 65 DD              ADCZ    .WCAP0          OF 1.0
573  17B5 65 DD              ADCZ    .WCAP0
574  17B7 65 E1              ADCZ    .WCAP1
575  17B9 38                 SEC                     [UNDER OR OVER-
576  17BA E5 E4              SBCZ    .BMAXC           FLOW MAY OCCUR
577  17BC E5 E4              SBCZ    .BMAXC           FROM THIS
578  17BE E5 E5              SBCZ    .BCC             SECTION]
579  17C0 E5 E5              SBCZ    .BCC
580  17C2 E5 E0              SBCZ    .BCAP1
581  17C4 A6 B1              LDXZ    .SQUARE         ************
582  17C6 E0 33              CPXIM   33
583  17C8 F0 16              BEQ     POSN            POSITION
584  17CA E0 34              CPXIM   34              BONUS FOR
585  17CC F0 12              BEQ     POSN            MOVE TO
586  17CE E0 22              CPXIM   22              CENTRE
587  17D0 F0 0E              BEQ     POSN               OR
588  17D2 E0 25              CPXIM   25              OUT OF
589  17D4 F0 0A              BEQ     POSN            BACK RANK
590  17D6 A6 B0              LDXZ    .PIECE
591  17D8 F0 09              BEQ     NOPOSN
592  17DA B4 50              LDYZX   .BOARD
593  17DC C0 10              CPYIM   10
594  17DE 10 03              BPL     NOPOSN
595  17E0 18          POSN   CLC
596  17E1 69 02              ADCIM   02
597  17E3 4C 77 03    NOPOSN JMP     CKMATE              CONTINUE
598                          ;
599                          ;
600                          ;
```

SYMBOL TABLE

SYMBOL	ADDR	DEF	CROSS REFERENCES
CHESS	0000	5	1 45 171 519
OUT	0008	15	18
WHSET	001A	24	27
NOSET	0027	32	22
NOREV	0032	38	33
CLDSP	0039	42	30 36
NOGO	0041	47	39
NOMV	004B	51	48
JANUS	0100	60	236 245 257 270
COUNTS	0104	67	
OVER	0110	74	68 70
NOQ	0118	79	76
ELOOP	011E	82	85
FOUN	0126	86	83
LESS	0131	92	88
NOCAP	0138	98	79
XRT	013E	101	72
ON4	013F	106	99
NOCOUNT	0160	121	61
RETJ	016E	132	129 138 144
TREE	016F	138	100 122
LOOPX	0175	141	145
FOUNX	017F	146	142
NOMAX	0188	150	148
UPTREE	0193	155	153
INPUT	0196	160	51
DISP	019D	163	50
SEARCH	019F	164	168
HERE	01A8	169	166
ERROR	01AC	171	161
GNMZ	0200	178	112 505
GNMX	0202	179	501
CLEAR	0204	180	182
GNM	0209	184	117 355 386
NEWP	020D	186	206 209 215 221 229 243 244 250
NEX	0212	190	187
KING	022D	204	205
QUEEN	0234	207	201 208
ROOK	023B	211	202
AGNR	023F	213	214
BISHOP	0246	217	199 220
KNIGHT	0251	223	197
AGNN	0255	225	228
PAWN	0260	231	195
P1	0264	233	241
P2	026E	237	234 235
P3	0279	242	249
SNGMV	028E	255	204 225
ILL1	0296	258	256
LINE	029C	265	207 213 217 267 272
OVL	02A3	268	266
ILL	02AC	273	268
REVERSE	02B2	280	34 111 113 354 385 387
ETC	02B4	281	292
CMOVE	02CA	311	233 242 255 265
LOOP	02DB	321	324
NO	02EC	333	322

```
SYMBOL    ADDR DEF  CROSS REFERENCES

SPX       02ED 335   331
CHKCHK    02F5 348
RETL      0315 366   336 338 361
ILLEGAL   0319 370   317 327 343
RESET     031E 377   190 237 258 273
GENRM     0325 384   154
GENR2     0328 385
RUM       032E 387   356
UMOVE     0331 392   118
MOVE      034B 415    49 110 353 384 518
CHECK     0357 423   426
TAKE      035E 427   424
STRV      0370 439   408
CKMATE    0377 449   597
NOCHEK    0381 455   451
RETV      038B 461   453 456 458
PUSH      038F 468
RETP      039F 476   469 470
GO        03A2 481    40
END       03BB 496   485
NOOPEN    03BD 497   482
MV2       03D5 512   494
MATE      03E7 521   510
DISMV     03EA 527   162
ROL       03EC 528   531
STRATGY   1780 543   120
POS       17A0 560   558
POSN      17E0 595   583 585 587 589
NOPOSN    17E3 597   591 594
.BOARD    0050 602    25 164 284 286 289 290 323 378 402 407
                      423 428 432 433 513 592
.BK       0060 603    82 127 141 282 285
.SETW     0070 604    24
.MOVEX    008F 605   314
.POINTS   00A0 606    86 146 450
.PIECE    00B0 607    67 170 185 186 191 377 399 431 472 515
                      590
.SQUARE   00B1 608    81 128 140 246 311 315 318 379 406 419
                      474 517 534 581
.SP2      00B2 609     9 394 417 440
.SP1      00B3 610   393 416 441
.INCHEK   00B4 611   131 352 360
.STATE    00B5 612    60 109 116 150 152 155 335 351 359 462
                      498 504
.MOVEN    00B6 613   193 212 218 224 226 232 238 239 259 274
                      312 397 437
.OMOVE    00DC 614    28 481 493 496
.OPNING   00DC 615   484 487 490
.WCAP0    00DD 616   107 570 571 572 573
.COUNT    00DE 617   180
.BCAP2    00DE 618   555
.WCAP2    00DF 619   549
.BCAP1    00E0 620   554 580
.WCAP1    00E1 621   548 574
.BCAP0    00E2 622   147 149 553
.MOB      00E3 623    74  77
.MAXC     00E4 624    87  90
.CC       00E5 625    94  95
```

```
SYMBOL    ADDR DEF  CROSS REFERENCES

.PCAP     00E6 626    89
.BMOB     00E3 627   455 557
.BMAXC    00E4 628   449 566 576 577
.BCC      00E5 629   578 579
.BMAXP    00E6 630    71
.XMAXC    00E8 631   106
.WMOB     00EB 632   545
.WMAXC    00EC 633   546 563
.WCC      00ED 634   547 564
.WMAXP    00EE 635   457
.PMOB     00EF 636   556
.PMAXC    00F0 637   551
.PCC      00F1 638   552
.PCP      00F2 639
.OLDKY    00F3 640    17  19
.BESTP    00FB 641   473 512
.BESTV    00FA 642   468 471 499 508 514
.BESTM    00F9 643   475 516
.DIS1     00FB 644    42 169 488
.DIS2     00FA 645    43 165 529
.DIS3     00F9 646    44 483 491 528 532 533
*OUT      1F1F 647    15 476
*GETKEY   1F6A 648    16
```

BLOCK DATA

```
.SETW     0070    03 04 00 07 02 05 01 06 10 17 11 16 12 15 14 13
                  73 74 70 77 72 75 71 76 60 67 61 66 62 65 64 63

.MOVEX    0090    F0 FF 01 10 11 0F EF F1 DF E1 EE F2 12 0E 1F 21

.POINTS   00A0    0B 0A 06 06 04 04 04 04 02 02 02 02 02 02 02 02

.OPNING   00C0    99 25 0B 25 01 00 33 25 07 36 34 0D 34 34 0E 52
                  25 0D 45 35 04 55 22 06 43 33 0F CC
```

NOTE THAT 00B7 TO 00BF, 00F4 TO 00F8, AND 00FC TO 00FF ARE AVAILABLE FOR USER EXPANSION AND I/O ROUTINES.

Micro-ADE 6502

ASSEMBLER

This flexible two pass assembler can be used to assemble small programs directly in memory, or with up to two computer controlled cassettes for easy handling of large programs. The allocation of memory to the source, object, and symbol table is user defined. The symbol table may be listed at any time in alphabetical or address order. Efficient packed ASCII coding reduces the memory required by the symbol table. Error messages warn you of mistakes before the program crashes the system.

DISASSEMBLER

The disassembler translates object code into assembler source language. Symbolic arguments and labels are defined from the symbol table. The assembler symbol table can be saved at assembly time for use with the disassembler for easy debugging. Relocation of undocumented programs becomes a snap. Use this disassembler once, and you'll never look at a hex dump again!

EDITOR

Quick edit features include the FIX, INSERT, MOVE, and DELETE commands. Lines are automatically numbered. Cassette commands: GET, SAVE, and REPRODUCE simplify the editing of multiple file source programs on cassettes. A page mode formats the output for CRT terminals to allow easy viewing of long listings.

Micro-ADE is a well documented package of programs which may be used with any 6502 microcomputer system. The comprehensive 56 page user manual includes the full source listing for all input/output and KIM cassette I/O routines enabling you to interface your own peripheral devices with ease. All programs and utility routines coreside in 4K. Schematics are included for automatic control of two cassette recorders.

Full documentation is available from Micro-Ware Limited. The User Manual, hex dump, and object program on paper tape or KIM cassette costs only $25.00. The complete annotated source listing is also available for an additional $25.00. This is the program development tool that you have been waiting for. Send today to:

Micro-Ware Ltd 27 FIRSTBROOKE ROAD, TORONTO, ONTARIO, CANADA. M4E 2L2.

Micro-Ware Limited
27 Firstbrooke Road
Toronto Ontario
Canada M4E 2L2

Printed in Canada

MICROCHESS

A CHESS PLAYING PROGRAM FOR THE 8080 MICROCOMPUTER

Written By: P. Jennings & T. O'Brien

MICRO-WARE LTD.

MICROCHESS

MICROCHESS was originally conceived as a program which would play chess using only a minimum hobbyist microcomputer system. The program which was developed will run on any 8080 based microcomputer configured with at least 4K of contiguous RAM, and an ASCII input/output device.

Although MICROCHESS does not always play chess at the expert level, it will make a reasonable move under most circumstances. In addition to being great fun to play, it can provide a useful and tireless opponent for practising checkmates, learning openings, and sharpening general playing skills.

The MICROCHESS program is supplied on paper tape or on Tarbell(TM) cassette. The documentation provided includes complete player's instructions, a description of the program operation, and an appendix with details for modifying the I/O to suit the individual requirements of each user's personal computer system. If you should have any problems putting MICROCHESS up on your system, please send the details of your system and the exact problem to the address below. We will do our best to assist you in any way possible.

M I C R O C H E S S,
MICRO-WARE LIMITED,
27 FIRSTBROOKE RD.,
TORONTO, ONT.,
M4E 2L2,
CANADA

TABLE OF CONTENTS

Player's Manual

NOTATION

MICROCHESS uses a special octal notation to identify the squares of the chess board. Each square is represented by a two digit number. The first digit specifies the rank(0 to 7) from the computer's end of the board. The second digit specifies the file (0 to 7) from the computer's right (your left). A completely numbered board is shown below:

M I C R O C H E S S

00	01	02	03	04	05	06	07
10	11	12	13	14	15	16	17
20	21	22	23	24	25	26	27
30	31	32	33	34	35	36	37
40	41	42	43	44	45	46	47
50	51	52	53	54	55	56	57
60	61	62	63	64	65	66	67
70	71	72	73	74	75	76	77

C H A L L E N G E R

PROGRAM EXECUTION

The MICROCHESS program is executed from address ØØØØ. After printing the initial sign-on message, MICROCHESS will ask: "DO YOU WANT WHITE? (Y,N)". If you wish to play white, respond with 'Y'. If you wish to play black, respond with 'N'. If you wish MICROCHESS to decide which colour to play, respond with any other character. MICROCHESS will then display the board and prompt with a colon, indicating that the program is ready to receive any operating command.

MICROCHESS COMMANDS

MICROCHESS has seven special commands to which it will respond. Commands may be abbreviated to the first letter of the command word. All commands must be terminated with a carriage return. Typing errors may be corrected at any time by typing a control-X. This will clear the input buffer and allow you to retype the entire line.

COMMAND SUMMARY

COMMAND	FUNCTION
DISPLAY	Display the board at the terminal.
GO	Make a move from the current position.
SPEED	Change the mode of the computer's play.
RESIGN	End the game.
EXCHANGE	Exchange sides.
AUTO DISPLAY	Display the board after each move.
NO DISPLAY	Do not display the board automatically.

THE DISPLAY COMMAND

The DISPLAY command instructs the computer to display the current position of the internal chess board at the terminal. MICROCHESS is always illustrated at the top of the display, and you are always at the bottom. Each piece is indicated by a two character mnemonic. The first character shows the colour of the piece. The second character shows the type of piece occupying that square. Black squares which are unoccupied are illustrated by :: . The sample display below shows the board set up to begin a game with MICROCHESS playing white.

```
+------- MICROCHESS -------+
! WR WN WB WK WQ WB WN WR !
!                         !
! WP WP WP WP WP WP WP WP !
!                         !
!    ::    ::    ::    :: !
!                         !
! ::    ::    ::    ::    !
!                         !
!    ::    ::    ::    :: !
!                         !
! ::    ::    ::    ::    !
!                         !
! BP BP BP BP BP BP BP BP !
!                         !
! BR BN BB BK BQ BB BN BR !
+------- CHALLENGER -------+
```

THE GO COMMAND

The GO command instructs MICROCHESS to examine the current postion of the board, choose the best move available, make that move, and then print out the move that it has made. This command may be entered at any time. The computer will not check to see if you have made any moves since the last computer move, or if it is making the first move with the black men. MICROCHESS trusts you. You must referee the game.

THE SPEED COMMAND

MICROCHESS can play chess at three different levels. The best level is called the NORMAL speed, and requires from 60 to 300 seconds per move for analysis. By eliminating some time consuming portions of the strategic analysis, the speed can be increased. BLITZ mode requires only 20 seconds per move on the average, and SUPERBLITZ will make a move in about 10 seconds. In response to the SPEED command, MICROCHESS will ask: "WHICH MODE? (S,B,N)". Type one of the characters S,B, or N to choose the desired speed. This command may be entered at any time during the game.

SPEED SUMMARY

ENTER	SPEED	TIME PER MOVE
S	SUPERBLITZ	5 TO 10 SECONDS
B	BLITZ	10 TO 30 SECONDS
N	NORMAL	30 TO 300 SECONDS

THE RESIGN COMMAND

The RESIGN command may be entered at any time to end the game. MICROCHESS will display the final position of the board, and then ask if you wish to play again.

THE EXCHANGE COMMAND

The EXCHANGE command enables you to turn the board around at any point during the game. This forces MICROCHESS to play with your pieces in the position that you have left them. You must play with the computer's men. The relative positions of the pieces remain the same, but the numbering of the squares changes because the notation always has its origin at the computer's lower right.

It is possible to have MICROCHESS play a game against itself by entering the EXCHANGE command, then the GO command, then the EXCHANGE command, and so on. Remember that each move printed is being described from opposite ends of the board because of the intervening exchanges. It is best to display the board every two or three moves to be sure that you are following the game correctly.

ENTERING YOUR MOVE

Your move is described to MICROCHESS by specifying the square the piece was moved from, and the square the piece was moved to, using the octal notation described above. For example, with the computer playing white, a KP to KP4 response would be entered at the colon prompt as:

```
:       63-43
```

MICROCHESS will immediately move the appropriate piece internally and begin to consider its response. The GO command is assumed as soon as the move is entered. Note that MICROCHESS carries out no legal validity check on your move. The program will accept a move of any piece on the board to any square on the board. If the square you move the piece to is occupied, the occupying piece will be captured and removed from the board. Therefore, it is very important when entering your move, to take great care not to enter an incorrect square number. As with the commands, typing errors may be corrected by typing a control-x and retyping the entire line.

SPECIAL MOVES

Normally, MICROCHESS begins to consider its response as soon as you have entered your move in the format shown above. If you wish to inhibit this action, in order to make two consecutive moves to set up a test position, or to make an en passant capture as described below, enter an M after the move. For example:

```
:       63-43M
:
```

MICROCHESS will move the appropriate piece on its internal chess board, and then return to the command mode for further commands or moves. Note once again, that you may move any piece on the board in this manner, This includes the computer's pieces , which you may wish to move in order to set up a special position.

CASTLING

Castling is accomplished by entering O-O to castle on the king's side (short), and O-O-O to castle on the queen's side (long). The letter O is used, not the numeral Ø.

```
:O-O
```

PAWN PROMOTION

If you move a pawn to the eighth rank (rank Ø in the octal notation of MICROCHESS), you may promote it to a piece. This may be done by following the move entry by an equal sign and the mnemonic of the piece you wish the pawn promoted to. For example, if you wish to promote the king pawn to a Queen, the following move would be entered:

```
:       13-Ø3=Q
```

Because of the internal board representation of MICROCHESS, only one queen is allowed per side at any given time. If you already have a queen, it will be necessary to choose another piece which has already been lost.

EN PASSANT

En passant pawn capture may be accomplished by making two moves with the capturing pawn. The first move is a lateral move to capture the computer's pawn. The second move is forwards to the final square that you are moving your pawn to. For example, a capture of the computer's queen pawn which has just moved from 14 to 34 with your king pawn, now located at 33, is accomplished by first moving 33 to 34 to capture the pawn (using the M suffix to prevent MICROCHESS from moving), and then moving from 34 to 24 to move your pawn to the appropriate final square.

```
:       33-34M
:       34-24
```

THE COMPUTER MOVE

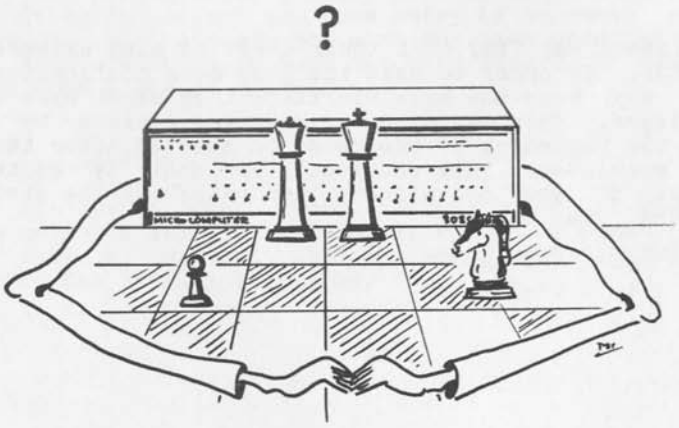
MICROCHESS indicates its move using the same notation that you use to enter your moves. To distinguish your moves from those of the computer when going over an old listing, the computer's moves are preceded by the notation: MC : , as shown in the example game illustrated in appendix F. En passant capture is not a part of the MICROCHESS move generation routines. Consequently, the computer will never capture en passant or recognize the danger of you capturing en passant when it formulates its optimum move.

NOTES

Some players may find that their level of play exceeds that of MICROCHESS. In order to make the game more challenging, these players may make the same sacrifice they might make to a weak human player. They can spot the computer a piece by removing it at the beginning of the game, or shortly after the opening play is concluded. This can easily be done by capturing it with one of your own pieces, then returning the piece to its own square. For example:

```
:   74-73M
:   73-74M
```

MICROCHESS has been designed for your enjoyment. Have fun! In addition, we are always open to suggestions, ideas, or criticisms. Please let us know if you feel that there is anything we can do to improve our products, or if there are any new products you would like to see us present.



APPENDIX A

THE PROGRAM

The program is divided into three functionally distinct sections: the control and input/output routines, the move generation and data collection routines, and the strategic analysis routines.

CONTROL AND INPUT/OUTPUT

This section of the program is responsible for all communications between the computer and the human player. The primary functions carried out are the board set up, and data table initialization sections. In addition to this, the various input commands are interpreted and subroutines are called which execute them. The most important subroutine called by the control section is the chess program itself. This is a complex set of routines which examine the current state of the chess board and return a move which has been evaluated as the best available.

MOVE GENERATION

The second major subsection of the program consists of a set of subroutines which generate legal moves from a given position. MICROCHESS, unlike most larger chess playing programs, evaluates its opportunities in a serial manner. That is, it generates an available move, and evaluates it completely before generating the next available move. The evaluation routines calculate a value for each move which is compared with the value of the best move found so far. If it is better, it becomes the best move for comparison with future moves generated. The move with the highest value will be selected by MICROCHESS.

To generate all the moves for a side MICROCHESS works through a table which contains the board position of each piece. This is the table shown in appendix E. First, a king pawn move is generated and evaluated. The evaluation includes the actual moving of the piece, and the generation of potential reply moves by the challenger. The sequence of trial moves of the computer's pieces and the challenger's pieces may extend as far as three moves for each side beyond the current position. At the end of this time, each move made will be taken back, until the board is returned to its original state. Then, the next available move will be made, and the replies tested. This continues until all the moves for each piece have been tested. MICROCHESS is capable of generating and evaluating about 10,000 moves per second. Thus, in a 300 second analysis 3,000,000 moves will be made and taken back in an attempt to evaluate the available moves.

DATA COLLECTION

For each test move available to the computer data are collected which will allow it to evaluate the resulting position. In the normal mode of operation MICROCHESS collects the folowing information for use by the strategy algorithms.

MOBILITY (μ) This represents the number of legal moves that a side has available to it from a given position.

MAXIMUM CAPTURABLE PIECE (ρ). The value of the most valuable piece presently being attacked by a side.

TOTAL ATTACK (α). The sum of the values of all the pieces under attack by a side.

CAPTURE (ψ). The value of any piece captured by the current move, or the maximum available capture in a future move which can be achieved by a series of captures (an exchange).

The mobility, maximum capturable piece, and the total attack are obtained for the current position, and the position after the test move has been made for both the computer and its opponent. Capture values are calculated to a depth of three moves per side beyond the current position provided the position examined can be achieved by a continuous sequence of piece captures. In addition, the value of the moving piece, and the squares it occupies before and after the move are used in the evaluation.

STRATEGY

After a test move has been generated, and the parameters above have been collected by the data collection routines, the strategic analysis algorithm assigns a value to the move. The basic algorithm is a linear combination of the various parameters. The basic value is then modified by factors such as the availability of a checkmate, or a positional bonus for motion to the center or out of the back rank.

$$\begin{aligned} VALUE = \; & 4.00\psi_1 + 1.25\psi_2 + 0.75\rho_1 + 0.75\alpha_1 + 0.25\mu_1 \\ & + 0.25\psi_3 - 3.00\rho\acute{1} - 2.00\alpha\acute{1} - 1.25\psi\acute{1} - 0.25\rho_0 \\ & - 0.25\alpha_0 - 0.25\psi\acute{2} - 0.25\psi\acute{3} - 0.25\mu_0 \end{aligned}$$

(´) signifies the challenger's value.

(n) subscript signifies the position at time n.
(time 0 is the current board position)

VALUE = VALUE + 02 if a piece is moved from the back rank.

VALUE = VALUE + 02 if a piece is moved to the centre.

VALUE = FF if the challenger is checkmated.

The algorithm used by MICROCHESS is a relatively simple one compared to major chess programs which can compete at an expert level of play. As a result, the computer must make the decision between positional development, or material advantage based upon the few factors outlined above. Good chess is considerably more complex, and requires that the player use algorithms which vary from time to time during the game. MICROCHESS has only a single algorithm which must be used at all stages during the game (except for a few opening moves which can be played from a limited book). This single algorithm is a compromise of the possible opening, middle game, end game, and special situation algorithms. It is because of this compromise that MICROCHESS sometimes makes moves which are not optimal.

APPENDIX B

INPUT AND OUTPUT SUBROUTINES

MICROCHESS is supplied with input and output subroutines for use with an ASR 33 or equivalent ASCII terminal. These routines are shown below in source format:

```
                    4130 ********************************************
                    4140 * TELETYPE INPUT/OUTPUT ROUTINES SUPPLIED. *
                    4150 ********************************************
                    4160 *                OUTPUT ROUTINE            *
                    4170 ********************************************
0DE6 DB 00          4180 TTYO   IN    0
0DE8 E6 80          4190        ANI   80H
0DEA CA E6 0D       4200        JZ    TTYO
0DED 78             4210        MOV   A,B
0DEE D3 01          4220        OUT   1
0DF0 C9             4230        RET
0DF1                4240 ********************************************
0DF1                4250 *                 INPUT ROUTINE            *
0DF1                4260 ********************************************
0DF1 DB 00          4270 TTYI   IN    0
0DF3 E6 40          4280        ANI   40H
0DF5 CA F1 0D       4290        JZ    TTYI
0DF8 DB 01          4300        IN    1
0DFA E6 7F          4310        ANI   7FH
0DFC 47             4320        MOV   B,A
0DFD C9             4330        RET
```

The conventions used by these routines are:

1- Status is on channel 0.
2- Data is on channel 1.
3- Data available is signalled by bit 6 (40H).
4- Transmit buffer empty is signalled by bit 7 (80H).

These routines are shown, so that you may modify them if necessary to suit the individual requirements of your system.

If you wish to use your own I/O routines replace the data at address 09DA (C3 F1 0D) with a JMP to your own input routine (C3 XX XX). Then, replace the data at address 09D7 (C3 E6 0D) with a JMP to your own output routine (C3 XX XX).

The data is passed in the B register. The parity bit may be 0 or 1 for an input instruction. Output from MICROCHESS has the parity bit set to 0. There is no requirement for saving any of the registers; however, the stack pointer must be preserved and the routines must end with a return instruction.

APPENDIX C

DISPLAY OPTIONS

Two display option commands are available at the MICROCHESS command prompt. These are AUTO DISPLAY and NO DISPLAY. Entering the AUTO DISPLAY command causes the program to display the board immediately after each move made by either side. Entering the NO DISPLAY command will turn off the automatic display feature. This is demonstrated in the sample game in Appendix F.

The default option in the copy of MICROCHESS you have received is NO DISPLAY. The user may change the default option to allow the program to display the board after each computer move, after each of the challenger's moves, or both. Replacing the three NOP instructions at address Ø12Ø (ØØ ØØ ØØ) with a call to the display subroutine (CD 42 Ø2) will cause the board to be automatically displayed after each move made by MICROCHESS. If you wish to have the board displayed automatically after each of your moves as well, replace the three NOP instructions at address ØØD4 with the same subroutine call (CD 42 Ø2).

CRT DISPLAY

If you are using a CRT display with only 16 lines on the screen, you may wish to shorten the board display provided by MICROCHESS. This is easily accomplished by entering 3 NOP instructions (ØØ ØØ ØØ) at address Ø258. This replaces the CD DA Ø1 which appears in the original code.

CUSTOM BOARD DISPLAY

If you wish to design your own board display for use with a graphic terminal or just to gratify your own artistic ambitions, you may replace the MICROCHESS display routine by replacing the data at address Ø242 (CD AC Ø9) with a JMP to your own display subroutine (C3 XX XX).

The data required to display the board is contained in a table at address Ø9ED. This table contains the board location of each piece. The address and location of each piece as it would appear at the start of a game with MICROCHESS playing white is shown below.

PIECE ADDRESSES FOR BOARD DISPLAY

PIECE	MICROCHESS		CHALLENGER	
King	Ø9ED	Ø3	Ø9FD	73
Queen	Ø9EE	Ø4	Ø9FE	74
King Rook	Ø9EF	ØØ	Ø9FF	7Ø
Queen Rook	Ø9FØ	Ø7	ØAØØ	77
King Bishop	Ø9F1	Ø2	ØAØ1	72
Queen Bishop	Ø9F2	Ø5	ØAØ2	75
King Knight	Ø9F3	Ø1	ØAØ3	71
Queen Knight	Ø9F4	Ø6	ØAØ4	76
KR Pawn	Ø9F5	1Ø	ØAØ5	6Ø
QR Pawn	Ø9F6	17	ØAØ6	67
KN Pawn	Ø9F7	11	ØAØ7	61
QN Pawn	Ø9F8	16	ØAØ8	66
KB Pawn	Ø9F9	12	ØAØ9	62
QB Pawn	Ø9FA	15	ØAØA	65
Q Pawn	Ø9FB	14	ØAØB	64
K Pawn	Ø9FC	13	ØAØC	63

APPENDIX D

RETURNING TO YOUR OPERATING SYSTEM

If you wish to return directly to your operating system at the end of a game, this can be accomplished by replacing the HALT instruction at address Ø1D7 with a JMP xx xx to your operating system entry point. Two NOPs have been included for your convenience in adding this patch.

Please note, that it is impossible to call MICROCHESS as a subroutine because the program manipulates the stack pointer several times during program execution. Thus, the original return address will not be at the top of the stack when the return instruction is executed.

APPENDIX E

```
0000 31 82 0D CD AC 09 21 6D 0B CD DA 01 21 93 0B CD
0010 DA 01 21 B9 0B CD DA 01 CD AC 09 CD AF 05 21 95
0020 0A 22 7D 0A 21 CB 0A 22 7F 0A 21 01 0B 22 79 0A
0030 21 37 0B 22 7B 0A AF 32 74 0A 32 78 0A 32 82 0A
0040 32 81 0A 32 75 0A 32 76 0A 3E 10 32 77 0A 3E EE
0050 32 4D 0A 32 4E 0A 21 F5 0B CD 5C 01 CD C9 09 CD
0060 BF 09 78 0F D2 6F 00 3E 01 32 82 0A CD C3 05 CD
0070 42 02 CD AC 09 31 82 0D 21 E6 0D 22 59 0A 21 20
0080 20 22 20 0C 21 10 0C CD 5C 01 CD 20 03 CD AC 09
0090 21 0A 0D 7E FE 47 CA D7 00 FE 44 CA 42 03 FE 45
00A0 CA 2C 02 FE 53 CA E1 01 FE 4F CA D1 03 FE 52 CA
00B0 20 02 FE 41 CA F1 02 FE 4E CA 05 03 CD 68 01 3A
00C0 0F 0D FE 4D CA 26 01 FE 3D CA 48 03 FE 0D C2 2C
00D0 01 CD 6D 08 00 00 00 CD 7D 04 3A 78 0A B7 CA F4
00E0 00 21 4F 20 22 1E 0C 21 20 4F 22 1B 0C AF 32 78
00F0 0A C3 02 01 CD AD 01 CD 97 03 3A 4F 0A FE FF CA
0100 38 01 21 16 0C CD 5C 01 3A 81 0A FE FF CA 17 03
0110 3A 74 0A B7 CA 1D 01 21 AE 0C CD 5C 01 CD AC 09
0120 00 00 00 C3 75 00 CD 6D 08 C3 75 00 21 23 0C CD
0130 5C 01 CD AC 09 C3 75 00 21 30 0C CD DA 01 CD 42
0140 02 CD AC 09 21 47 0C CD 5C 01 CD C9 09 CD BF 09
0150 78 FE 59 C2 C8 01 CD AC 09 C3 18 00 7E FE 0D C8
0160 47 CD BF 09 23 C3 5C 01 2A 0A 0D CD 99 01 32 50
0170 0A 2A 0D 0D CD 99 01 32 51 0A 32 4E 0A 3A 50 0A
0180 21 0C 0A 0E 1F BE CA 91 01 2B 0D F2 85 01 C3 2C
0190 01 79 32 4D 0A 32 4F 0A C9 7D E6 0F 17 17 17 17
01A0 47 7C E6 0F B0 47 E6 88 C2 2C 01 78 C9 3A 50 0A
01B0 47 CD 8F 09 2A DD 09 22 1B 0C 3A 51 0A 47 CD 8F
01C0 09 2A DD 09 22 1E 0C C9 CD AC 09 CD AC 09 21 5B
01D0 0C CD 5C 01 CD AC 09 76 00 00 CD 5C 01 CD AC 09
01E0 C9 CD AC 09 21 DF 0B CD 5C 01 CD C9 09 CD BF 09
01F0 78 FE 53 CA 03 02 FE 42 CA 0A 02 FE 4E CA 11 02
0200 C3 2C 01 06 00 0E FF C3 15 02 06 00 0E FB C3 15
0210 02 06 08 0E FB 78 32 1E 07 79 32 61 08 C3 72 00
0220 CD AC 09 21 7E 0C CD 5C 01 C3 3E 01 CD C3 05 3A
0230 82 0A B7 CA 3A 02 AF C3 3C 02 3E 01 32 82 0A C3
0240 75 00 CD AC 09 CD AC 09 16 00 21 B6 0C CD DA 01
0250 7A B7 CA 5B 02 21 D2 0C CD DA 01 06 21 CD BF 09
0260 06 20 CD BF 09 21 0C 0A 0E 1F 7E BA CA B8 02 2B
0270 0D F2 6A 02 7A E6 0F 5F 7A E6 F0 0F 4F 4F 4F C3
0280 1F DA 89 02 06 20 C3 8B 02 06 3A CD BF 09 CD BF
0290 09 06 20 CD BF 09 14 7A E6 0F FE 08 C2 65 02 06
02A0 21 CD BF 09 CD AC 09 7A C6 08 57 F2 50 02 21 EE
02B0 0C CD DA 01 CD AC 09 C9 79 FE 10 D2 D3 02 3A 82
02C0 0A B7 C2 CC 02 3A 83 0A 47 C3 DD 02 3A 84 0A 47
02D0 C3 DD 02 3A 82 0A B7 C2 C5 02 C3 CC 02 CD BF 09
02E0 79 E6 0F 4F 06 00 21 85 0A 09 46 CD BF 09 C3 91
02F0 02 3E CD 32 D4 00 32 20 01 21 42 02 22 D5 00 22
0300 21 01 C3 72 00 21 00 00 22 D4 00 22 D5 00 22 20
0310 01 22 21 01 C3 72 00 21 96 0C CD 5C 01 C3 3E 01
0320 21 0A 0D 0E 00 CD C9 09 78 77 FE 0D CA 3C 03 FE
0330 18 CA 72 00 CD BF 09 23 0C C3 25 03 AF B9 CA 25
0340 03 C9 CD 42 02 C3 75 00 CD 68 01 21 0C 0A 0E 0F
0350 3A 50 0A BE CA 5C 03 2B 0D F2 53 03 36 CC 21 85
0360 0A 0E 07 3A 10 0D BE CA 72 03 2B 0D F2  6 03 C3
0370 2C 01 21 FD 09 06 00 09 3E CC BE CA 83 03 2B BE
0380 C2 2C 01 3A 51 0A 77 3A 11 0D FE 4D CA 72 00 FE
0390 0D CA D7 00 C3 2C 01 3A 51 0A E6 F0 FE 70 C0 3A
03A0 4F 0A 4F E6 08 C8 21 ED 09 06 00 09 3E CC 77 21
03B0 EE 09 1E 00 BE CA BD 03 23 1C C3 B4 03 3A 51 0A
03C0 77 3E 3D 32 20 0C 21 85 0A 16 00 19 7E 32 21 0C
03D0 C9 3A 0C 0D FE 4F C2 2C 01 3A 0E 0D FE 4F C2 2F
03E0 04 3A 82 0A B7 CA 05 04 3A FD 09 FE 74 C2 2C 01
03F0 3E 72 32 FD 09 3E 73 32 00 0A 32 4E 0A 3E 13 32
```

```
0400 4D 0A C3 1F 04 3A FD 09 FE 73 C2 2C 01 3E 75 32
0410 FD 09 3E 74 32 00 0A 32 4E 0A 3E 13 32 4D 0A 3A
0420 0F 0D FE 4D CA 72 00 FE 0D CA D7 00 C3 2C 01 3A
0430 82 0A B7 CA 53 04 3A FD 09 FE 74 C2 2C 01 3E 76
0440 32 FD 09 3E 75 32 FF 09 32 4E 0A 3E 12 32 4D 0A
0450 C3 6D 04 3A FD 09 FE 73 C2 2C 01 3E 71 32 FD 09
0460 3E 72 32 FF 09 32 4E 0A 3E 12 32 4D 0A 3A 0D 0D
0470 FE 4D CA 72 00 FE 0D CA D7 00 C3 2C 01 2A 75 0A
0480 7D FE 36 D2 6B 05 CD CF 04 F5 2A 75 0A 23 22 75
0490 0A F1 D2 6B 05 CD 4F 05 CD B7 04 CD 6D 08 3A 4D
04A0 0A FE 00 C0 3E 02 32 4D 0A 3A 4E 0A EE 03 32 4E
04B0 0A 32 78 0A C3 98 04 21 ED 09 3A 4D 0A 32 4F 0A
04C0 4F 06 00 09 7E 32 50 0A 3A 4E 0A 32 51 0A C9 CD
04D0 04 05 CA EB 04 CD 1C 05 21 77 0A 35 F2 E7 04 AF
04E0 26 00 2E F0 22 75 0A DA CF 04 C9 CD 29 05 2A 75
04F0 0A EB 2A 79 0A 19 7E 32 4D 0A 2A 7B 0A 19 7E 32
0500 4E 0A 37 C9 2A 75 0A EB 2A 7D 0A 19 3A 4D 0A BE
0510 C2 1B 05 2A 7F 0A 19 3A 4E 0A BE C9 2A 75 0A 7D
0520 C6 09 6F 22 75 0A FE 36 C9 2A 75 0A E5 3A 77 0A
0530 4F 3E FF 32 77 0A CD 1C 05 21 77 0A 34 D2 4A 05
0540 0D FA 4A 05 CD 04 05 CA 36 05 E1 22 75 0A C9 3A
0550 1E 07 3C 57 06 05 0E FF 3E FF 3D C2 5A 05 0D C2
0560 58 05 05 C2 56 05 15 C2 54 05 C9 3E 0C 32 54 0A
0570 32 50 0A 0E 14 CD DF 05 3E 04 32 54 0A CD DD 05
0580 3A 50 0A FE 0F DA A3 05 21 ED 09 06 00 3A 4F 0A
0590 32 4D 0A 4F 09 7E 32 50 0A 3A 51 0A 32 4E 0A CD
05A0 6D 08 C9 3E FF 32 4F 0A 32 50 0A 32 51 0A C9 21
05B0 0D 0A 11 ED 09 0E 20 7E 12 23 13 0D C2 B7 05 C9
05C0 CD AF 05 21 ED 09 11 FD 09 0E 10 3E 77 96 47 EB
05D0 3E 77 96 70 EB 77 23 13 0D C2 CB 05 C9 0E 10 21
05E0 5D 0A AF 77 23 0D F2 E3 05 3E 10 32 4D 0A 21 4D
05F0 0A 35 F8 CD 58 07 3E 08 32 55 0A 3A 4D 0A FE 08
0600 F2 56 06 FE 06 F2 43 06 FE 04 F2 35 06 FE 01 CA
0610 1E 06 F2 27 06 CD 98 06 C2 15 06 C3 EE 05 CD A9
0620 06 C2 1E 06 C3 EE 05 3E 04 32 55 0A CD A9 06 C2
0630 2C 06 C3 EE 05 CD A9 06 3A 55 0A FE 04 C2 35 06
0640 C3 EE 05 3E 10 32 55 0A CD 98 06 3A 55 0A FE 08
0650 C2 48 06 C3 EE 05 3E 06 32 55 0A CD DA 06 FA 6B
0660 06 3A 52 0A B7 CA 6B 06 CD 93 07 CD 58 07 21 55
0670 0A 35 7E FE 05 CA 5B 06 CD DA 06 DA 8B 06 FA EE
0680 05 3A 52 0A B7 C2 EE 05 CD 93 07 3A 4E 0A E6 F0
0690 FE 20 CA 78 06 C3 EE 05 CD DA 06 FA A1 06 CD 93
06A0 07 CD 58 07 21 55 0A 35 C9 CD DA 06 D2 C0 06 F5
06B0 E1 22 72 0A 3A 52 0A B7 CA A9 06 2A 72 0A E5 F1
06C0 FA D2 06 3A 52 0A F5 CD 93 07 F1 32 52 0A B7 CA
06D0 A9 06 CD 58 07 21 55 0A 35 C9 3A 55 0A 06 00 21
06E0 2C 0A 4F 09 3A 4E 0A 86 32 4E 0A E6 88 C2 51 07
06F0 3A 4E 0A 0E 1F 21 0C 0A BE CA 04 07 2B 0D F2 F8
0700 06 C3 12 07 79 FE 10 DA 51 07 3E 01 32 52 0A C3
0710 16 07 AF 32 52 0A 3A 54 0A B7 FA 4F 07 FE 00 F2
0720 4F 07 F5 3A 52 0A F5 3E F9 32 54 0A 32 53 0A CD
0730 6D 08 CD C3 05 CD E9 05 CD 70 07 F1 32 52 0A F1
0740 32 54 0A 3A 53 0A B7 FA 4F 07 3E 80 B7 37 C9 AF
0750 C9 AF 32 52 0A 2F B7 C9 21 ED 09 06 00 3A 4D 0A
0760 4F 09 7E 32 4E 0A C9 CD 6D 08 CD C3 05 CD E9 05
0770 CD C3 05 21 00 00 39 22 57 0A 2A 59 0A F9 D1 21
0780 55 0A 72 21 4D 0A 73 C1 E1 71 E1 70 78 32 4E 0A
0790 C3 A0 08 3A 54 0A B7 16 00 5F FA 20 08 3A 4D 0A
07A0 B7 CA B2 07 47 3E 08 BB C2 B2 07 3A 65 0A B8 CA
07B0 FB 07 21 62 0A 19 34 3E 01 B8 C2 BE 07 34 3A 52
07C0 0A B7 CA F2 07 0E 0F 21 0C 0A 3A 4E 0A BE CA D7
07D0 07 2B 0D F2 CD 07 76 21 3D 0A 06 00 09 7E 21 63
07E0 0A 19 BE DA EC 07 77 21 65 0A 19 71 21 64 0A 19
07F0 86 77 7B FE 04 CA FC 07 FA 34 08 C9 3A 67 0A 32
```

```
0800 5C 0A AF 32 54 0A CD 6D 08 CD C3 05 CD DD 05 CD
0810 C3 05 3E 08 32 54 0A CD E9 05 CD 73 07 C3 AC 08
0820 16 FF FE F9 C2 34 08 3A FD 09 21 4E 0A BE C0 AF
0830 32 53 0A C9 3A 52 0A B7 C8 3A 4E 0A 0E 07 21 04
0840 0A BE CA 4B 08 2B 0D C2 41 08 C9 21 3D 0A 06 00
0850 09 7E 21 61 0A 19 BE DA 5B 08 77 1B 7B 32 54 0A
0860 FE FF CA 68 08 CD 67 07 21 54 0A 34 C9 21 00 00
0870 39 22 57 0A 2A 59 0A F9 3A 4E 0A 47 0E 1F 21 0C
0880 0A BE CA 8A 08 2B 0D F2 81 08 36 CC E5 21 ED 09
0890 16 00 3A 4D 0A 5F 19 E5 4E 70 C5 21 55 0A 56 D5
08A0 21 00 00 39 22 59 0A 2A 57 0A F9 C9 97 3E 80 21
08B0 6A 0A 86 23 86 23 86 21 60 0A 86 21 5E 0A 86 21
08C0 6F 0A 96 23 96 21 61 0A 96 21 5F 0A 96 21 5D 0A
08D0 96 21 6E 0A 96 21 62 0A 96 D2 DD 08 97 1F C6 40
08E0 21 6B 0A 86 23 86 21 63 0A 96 1F C6 90 21 5C 0A
08F0 86 86 86 86 21 60 0A 86 21 63 0A 96 96 23 96 96
0900 21 5F 0A 96 F5 3A 4E 0A FE 33 CA 30 09 FE 34 CA
0910 30 09 FE 22 CA 30 09 FE 25 CA 30 09 3A 4D 0A B7
0920 CA 34 09 21 ED 09 06 00 4F 09 7E FE 10 F2 34 09
0930 F1 C6 02 F5 3A 63 0A 21 3D 0A BE C2 43 09 F1 97
0940 C3 5B 09 3A 62 0A B7 C2 5A 09 3A 6D 0A B7 C2 5A
0950 09 F1 3E FF 32 81 0A C3 5B 09 F1 0E 04 21 54 0A
0960 71 21 50 0A 32 5B 0A BE DA 8E 09 CA 8E 09 32 50
0970 0A 3A 4D 0A 32 4F 0A 3A 4E 0A 32 51 0A AF 32 74
0980 0A 3A 6B 0A 21 3D 0A BE C2 8E 09 32 74 0A C9 21
0990 DD 09 47 1F 1F 1F 1F CD A2 09 77 23 78 CD A2 09
09A0 77 C9 E6 0F C6 30 FE 3A D8 C6 07 C9 06 0D CD BF
09B0 09 06 0A CD BF 09 06 7F CD BF 09 CD BF 09 C9 E5
09C0 C5 D5 CD D7 09 D1 C1 E1 C9 E5 D5 C5 CD DA 09 78
09D0 C1 D1 E1 E6 7F 47 C9 C3 E6 0D C3 F1 0D 00 00 00
09E0 00 00 00 00 00 00 00 00 00 00 00 00 00 03 04 00
09F0 07 02 05 01 06 10 17 11 16 12 15 14 13 73 74 70
0A00 77 72 75 71 76 60 67 61 66 62 65 64 63 03 04 00
0A10 07 02 05 01 06 10 17 11 16 12 15 14 13 73 74 70
0A20 77 72 75 71 76 60 67 61 66 62 65 64 63 F0 FF 01
0A30 10 11 0F EF F1 DF E1 EE F2 12 0E 1F 21 0B 0A 06
0A40 06 04 04 04 04 02 02 02 02 02 02 02 02 EE EE 00
0A50 00 00 00 00 00 00 00 00 00 E6 0D 00 00 00 00 00
0A60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0A70 00 00 00 00 00 00 00 10 00 01 0B 37 0B 95 0A CB
0A80 0A 00 00 57 42 4B 51 52 52 42 42 4E 4E 50 50 50
0A90 50 50 50 50 50 EE 1F 17 16 16 14 16 1B 16 EE 1F
0AA0 1E 16 14 16 11 10 1D 1E 1D 16 1A 14 12 17 11 11
0AB0 1F 16 17 14 12 1E 15 1B 12 1F 16 14 12 1E 11 14
0AC0 1E 17 1F 16 14 1D 1E 1D 17 12 1B EE 43 55 52 33
0AD0 63 54 55 66 EE 53 44 52 63 64 63 72 45 43 42 55
0AE0 56 66 75 52 62 52 44 55 52 31 75 53 36 52 74 44
0AF0 55 31 75 43 64 22 34 52 44 55 42 52 43 43 52 75
0B00 52 0F 06 04 00 0E 01 04 0E 07 0F 0E 07 05 0F 05
0B10 01 0C 06 06 0F 0B 05 04 00 06 06 0C 0F 07 06 04
0B20 00 0E 04 01 07 0F 07 06 06 04 06 0B 06 00 0F 07
0B30 04 06 0F 04 06 06 04 33 22 46 01 34 13 55 43 25
0B40 33 34 25 41 43 63 14 32 22 25 24 21 11 14 06 44
0B50 52 35 34 22 25 41 06 23 52 14 03 34 22 25 44 14
0B60 23 22 11 06 34 22 32 25 43 41 44 52 52 1D 49 43
0B70 52 4F 43 48 45 53 53 20 20 20 20 20 20 20 20 20
0B80 20 20 20 20 20 20 20 20 20 20 28 43 29 20 31 39 37
0B90 37 2E 0D 57 52 49 54 54 45 4E 20 42 59 3A 20 50
0BA0 2E 20 4A 45 4E 4E 49 4E 47 53 20 26 20 54 2E 20
0BB0 4F 27 42 52 49 45 4E 2E 0D 2D 2D 2D 2D 2D 2D 2D
0BC0 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D
0BD0 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 0D 57
0BE0 48 49 43 48 20 4D 4F 44 45 20 3F 20 28 53 2C 42
0BF0 2C 4E 29 20 0D 44 4F 20 59 4F 55 20 57 41 4E 54
```

```
0C00 20 57 48 49 54 45 20 3F 20 28 59 2C 4E 29 20 0D
0C10 3A 20 20 20 20 0D 4D 43 20 3A 20 00 00 2D 00 00
0C20 20 20 0D 49 4E 50 55 54 20 45 52 52 4F 52 2E 0D
0C30 43 48 45 43 4B 4D 41 54 45 20 2D 20 59 4F 55 20
0C40 57 49 4E 21 21 21 0D 50 4C 41 59 20 41 47 41 49
0C50 4E 20 3F 20 28 59 2C 4E 29 20 0D 54 48 41 4E 4B
0C60 53 20 46 4F 52 20 54 48 45 20 47 41 4D 45 2E 2E
0C70 2E 20 4D 49 43 52 4F 43 48 45 53 53 2E 0D 59 4F
0C80 55 20 52 45 53 49 47 4E 45 44 20 2D 20 49 20 57
0C90 49 4E 21 21 21 0D 20 20 43 48 45 43 4B 4D 41 54
0CA0 45 20 2D 20 49 20 57 49 4E 21 21 21 21 0D 20 43
0CB0 48 45 43 4B 21 0D 2B 2D 2D 2D 2D 2D 2D 20 4D 49
0CC0 43 52 4F 43 48 45 53 53 20 2D 2D 2D 2D 2D 2D 2D
0CD0 2B 0D 21 20 20 20 20 20 20 20 20 20 20 20 20 20
0CE0 20 20 20 20 20 20 20 20 20 20 20 20 20 21 0D 2B 2D
0CF0 2D 2D 2D 2D 2D 20 43 48 41 4C 4C 45 4E 47 45 52
0D00 20 2D 2D 2D 2D 2D 2D 2D 2B 0D 00 00 00 00 00 00
0D10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D70 00 00 15 D7 1F D0 CF 09 00 20 07 80 0A 0D 28 03
0D80 8D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0D90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0DA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0DB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0DC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0DD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0DE0 00 00 00 00 00 00 DB 00 E6 80 CA E6 0D 78 D3 01
0DF0 C9 DB 00 E6 40 CA F1 0D DB 01 E6 7F 47 C9 00 00
0E00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0E90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0EA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0EB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0EC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0ED0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0EE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0EF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0F90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0FA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0FB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0FC0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0FD0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0FE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0FF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

APPENDIX F
TYPICAL OUTPUT FROM MICROCHESS

```
MICROCHESS                          (C) 1977.
WRITTEN BY: P. JENNINGS & T. O'BRIEN.
-------------------------------------
```

```
DO YOU WANT WHITE ? (Y,N) N
```

THE USER DECIDES TO PLAY BLACK.

```
+------ MICROCHESS -------+
! WR WN WB WK WQ WB WN WR !
!                         !
! WP WP WP WP WP WP WP WP !
!                         !
!    ::    ::    ::    :: !
!                         !
! ::    ::    ::    ::    !
!                         !
!    ::    ::    ::    :: !
!                         !
! ::    ::    ::    ::    !
!                         !
! BP BP BP BP BP BP BP BP !
!                         !
! BR BN BB BK BQ BB BN BR !
+------ CHALLENGER -------+
```

THE BOARD IS AUTOMATICALLY DISPLAYED AT THE BEGINNING OF THE GAME.

```
:    SPEED

WHICH MODE ? (S,B,N) S
:    GO
MC : 13-33
:    63-43
MC : 01-22
:    76-55
MC : 02-46
:    71-52
MC :  0-0
:    AUTO

:    52-33
```

THE USER WISHES TO SELECT THE SPEED OF PLAY.

HE SELECTS THE SUPERBLITZ MODE.

GO: CAUSES MICROCHESS TO MAKE A MOVE.

1	P-K4	P-K4
2	N-KB3	N-QB3
3	B-B4	N-B3
4	O-O	

THE USER TURNS ON THE AUTOMATIC DISPLAY FEATURE.

```
                    N X P
...
```

```
+------ MICROCHESS -------+
!    WK WR :: WQ WB WN WR !
!                         !
! WP WP WP    WP WP WP WP !
!                         !
!    :: WN ::    ::    :: !
!                         !
! ::    :: BN ::    ::    !
!                         !
!    ::    BP    :: WB :: !
!                         !
! ::    ::    :: BN ::    !
!                         !
! BP BP BP :: BP BP BP BP !
!                         !
! BR    BB BK BQ BB :: BR !
+------ CHALLENGER -------+
```

THE BOARD IS NOW DISPLAYED AFTER EVERY MOVE.

```
MC : 14-34
```

5 P-Q4

```
+------ MICROCHESS -------+
!    WK WR :: WQ WB WN WR !
!                         !
! WP WP WP    :: WP WP WP !
!                         !
!    :: WN ::    ::    :: !
!                         !
! ::    :: BN WP    ::    !
!                         !
!    ::    BP    :: WB :: !
!                         !
! ::    ::    :: BN ::    !
!                         !
! BP BP BP :: BP BP BP BP !
!                         !
! BR    BB BK BQ BB :: BR !
+------ CHALLENGER -------+
```

THE USER TURNS OFF THE AUTOMATIC DISPLAY.

```
:    NO DISPLAY

:    72-63
MC : 04-13
:    33-54
MC : 46-55
:    DISPLAY
```

	...	B-K2
6	Q-K2	N-Q3
7	B X N	

THE USER REQUESTS A BOARD DISPLAY.

```
+------ MICROCHESS -------+
!    WK WR ::    WB WN WR !
!                         !
! WP WP WP WQ :: WP WP WP !
!                         !
!    :: WN ::    ::    :: !
!                         !
! ::    ::    WP    ::    !
!                         !
!    ::    BP    ::    :: !
!                         !
! ::    ::    BN WB ::    !
!                         !
! BP BP BP BB BP BP BP BP !
!                         !
! BR    :: BK BQ BB :: BR !
+------ CHALLENGER -------+

:    EXCHANGE
```

THE USER EXCHANGES MEN WITH MICROCHESS.

```
:    DISPLAY
```

```
+------ MICROCHESS -------+
! BR :: BB BQ BK ::    BR !
!                         !
! BP BP BP BP BB BP BP BP !
!                         !
!    :: WB BN    ::    :: !
!                         !
! ::    ::    BP    ::    !
!                         !
!    ::    WP    ::    :: !
!                         !
! ::    ::    :: WN ::    !
!                         !
! WP WP WP :: WQ WP WP WP !
!                         !
! WR WN WB    :: WR WK    !
+------ CHALLENGER -------+
```

THE BOARD IS DISPLAYED WITH THE MEN EXCHANGED. MICROCHESS IS NOW BLACK, AND THE CHALLENGER IS WHITE.

```
:    EXCHANGE
:    AUTO

:    66-55
```

EXCHANGE BACK TO THE ORIGINAL POSITION.

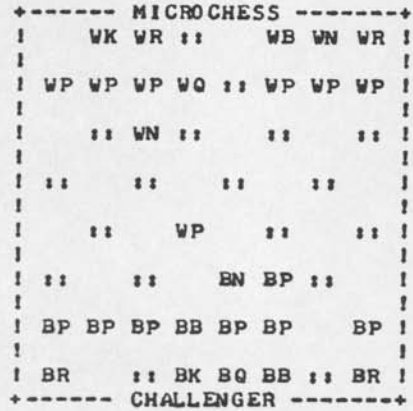
TURN ON THE AUTO DISPLAY.

```
...          P X B
```

```
+------ MICROCHESS -------+
!    WK WR ::    WB WN WR !
!                         !
! WP WP WP WQ :: WP WP WP !
!                         !
!    :: WN ::    ::    :: !
!                         !
! ::    ::    WP    ::    !
!                         !
!    ::    BP    ::    :: !
!                         !
! ::    ::    BN BP ::    !
!                         !
! BP BP BP BB BP BP    BP !
!                         !
! BR    :: BK BQ BB :: BR !
+------ CHALLENGER -------+
```

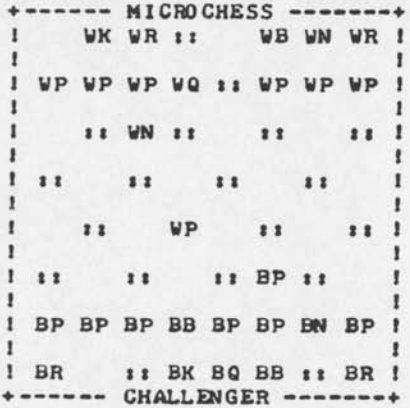
MC : 34-43

8 P X P



: 54-66

... N-N2



MC : 06-25

9 N-QB3

```
+------- MICROCHESS --------+
!    WK WR ::    WB    WR  !
!                           !
! WP WP WP WQ :: WP WP WP  !
!                           !
!    :: WN ::    WN    ::  !
!                           !
! ::    ::    ::    ::     !
!                           !
!    ::    WP    ::    ::  !
!                           !
! ::    ::    :: BP ::     !
!                           !
! BP BP BP BB BP BP BN BP  !
!                           !
! BR    :: BK BQ BB :: BR  !
+------- CHALLENGER --------+
```

: RESIGN

... RESIGNS

YOU RESIGNED - I WIN!!!

```
+------- MICROCHESS --------+
!    WK WR ::    WB    WR  !
!                           !
! WP WP WP WQ :: WP WP WP  !
!                           !
!    :: WN ::    WN    ::  !
!                           !
! ::    ::    ::    ::     !
!                           !
!    ::    WP    ::    ::  !
!                           !
! ::    ::    :: BP ::     !
!                           !
! BP BP BP BB BP BP BN BP  !
!                           !
! BR    :: BK BQ BB :: BR  !
+------- CHALLENGER --------+
```

THE GAME IS OVER.

PLAY AGAIN ? (Y,N) N

THANKS FOR THE GAME... MICROCHESS.

Micro-Ware Limited
27 Firstbrooke Road
Toronto Ontario
Canada M4E 2L2

Printed in Canada

KIM Cassette Loading Instructions

1 Enter (RS) to reset KIM.
2 Enter (AD) 0 0 F 1 (DA) 0 0 to reset decimal flag
3 Enter (AD) 1 7 F 9 (DA) C 1 to enter tape ID.
4 Enter (AD) 1 8 7 3 (GO) to begin read routine.
5 Start your cassette player.
6 When you see: 0000 D8 stop your cassette player.
7 Enter (RS) (AD) 1 8 7 3 (GO) to read block 2.
8 Start your cassette player.
9 When you see: 0000 D8 stop your cassette player.
10 Enter (RS) (GO) to start program execution.

If you wish KIM to play a specific opening, enter the ID in address 17F9 and load the opening data. Enter (RS) before and after each tape load.

Data for Openings

Microchess plays white	black	Opening
A0	A1	Four Knights
A2	A3	French Defence
A4	A5	Ruy Lopez
A6	A7	Queen's Indian
A8	A9	Guioco Piano

Remember to always press (RS) between each tape load. Otherwise, data at 0100 and 0101 may be overwritten by the stack. Verify these locations against the program listing if you have trouble executing the program.

A second copy of the two main programs can be found after the openings.

M I C R O C H E S S

KIM Cassette Loading Instructions

1 Enter (RS) to reset KIM.
2 Enter (AD) 0 0 F 1 (DA) 0 0 to reset decimal flag.
3 Enter (AD) 1 7 F 9 (DA) C 1 to enter tape ID.
4 Enter (AD) 1 8 7 3 (GO) to begin read routine.
5 Start your cassette player.
6 When you see: 0000 D8 stop your cassette player.
7 Enter (RS) (AD) 1 8 7 3 (GO) to read block 2.
8 Start your cassette player.
9 When you see: 0000 D8 stop your cassette player.
10 Enter (RS) (GO) to start program execution.

If you wish KIM to play a specific opening, enter the ID in address 17F9 and load the opening data. Enter (RS) before and after each tape load.

Data for Openings

Microchess plays white	black	Opening
A0	A1	Four Knights
A2	A3	French Defence
A4	A5	Ruy Lopez
A6	A7	Queen's Indian
A8	A9	Guioco Piano

Remember to always press (RS) between each tape load. Otherwise, data at 0100 and 0101 may be overwritten by the stack. Verify these locations against the program listing if you have trouble executing the program.

A second copy of the two main programs can be found after the openings.

COMPUTER CHESS NEWSLETTER

Issue #1 Price: 75¢

This first issue of 8 pages is being sent to about 100 who have requested it, plus about 200 who have not. Since I'm paying for this out of my own pocket, I hope that all recipients will send me 75¢. Since some of you have already sent me one or two 13¢ stamps, send only 49 or 62¢, of course. This issue is being sent First Class, because of the likelihood that some of the addresses may be out of date.

The next issue will be 16 pages, and will be sent Third Class, after replies from the first issue have updated the addresses. The price of the 2nd issue is also 75¢, paid in advance. Material for the 2nd and 3rd issues is already in hand, and they will be sent as soon as practical. I can't yet set a price for regular subscriptions, since I don't know how many subscribers there will be, and I don't know the rate at which letters and articles for publication will arrive. In the meantime, it's one issue at a time. I expect to print 16 pages per issue (optimum for Third Class postage).

This newsletter is patterned on Hal Singer's pioneer computer hobbyist "Micro-8 Newsletter", and, like it, will consist mainly of contributions from the readers. I placed an ad in ON-LINE, and wrote letters to the editors of the computer hobby magazines, and received about 90 requests for the newsletter so far, including much material for publication.

I urge you to contribute a letter or article for the newsletter. If you have written a chess program, tell us about it--its philosohy, its implementation. Tutorial articles are especially needed, as there are many beginners who need help to get started. Also welcome are news reports, book reviews, information about available programs, and records of interesting games played against a computer, or between computers. If you have played against a computer chess program, tell us what you think of the program--its strength, how it displays the board and moves, its speed, etc.

I am starting this newsletter because I couldn't get anyone else to. I hope that some dedicated organization will take it over--perhaps a computer science department (with lots of cheap student labor). There has been a great rise in interest in computer chess, and I feel that the time is ripe for a publication to serve as a means for exchange of information among writers of computer chess programs, and to provide information for those interested in playing computer chess.

Two chess-playing machines have been announced; one is now on the market for $200 or less. A chess program in BASIC is available for $6, one in 1100 bytes of machine language for computers using the MOS Technology 6502 microprocessor for $10, and one for $15 for computers using the 8080 microprocessor. Hobby computers are getting better and cheaper at a high rate, and for $300 or less you can have a KIM-1 and power supply to play chess on the 6502. Also, there is an annual U.S. Computer Chess Tournament, and others in Canada and Europe. And on campuses and elsewhere, an increasing number of people are writing computer chess programs.

Most chess programs are developed independently, and much effort is spent on re-invention. Perhaps by sharing information, comparing programs and program philosophies etc., the state of the art could be advanced more rapidly.

Access to the literature is important, and I expect to publish as complete a bibliography as possible, perhaps 2 pages per issue for 5 issues.

This issue presents the first of several articles by John Ford, author of a program for the Intel 8008 microprocessor, the first to be used in hobby computers. Since his machine had a 4k memory and interfaced with a magnetic tape cassette recorder, and his program occupied 8k, the first segment automatically called in the second from the cassette recorder.

Please send in your comments and criticisms of this issue, plus your suggestions for future issues. Especially, send material for publication.

One thing we hope to do is to establish the ratings of the various program-machine combinations, using the usual USCF system. Perhaps local tournaments can be organized, with machines competing against rated human players.

McGill University

School of Computer Science
Burnside Hall (514) 392-8275

22nd March 1977

Mr. Douglas L. Penrod
1445 La Cima Road
Santa Barbara
California 93101
U.S.A.

Dear Mr. Penrod,

I received your recent letter regarding setting up some sort of computer chess newsletter. I certainly would like to encourage you to do so. I am enclosing a mailing list for your information. I, myself, am involved in too many other activities at this time to give you much help.

I am not sure whether the ACM is appropriate or not. SIGART would probably be willing to some degree. I doubt there is presently enough interest to maintain an independent newsletter.

I wish you good luck in your efforts and I look forward to hearing from you again.

Sincerely,

Monroe Newborn
Associate Professor and
Acting Director.

Postal address: 805 Sherbrooke Street West, Montreal, PQ, Canada H3A 2K6

Thomas E. Doyle
5222 Big Bow Road
Madison, WI 53771

4/7/77

Doug; Very interested in the computer chess newsletter--I have a homebrew 8080 system w/20k of RAM and have been running Randy Millers chess program.

News-- delete Line 3001 from Millers chess program & it runs better (much better.)

-- Millers program has an error in line 8607

8607 PS = SGN (A-B): FOR PS = ...
↑ This should be a 2 not an S

Any idea on chess programs for 8080 or in BASIC sould be appreciated--

Thanks

Tom

© 1977 Bobby Fischer May 17 1977

Dear MR. PENROD, & think your computer-chess newsletter is a very good idea. & recently played some games on a terminal with the Greenblatt program. ENCLOSED are three of them. & made the mistake of buying the "chess challenger". It's ridiculously weak - they really shouldn't have come out with it. they also made a botch of the keyboard so it's hard to follow the moves. Somehow they reversed the algebraic notation so that the files are numbered and the ranks are lettered, if you can believe that! & KNOW & can give it a queen and a rook, because & gave them away IN the opening and WON. But & can probably give it much more. IN the endgame it's almost impossible to lose to it. Provided you agree and acknowledge, that & have all the publication rights to this letter and the enclosed game scores You can publish them in your newsletter. Regards Bobby Fischer

John Ford
5561 Esplanada
Santa Maria, Calif.
93454

Dear Doug,

In response to your request for material I will attempt to describe several rather basic aspects of my chess playing program. Actually, I believe that the following method of processing moves is fairly common but my own experience is not sufficient to warrant the assertion as fact. My program constructs a "move-tree" during the search for checkmate. This move tree is an ordered, identifiable (white or black) list of moves. The process of selecting a "best-move" in the absence of checkmate is a byproduct of the checkmate search. It involves some very complex scoring techniques which are, in my program, still primitive. Perhaps some of your other readers would care to comment on the scoring technique.

The generation of the move tree involves the calculation of all possible moves for each man and the storing of these moves into suitable memory locations. In my program, initial processing consists of making the list of all possible white moves. Then, the list is evaluated in a complex loop. The first move in the list is made, a second list of all possible black moves in response is created and concatenated on the list of white moves. The first move in blacks list is made and a third list of all possible white moves in response is created and concatenated on the list of black moves. The first move in whites list is made and a fourth list of all possible black responses is made. This type of processing continues until a specified search depth is reached. Early in the game, when all the pieces are on the board, the search depth is kept at 2 (2 moves for white and 2 for black), but as pieces disappear, the search depth is dynamically extended. This is necessary for two reasons: memory is not infinite and CPU speed is very finite on an 8008!

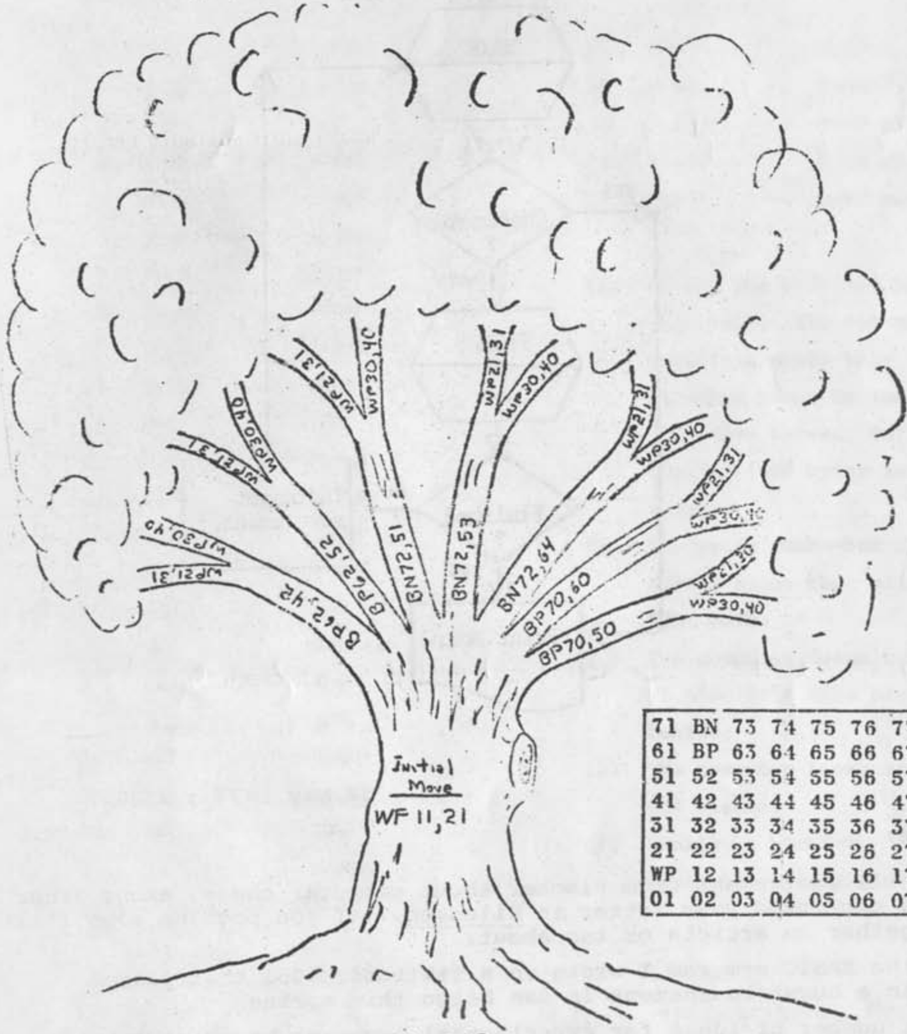
Having reached the end of the search depth, the position is evaluated for checkmate. In the absence of checkmate, the board position is reset and the next possible move in black's last list is made. Again, checkmate is evaluated. This processing is continued until black's list is exhausted. Then, the board position is reset to the white list second move and again, all black's responses are checked. When we have exhausted all of white's moves at the current search depth, the board is reset again and we start with black's second move at the next earlier search depth.

Finally, all possible moves for white and black to the required search depth will have been made and evaluated for checkmate. When checkmate is possible, processing halts and the board position is displayed. Normally however, checkmate is not available and the board position is reset to the starting position at the end of the processing. An array of best move choices, which was created during the checkmate search, is then checked for the highest score and that move is displayed.

Assuming anyone is interested, I will attempt to describe a method for identifying the board squares and making the individual piece moves in a later correspondence. And then, if interest continues, perhaps your readers might want a description of my primitive scoring technique and method for introducing book openings into the game. I've done a considerable amount of research in the display of the board for simple TV devices such as the hobbyist has at his disposal commonly, and would be happy to discuss several applications.

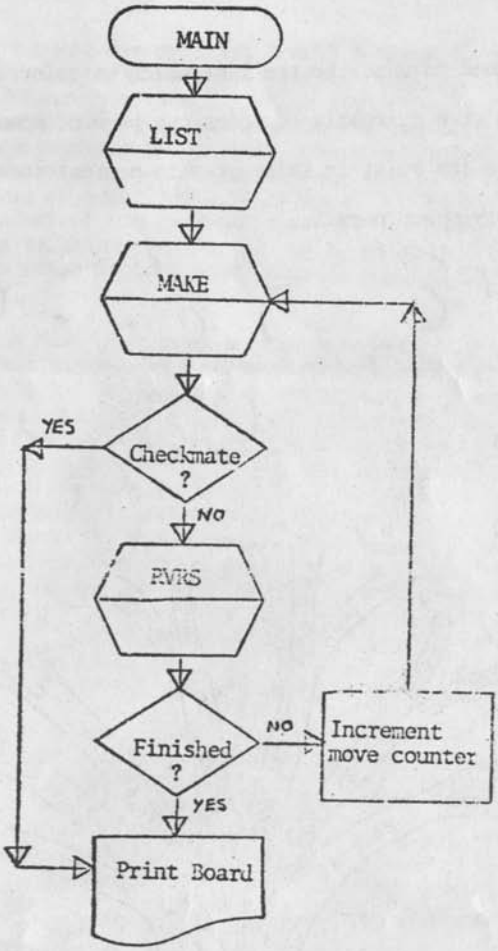
John Ford

We need to describe the list which we referred to as the "move tree". The move tree currently occupies one page of memory (page 57 in my system). There are built-in safeguards to prevent storing more moves than existing memory space permits.



71	BN	73	74	75	76	77	00
61	BP	63	64	65	66	67	BP
51	52	53	54	55	56	57	60
41	42	43	44	45	46	47	50
31	32	33	34	35	36	37	40
21	22	23	24	25	26	27	WP
WP	12	13	14	15	16	17	20
01	02	03	04	05	06	07	10

Move Tree



22 May 1977 ; 2230h.

Dear Doug;

I got a call last night from Fischer about computer chess, among other things he mentioned your letter in Kilobaud. If you prod me some I'll throw together an article or two about:

1) the BASIC program I wrote on a Tektronix 4051 that played in a human tournament in San Diego this spring

2) a number of ideas for experimental programs based somewhat differently than the Shannon tree-search procedure that all the other codes use.

3) a selling price for my functional programs (pretty cheap).

I'm looking around for somebody or company to pay me to write chess codes - actually just about anything with micros would be good. Any ideas?

Am currently working on an 8080 assembler code eventually based around my North Star minifloppy. This will make it compatible with most people - the 65xx and 68xx types may have to suffer for awhile.

Enclosed a copy of the computers best game, against a USCF rated "B" player. The machine (PATZER 451) was white.

regards

Tom

White - PATZER 451

Black - R. Schwartz (USCF 1670)

1.	P-K4	P-QB3
2.	N-KB3	P-Q4
3.	PxP	PxP
4.	P-Q4	B-N5
5.	B-K2	P-K3
6.	O-O	N-KB3
7.	P-B4	N-B3
8.	N-B3	R-B1
9.	B-N5	B-K2
10.	P-KR3	BxN
11.	BxB	PxP
12.	N-N5	Q-N3
13.	Q-R4	P-QR3
14.	N-R3	Q-B2
15.	QxBP (A)	P-N4
16.	Q-B2	NxP
17.	QxQ	NxB ch
18.	PxN (B)	RxQ
19.	R(B1)-B1	RxR
20.	RxR	K-Q2
21.	R-Q1 ch (C)	N-Q4
22.	BxB (D)	KxB
23.	R-Q4 (E)	R-QB1
24.	R-K4	P-B4
25.	R-K5 (F)	P-N5
26.	RxN	PxR
27.	N-N1	R-B8 ch
28.	K-N2	RxN
29.	P-N3	P-Q5
30.	P-B4	P-Q6
31.	P-B3	P-Q7
32.	K-B2	P-Q8(Q)
33.	P-KR4	R-N7 ch
34.	K-N3	Q-N8 ch
35.	K-R3	Q-R7 mate

(A) So far the computer has played very well. The subroutine which penalizes moves that undefend attacked pawns is inadequate, as this move shows. But I am core limited (400 bytes left) at the moment.

(B) The wrong rook--but this is a subtle point that will require much work.

(C) The computer doesn't realize that it shouldn't take pieces when behind.

(D) The computer loves to centralize its pieces.

(F) Finally a blunder. Pins are only partially understood.

N.B. this program has no lookahead in the usual sense. and no opening book.

The following Computer Chess books are available in English:
1) "Computers, Chess and Long-Range Planning", by M. M. Botvinnik, 1970, published by Springer-Verlag, New York.
2) "Computer Chess", by Monroe Newborn, 1975, published by Academic Press.
3) "Chess Skill In Man and Machine", edited by Peter W. Frey, 1977, published by Springer-Verlag. The following 3 books are published by Computer Science Press, 4566 Poe Avenue, Woodland Hills, California:
4) "Chess and Computers", by David Levy, 1975.
5) "1975 U.S. Computer Chess Championship", by David Levy, 1976.
6) "1976 U.S. Computer Chess Championship", by David Levy, 1977.

#2 has bibliographic references after each chapter, and #3 and #4 have an extensive bibliography in the back of the book.

The following chess programs are currently available for hobby computers:
1) A BASIC-language program by Randy Miller. A listing in MITS Altair 8k Basic, Version 3.2, with documentation explaining the program, plus a Tarbell cassette, all for $6, from Tarbell Electronics, 144 Miraleste Drive #106, Miraleste, California 90732.
2) MICROCHESS 8080 for computers using the 8080 microprocessor, at $15 for paper tape or Tarbell cassette, with documentation, and
3) MICROCHESS for the KIM-1 microcomputer, a listing with documentation, $10 from MICRO-WARE Ltd., 27 Firstbrooke Road, Toronto, Ontario, M4E 2L2, Canada.
4) A machine-language program for the PDP-8, $5 for paper tape and instructions, from John Youngquist, Verus Inst., Box 122, Fort Erie, Ontario, Canada.

A chess-playing machine--a dedicated 8080 microcomputer--is now available: The CHESS CHALLENGER, by Fidelity Electronics, sells for $200 or less. Russ McNiel got his at a discount from Markline Inc., 767 Main Street, Waltham, Massachusetts 02154, and is available in department stores in major cities. Russ, rated about 1450, enjoys the machine very much, although he can beat it unless he is careless or plays speed chess. He rates the machine at about 900; others I've talked to rate it at from 700 to 1100. High-ranking players tend to rate it lower than do ordinary players. I've heard only one player complain about the machine, and that was on the grounds of being too weak. Fidelity says you can trade your Challenger in on a stronger version in July, for $75 additional. -- Doug Penrod, editor

Here's a letter from James L. Purdie, 42 W Kenworth Rd, Columbus OH 43214:

"I read your letter in Kilobaud & am interested in hearing some more from others who are "into" computer chess. I have a Kim-1 with MICROCHESS that I have been experimenting with, mostly in the areas of tuning the value weights and eliminating evaluation of previously evaluated moves."

Doug Penrod, editor
COMPUTER CHESS NEWSLETTER
1445 La Cima Road
Santa Barbara
California 93101

PETER R. JENNINGS
MICRO WARE LTD.
27 FIRSTBROOKE ROAD
TORONTO, ONTARIO M4E 2L2
CANADA

FIRST CLASS MAIL

COMPUTER CHESS NEWSLETTER

Issue #2 **Price: 75¢, but $1.25 Overseas**

Issue #1 has been sent to 447, and Issue #2 has been requested by 136 computer chess fans so far, with requests for both arriving daily. I delayed preparation of Issue #2, hoping to be able to go to Toronto the the IFIPS meeting, where the 2nd World Computer Chess Tournament was held last month, but had to stay here for radiation treatment. I may, however, attend the ACM meeting in Seattle next month, where the annual U.S. Computer Chess Tournament will be held.

In Issue #1 I omitted two books from the list:

"The World Computer Chess Championship" by Jean Hayes and David Levy, 1976, and "Advances In Computer Chess", edited by Clarke, 1977. Both are published by Edinburgh University Press.

I must thank Don Cyr and Russ McNiel for their help.

Russ sent his Chess Challenger to the factory with $75, and it was returned with improvements including proper notation and 2 more levels of play; even the first level is improved. Tom Crispin and Dennis Cooper have played against it; I hope they'll have some comments in the next issue.

Another chess machine and another available program are mentioned in this issue. I hope that available machines and programs will be entered in enough (human) tournaments to get USCF ratings.

John Ford, Tom Crispin, and Dennis Cooper (all within my reach) have promised articles for future issues, as have several qualified correspondents, so help for beginners should be on the way. In the meantime the listed books and the material listed in their bibliogaphies should provide enough to keep one busy. -- Doug Penrod, editor

Paul Copeland
2 Stephen Crescent
Croydon Victoria
Australia 3136

Dear Mr. Penrod,

I read about your computer chess newsletter in Kilobaud. Enclosed is a cheque to cover costs of the first newsletter.

Although I do not have a microcomputer, I am very keen to obtain a chess program available in Basic. Perhaps you could advise.

I have never played chess against a computer, and what I have to say regarding programs is what I would like to see in an utopian chess playing maching. Suggestions-----

1. The program would be written in such a way that the computer responses to moves are never identical. At each computer move, the computer should be able to randomly choose between two 'best' moves.
2. For endgame study, the player should be able to enter an end game position into the computer. Play would commence from the position entered.
3. It would be useful if the player could at any time ask the computer for an analysis of the next four or five moves with variations. To this request, the computer would respond with black and white's next four moves or so.
4. The computer could be programmed to play against itself (with minimal time delay between moves). This would or should result in drawn games.
5. After checkmate, or a drawn game, the computer should be able to go through the game again pointing out to the player the good and bad moves that were played.

You may wish to mention some of these points in the next issue.

The unidentified "Tom", author of a letter printed in the first issue and of the program Patzer 451, is Tom Crispin, now at P.O. Box 1055, Goleta, CA 93017. Here are excerpts from another letter from Tom:

My experience with Bobby Fischer's Chess Challenger is that it should be considered weaker than 1000, perhaps as low as 700. It falls for almost any two-mover. (ed. note: Tom Crispin and Dennis Cooper have played against Russ McNiel's upgraded Chess Challenger and agree that it is much stronger).

I'll be writing an article or two on BASIC chess programs. However, for a program as complex as chess - complex to make it play well - BASIC as an interpretive language is much too slow. As I mentioned, it currently plays at about 2 min. a move, but a FORTRAN version on the same microprocessor should be 10-50 times faster.

My own preference is to sell listings of the program directly to micro-owners. It seems stupid to me to let large companies sell games packages to TV owners when for a little more $ the same owner could have a micro, with all the same games: but also a computer. I want the micro-industry to follow the pattern of hi-fi. Software should be sold much like LP's - if it isn't simply placed in the public domain.

I am hoping to collaborate with Bobby on the chess programming. I can provide the equipment and programming expertise; he could provide a somewhat better evaluation of the computer's play than I (though as I am rated somewhere near 2100 I'm not too bad in that department).

Part of my interest in the chess programs stems from my interest in AI. I want to work on either that aspect or games programming - or perhaps to develop a better BASIC with compiling options - it would be so useful in scientific work.

The limits of the Shannon approach (worth an article) have really only surfaced as a result of the past work. There is an article in the book Advances In Computer Chess (pub. by Edinburg Univ, edited by Clarke, 1977, $9.66) by the KAISSA programmers discussing better methods of tree pruning. My interest is to write a program that does no tree search. The motivation comes from the observation (this is yet another article!) that masters and grandmasters play rather good quality chess at blitz tempo - if the tempo is fast enough we can be reasonably certain that they are not searching a tree. True, they often make tactical oversights, but some form of pattern recognition allows them to play three, four or longer move combinations. Also, their positional play is often exceptional in fast games. So - why not try to emulate that aspect of master play on the micros? Suppose that, purely with my "static" approach, I achieved 1490 level chess on an 8080. Put the program on a Cyber 176, add a tree search, and I bet we have a master level program.

The Greenblatt program played in a number of (human) tournaments and has a rating. I would not accept as definitive any computer/program rating that is not based on tournament play. The human player should have something at stake to avoid his experimenting to see what the computer will do.

Michael Klos
Box 1053
Cortland, New York 13045

Mr. Penrod.

Enclosed is a money order for $1.00 to help cover costs. I would prefer to pay a little extra for first class mail.

I have copies of almost every article on computer chess, and some thesis papers. I will make photo copies of any for cost for any of your readers. The few I am missing I am trying to get. If someone has those I'd be glad to trade, etc.

If I can help you in any way please let me know.

C. Robert Leach *Systems Manager*
PETERBOROUGH NH 03458 (603) 924-3873

Dear Doug:

Congratulations on your first issue of the newsletter! Well done! I'm sure each succeeding issue will be better than the last.

Doug, I would appreciate your passing along the following information to the letter writer named "Tom" and to anyone else you think might be interested.

> The Kilobaud Software Library (soon to be initiated) is interested in locating high quality programs....systems programs....application programs.....CHESS programs...for all major hobbyist systems.
>
> We will pay programmers a royalty (quarterly) of 15% , and will distribute cassette tapes of the programs nationwide.
>
> Any interested parties should contact me at the Kilobaud Microcomputer Lab, 73 Pine St. Peterborough, New Hampshire 03458.

Thanks, Doug, and best of luck with your new venture!

Dear Doug,

I'm glad to see something started in a computer chess newsletter. As you know my goal was to be able to play a fair game of chess at home with a computer. Not knowing much about software or hardware it seemed nigh onto impossible. As a starter I've obtained two programs hoping that parts of them might be improved when more is learned. I haven't decided on hardware yet but in the meantime I purchased the "Chess Challenger" made by Fidelity Electronics at a good discount. It is supposed to beat an average player 25 to 75 percent of the time. After June they will add additional hardware for $75 to improve its game even more.

I can't be of any help to anyone already into this but maybe for someone just getting started I can offer some references on how to go about it. As we discussed one time, maybe if enough people worked on this problem someone will come up with a better approach to enable a computer to play better with less tree searching. As you know the "big" machines are examining 75 to 500-thousand positions per move and still not playing top chess. There has to be a better way so that hobby computers canplay an excellent game.

If anyone thinks I can be of any help or wants to trade information I will be glad to communicate with them.

Sincerely,

Russell McNiel
1343 La Manida
Carpinteria, Ca. 93013

Egbert Meissenburg D209 Winsen/Luhe, July 2, 1977
Ilmerweg 70 a
Bundesrepublik Deutschland

Dear Mr. Penrod:

I just received issue ## 1 of your COMPUTER CHESS NEWSLETTER. It is a very good idea to edit a fully speicalised journal for computer chess. I should like to receive all future issues of COMPUTER CHESS NEWS and all you publications on computer chess. I add $ 5.-- for subscription in banknote.

I miss your mentioning of Hayes/Levy Edinburgh 1976.

I offer you to compile a list of literature on computer chess from Russion sources as the material of Levy 1975 seems to be very incomplete.

Yours sincerely

Micro-Ware Ltd 27 FIRSTBROOKE ROAD, TORONTO, ONTARIO, CANADA. M4E 2L2.

Dear Doug,

Enclosed please find $3.00 for the first 4 issues or so of your newsletter. I look forward to reading some interesting articles and may contribute myself if time permits. I will be attending the World Computer Chess Championships and may find time to comment for you.

Your readers may be interested to know that MICROCHESS 2.0 is close to being completed. It will be initially available for the 6502, followed closely by a 6800 version, and then an 8080 version. It will offer a fairly sophisticated playing strategy as well as many of the frills that users of MICROCHESS have suggested in their letters to me. E.g. English notation, move validation, a set-up procedure, etc. The playing skill seems to be adequate to challenge the average club player. MICROCHESS 1.0 or the Chess Challenger are easily swiped by the program. I will be playing it against CHEKMO and COKO 3 in the near future and will let you know the results. If possible I will try to arrange other games with people (or computers) at the WCC in August.

Let me know if there is anything specific I can do to help you and your newsletter.

Sincerely,

Peter R. Jennings

SAMPLE GAMES PLAYED BY MICROCHESS

Human	MICROCHESS (Ruy Lopez)	
1. P-K4	P-K4	(1)
2. N-KB3	N-QB3	(1)
3. B-N5	N-B3	(1)
4. O-O	NxP	(1)
5. P-Q4	B-K2	(1)
6. Q-K2	N-Q3	(1)
7. BxN	NPxB	(1)
8. PxP	N-N2	(1)
9. N-B3	O-O	(1)
10. R-K1	B-N5	(37)
11. B-Q2	Q-K2	(46)
12. P-QR3	B-R4	(59)
13. P-N4	B-N3	(47)
14. N-K4	P-Q4	(60)
15. PxP (e.p.)	PxP	(68)
16. P-B4	B-KB4	(65)
17. P-QB5	PxP	(82)
18. NxP	Q-B2	(121)
19. Q-K7	QxQ	(170)
20. RxQ	BxN	(58)
21. PxB	NxP	(45)
22. R-QB1	N-N6	(50)
23. RxBP	KR-Q1	(50)
24. R-B7	K-B1	(53)
25. RxBP+	K-N1	(4)
26. RxB	P-KN3	(32)
27. R(5)-B7	R-Q6	(27)
28. B-N4	R-Q8+	(25)
29. N-K1	RxN+	(30)
30. BxR	R-K1	(14)
31. R-N7+	K-R1	(2)
32. R-R7+	K-N1	(2)
33. R(B7)-N7+	K-B1	(2)
34. B-N4+	N-B5	(2)
35. BxN+	R-K2	(2)
36. BxR+	K-K1	(1)
37. B-B5	P-R3	(3)
38. R-R8+ (mate)		

MICROCHESS (Queen's Indian)		Human
1. P-Q4	(1)	N-KB3
2. P-QB4	(1)	P-K3
3. N-KB3	(1)	P-QN3
4. P-KN3	(1)	B-N2
5. B-N2	(1)	B-K2
6. O-O	(1)	O-O
7. N-B3	(1)	N-K5
8. Q-B2	(1)	NxN
9. QxN	(1)	P-Q3
10. P-K4	(58)	BxP
11. R-K1	(55)	P-Q4
12. B-B4	(65)	N-Q2
13. PxP	(87)	PxP
14. Q-B6	(91)	N-KB3
15. N-N5	(122)	R-K1
16. B-R3	(158)	B-N5
17. R-K2	(198)	R-K2
18. R-B1	(134)	Q-KB1
19. R-R1	(139)	QR-K1
20. R-B1	(119)	B-Q6
21. RxR	(187)	RxR
22. R-Q1	(98)	R-K8+
23. RxR	(13)	BxR
24. BxP	(68)	P-N3
25. QxN	(65)	B-Q7
26. N-KB3	(63)	Q-N5
27. Q-Q8+	(61)	K-N2
28. B-K5+	(60)	K-R3
29. Q-R4+ (mate)	(60)	

(60) indicates the approximate time in seconds for the computer to make its move

The following is from a letter from Paul Burega, 1 Pleasant Bay, Winnipeg, Manitoba, Canada R2K 0C9:

I myself am in the process of writing a computer chess program. It is written in structured WATFIV FORTRAN and I am currently debugging what I have written on our local university's IBM 370-168. I currently have 600+ statements occupying about 30,000 bytes. At this point, all the program can do is input a human's move, check its legality, if it is a legal move, go through various update procedures and then generate all possible legal moves available for the computer. I am just starting to write the evaluation routine which will evaluate each move, give it a value, and then output the move with the highest value.

I hope that your newsletter will be a good one and I hope that I will be able to pick up some good pointers which I can use in my program.

CHARLES F. WILKES 6512 Rockhurst Road Washington, D.C. 20034

23 June 1977

Doug Penrod, editor
COMPUTER CHESS NEWSLETTER
1445 La Cima Road
Santa Barbara, CA 93101

Dear Doug:

Received your first newsletter, for which many thanks, and much encourgement for the future. I am enclosing $3.00 - please accept this in payment for the current issue, and as advance payment for the next three issues. I know you only asked for the next one issue in advance. But sending money one issue at a time is a pain. If in fact it turns out that there is no second, third or fourth issue, please accept the unearned balance as a donation for your investment in the first (and/or subsequent) issue(s).

I don't recall writing you - I write many people, so this is perhaps not as strange as it may sound. I also read Kilobaud, so may have, and then put it out of mind. It is also possible that you received my name from Monte Newborn, for a reason you will understand shortly.

The reason is that my son - Charlie - and I, are the joint authors of a computer chess program, which we named THE FOX. We competed with THE FOX in Atlanta at ACM three years ago, where we won against MIT in our first game, although by a time forfeit. You will find our game published in the SIGART bulletin of the time. We lost our second game to Newborn's OSTRICH by checkmate, both games on Sunday. We were playing on a timesharing system, which on Sunday was all ours. Unfortunately during gametime on Monday & Tuesday evenings, there were many other users of our system, and we lost both games by time forfeit. We think our game played very well, however, but have been unable to play in competition since due to the cost involved. We are continuing work on our program, and exercise it frequently.

We think our program is unique in many ways. For one thing, it is written in APL, a language which we feel is uniquely adapted for expression of the basic chess algorithm. For another thing, we wrote it entirely without knowledge of how others had to that point programmed. Due to our use of APL, we aren't even sure if it would have been any help had we known the details of other programs. We can recognize similiarities between our programs and descriptions of others. For example we use tree search, and our own equivalent of alpha-beta pruning.

For the record, I would like to detail how our game came to be. In the summer of 1972, my son and I both read George R.R.Martin's article in Analog magazine entitled THE COMPUTER IS A FISH. This concerned the earlier ACM computer vs. computer chess tournaments. My son was just entering the College of William and Mary in Williamsburg, Virginia, starting with the summer session. Since he had learned APL by going with me to my office on Saturdays over many years - I work for the IBM Corp. - Charlie already knew APL exceptionally well. William & Mary provide APL services to all students without charge. I challenged Charlie to write a Chess program, and that he did - using the College's machine. He came home Christmas with a crude but working version, which I promptly transferred to my own in-house IBM machine, which I was entitled to use.

Over the next few months, Charlie and I both worked on the program, and various people in my office played it. Since it won its very first game against the office chess expert, it got a lot of attention from our very active chess group of lunch-time players. By the time Charlie came home for the Spring break, we had an excellent program. Charlie had spent his time working on the guts of mid-game play. I had spent my time working on providing a sophisticated move acceptance routine, using descriptive notation, board display, etc., and in implimenting a book opening routine. Charlie had developed what we term a "programmed opening" as well, which we enter following departure from the book when a move is not found, and stay in until a threat of a certain level forces us to mid-game play. We stay in mid-game until the end, having no special end-game play (we would like to some day).

We signed up to compete at ACM at the end of the summer - 1973, representing the College of William & Mary, with the specific approval of its President. When Charlie came home for the summer, we both worked many many late late nights, all weekends, polishing the program. We obtained the assistance of a local chess master - Allan Savage, who was of great assistance. Through him we introduced many excellent strategies - i.e. Pawn strategies, etc., which greatly help the game. He played it the final game before the tournament, over a period of three nights. It was 62 moves to a checkmate for Savage of course, but he said the first 60 moves were a dead tie, until he passed a pawn. He then mated in two moves. Savage said he could not rate the program, but in a human would estimate between 1200 to 1600. The problem is that certain moves when made by a human, would indicate that a certain level of chess knowledge was present, which would apply in other situations as well. In the computer, however, "knowledge" is uneven. While it may make brilliant moves for a long sequence, it may then make an unforgiveable "dumb" move, which no human capable of making the good sequence would ever do. I think this is typical of most computer chess, except for the very very best.

There are many copies of THE FOX playing on IBM in-house APL systems, and periodically I get game print-outs which players want to show me. Our problem with chess in APL, is the tremendous demands made upon the host computer by the program. We played at ACM on a 370/145, using microcoded APL. When our program was in progress, we literally stopped the machine - we were in a 100% CPU bound execution mode, and of course could have used lots more speed. Even today, few machines execute APL any faster - I should know, since I am the individual within IBM who provides technical support for APL nation wide, within the data processing division of IBM. I am looking forward to faster machines - the 370/148 with microcode, and the 3033 when it is eventually available. I have a SPEED function which I use to measure machines for playing THE FOX, in order that I know it is capable of properly computing fast enough to make play reasonable. For example, I also have access to an IBM 5100 with APL, but this is far too slow to do anything but play through the book and programmed openings, coming to a complete halt with mid-game play.

Just a few details of THE FOX. We have a setting where we can alter the depth and width of the search. Our tournament level setting was as follows: Regardless of the setting, we make all possible moves for the first two half-moves - approximately 900 board positions. We then evaluate, and prune to the best three apparent moves, and make three more half-moves, for a total depth of five half moves. If we discover loss of material of major value, we may go wider than three, at the second half-move level. We are looking for an indication that each succesively position at the second half-move level (of the initial three)

generates a progressively worse evaluation at the fifth half-move. If it does, we make the assumption that an even wider search would not change this result. If we cannot prove this, we may go wider provided that our internal clock vs. move number indicates we have time to spare.

Obviously time is everything in a tournamet. We gain time points by our fast book at the opening, and our continuation of opening strategy via our programmed opening when we have to leave the book. We can move most mid-game moves in our allotted three minutes, except when we get in trouble - forcasting loss of material. Since we expect this to happen from time to time, we need the saved time from the opening moves in the bank to protect us much later in the mid-game, especially at the time call points.

Incidentally THE FOX can play either color, does play en passant of course - both allow & initiating such moves, and permits pawn promotion to any major piece value. It itself, however, will always promote to a Queen - the logic to test whether a lessor piece would be better was too much for the moment. The game can be started with any board setup. It can, in fact, play itself very nicely, flipping colors between moves. The capability of starting from a setup, and then play both colors, is most interesting to use in testing quality of moves from a given board position. All programs should have this ability for development purposes.

THE FOX was implemented in an APL workspace of 64K bytes. Today with much larger workspaces, this of course seems very small - we did struggle to keep it within this size, believe me. For example, the entire book is in the workspace initially, but only the moves for the one color to be played by the computer. Allan Savage personally selected the classical openings to be played by the computer for each color, omitting those openings which do not in the long run favor the player of the color being played by THE FOX. When the mid-game is entered, all code applicable to the opening is expunged, to provide room in memory for the expansion of the look-ahead three. The dynamic workspace storage management of APL is invaluable in this respect.

At my work at IBM these days, we have at least two fellow employees, who possess KIM-1 computers, exclusively for CHESS, using the available program. Of course they modify and improve the code as possible. The KIM-1 of course cannot begin to compete with THE FOX. I do think that the micro-computer is the future of computer chess however. Today I consider the Zilog and Mostek Z-80 to be the most appropriate CPU for the job - the availability of the bit instructions on the Z-80 are essential in my opinion. I know that sooner or later APL will also be available on the micro-computer. I don't know, however, if it will have the speed necessary to play satisfactory chess. I think there are strategies where multiple computers could work in consort to develop the move, and this may be the best approach.

In terms of development for THE FOX, we have one main strategy which we want to work on. The simple idea behind it really gripes us that we did not think of it earlier. This is to allow the computer to continue to "think" during the time the other computer is working out its move. Today we predict what we consider the best move for the other side to make is, as a function of generating our own move. We think we can take it, and allow the computer to make the assumption that this is the other sides move. If it is, we will signal through its entry at the proper

time that all the work to this point can be retained, and computation continue. If it is not, nothing is lost - merely return to the board position, make the actual move selected, and compute as we have done in the past.

My son Charlie has now graduated from William and Mary a year ago, and now has completed his first year as a medical student at the Univ. of Maryland school of medicine in Baltimore. He is deep at work in developing APL programs of interest in the field of medicine. I am certain that the development as an APL programmer which occurred with him in his work on THE FOX was so great, that he has great promise in this field in years to come. I know I am a much superior programmer today for the long hours of work on THE FOX - it taught me a great deal, particularly in how to write APL code for maximum performance on the computer, using minimum resources.

Thank you again for your efforts in behalf of Computer Chess. If you have the space in a future newsletter, you have my permission to use this letter, in the hopes that it may inspire some new developing author, as did George R.R.Martin's article in Analog for us.

Sincerely,

Charles F. Wilkes

IRA D. BAXTER
SOFTWARE DYNAMICS
17914 S. LAURELBROOK PLACE
CERRITOS, CALIF. 90701

DOUG PENROD
1445 LA CIMA ROAD
SANTA BARBARA, CAL. 93101

Sir:

This is to inform you that Software Dynamics has a Chess program written entirely in SD Compiler BASIC on a 6800 microcomputer, using a minimax tree lookahead scheme.

For BASIC, the program runs fairly quickly (blitz games are about 3 seconds a move). The program has a selectable lookahead minimum and maximum so that it can play from idiot-level chess to just plain mediocre (2 to 15 minutes a move). Lack of machine time prevents good to excellent play, but it is much better than all the other BASIC Chess games we have seen.

We would be happy to pit our 6800 running SD Chess against any other micro BASIC version of Chess.

If you have any questions, please call me at (213) 926-6492.

NORTHWESTERN UNIVERSITY
EVANSTON, ILLINOIS 60201

CRESAP LABORATORY OF
NEUROSCIENCE AND BEHAVIOR
2021 SHERIDAN ROAD
EVANSTON, ILLINOIS 60201

DEPARTMENT OF PSYCHOLOGY
312-492-7406

Mr. Douglas L. Penrod
1445 LaCima Road
Santa Barbara, California 93101

Dear Mr. Penrod:

Thank you for sending me the first issue of your newsletter. I hope your new venture is a successful one. I am enclosing a check for $3 for the first four issues. I have several observations on chess programming that may be of interest to your readers.

Past experience indicates that programs which play reasonable chess always involve a look-ahead search. In addition, they all have some scheme for assessing whether a terminal position is quiet or turbulent. If the position is not static, the search is continued. Programs which do not incorporate this strategy generally play a very poor game.

A reasonably comprehensive look-ahead search requires many, many CPU operations as well as considerable memory. It follows that a successful program should be written in assembly language to enhance the speed of execution or at least in a higher-level language for which a good complier is available such as Fortran. Writing in a language which uses an interpreter will generally increase the execution time for a deep search beyond ones patience or resources. A second implication of this line of reasoning is that individuals who wish to write a chess program on a personal computer should make every effort to develop extremely efficient code and should anticipate a need for at least 16K of RAM.

A second consideration is that much can be learned by spending a few hours in the library. There are certain references which are an absolute must. They include:

Shannon, C.E. Programming a computer for playing chess. Philosophical Magazine, 1950, 41, 256-275.

Berliner, H. Chess as problem solving: The development of a tactics analyzer. Unpublished doctoral thesis, Carnegie-Mellon University, 1974.

Knuth, D.E. and Moore, R. An analysis of alpha-beta pruning. Artificial Intelligence, 1975, 6, 293-326.

Slate, D.J. and Atkin, L.R. Chess 4.5 - The Northwestern University chess program. In Frey (ed.), Chess Skill in Man and Machine. New York: Springer-Verlag, 1977, pgs. 82-118.

The odds are about 100 to 1 that anyone who is writing a chess program and who has not read all of these references is probably wasting most of their time. A little reading can go a long, long way.

For those who like short cuts, a short Fortran program which incorporates a tree search was written several years ago by Jim Gillogly (Information Sciences Dept., The Rand Corp., 1700 Main St., Santa Monica, Calif. 90406). This program could probably be modified for home-brew systems. Another possibility for the game-playing enthusiast is to start with a more tractable game. An excellent starting point for an 8 or 16 bit CPU is the game of Hawaiian checkers (Konane). For a description of the rules, see an article (p. 5) by Joel Gyllenskog in the Feb., 1976 issue of the Sigart Newsletter (published by ACM).

Best regards,

Peter W. Frey

Peter W. Frey
Associate Professor

Dear Mr. Penrod,

Thank you for the copy of your newsletter. Enclosed is a small monetary contribution. I hope there is enough sustained interest to keep your publication going. When I opened it up, I wesn't sure what I'd find. I certainly didn't expect a letter from Bobby Fischer! That was quite interesting. I would like to have seen his games with the Greenblatt program.

There is a promising future for chess programs on micro-processors. Right now I think the field is held back by a lack of experience and know-how. People are still struggling with basic problems, like how to generate moves and grow trees, etc. I have had several inquires from computer-chess enthusiasts who would like to write a program but have little idea of where to start. Prospective chess-programmers should read the existing literature so they don't have to re-invent the wheel. This doesn't mean that they should try to emulate existing programs. There is room for lots of improvements and new ideas in chess programs for both little and big machines. To micro-processor chess programmers I also suggest :
1. That they equip their machine with a fair amount of memory, like 32k.
2. That they write modular, well-documented code.
3. That they carefully think out basic design issues, like the internal representation of moves, pieces, boards, trees, etc.

Some interesting recent games of Northwestern University's CHESS 4.5, which I co-authored with Larry Atkin, can be found in April, 1977 Sigart Newsletter, June, 1977 Chess Life and Review, and June, 1977 Scientific American.

Sincerely yours,

David J. Slate

David J. Slate

Stephen Stuart
2215 Rock St, Apt. 12
Mountain View, CA 94043

Dear Mr. Penrod:

I'd like to see a copy of your computer chess newsletter. Enclosed is my check for 75¢.

It might be of interest to you or your readers that there is some activity in this field where I work. An engineer at Amdahl Corp. (not myself) has put forward a challenge to the Amdahl Computer Club for microprocessor chess programs. The ground rules are only that the programs run in 2K (8 bits per word, presumably) or less, as the rumors say that the Fidelity Chess Challenger uses 2.5K.

So far I am the only person in the club with a working program, running on a Signetics 2650. The program (Revision 1) plays a little more poorly than the Chess Challenger, but I am currently at work on Revision 2, having learned a lot from my first shot at it. My club competitors are now only a few weeks away from their respective completions in some cases.

I brought my "toy" into work for a demonstration, which incidentally helped fire up interest in the tournament. The 2650 was soundly beaten by 90% of its human opponents over a 2-day period. It looks like once we can get a machine-vs.-machine game going, we will be seeing some low-level but interesting chess.

If you would like to hear more of this as it progresses, I would be glad to report on it.

This is from a letter from Arnold E. Jones, 2561 Alafaya Trail Apt. #118, Orlando, Florida 32807:

I am interested in your Chess special interest group, which you talked about in your letter published in Myte's Sept. issue. I think the idea of the Chess newsletter is a great idea. The Chess programs on the market today are being sold at high prices for the hobbyist.

I have the source to a good chess program in IBM 360 assembly language which I recently obtained. I would however like to get some source listings of a chess program in 8080 or Z-80 assembly. I have access to Zilog Development System, IBM 370/165, Varian 73, and a SOL. I am a freshman at Florida Tech. Univ. which has all the above Systems. I am planning to buy my own micro soon. I would appreciate any information you could send, and I can send a copy of the chess program to anybody who wants it.

John Griffin
1067 Enderby Way
Sunnyvale, CA 94087

Doug,

I received Issue #1 of the Newsletter recently. It looks like a good start, and I am looking forward to future issues.

I have been assembling notes and ideas for a computer chess program for a number of years now, and I am about to start coding. I have access to a Data General Eclipse and hope to achieve a system of some skill and sophistication. It is for this reason that I hope the newsletter will not be entirely devoted to chess on microcomputers.

Enclosed is a check for six more issues. Also enclosed is a copy of some material I recently received from USCF.

UNITED STATES CHESS FEDERATION
PUBLISHERS OF CHESS LIFE & REVIEW
186 ROUTE 9W • NEW WINDSOR, NEW YORK 12550

Dear Mr. Griffin,

Thank you for your letter of May 30th inquiring about computer programs in the USCF Rating Program. Enclosed you will find sheet of information which should help you greatly in pursuing your interest in this subject.

You'll be happy to know that our USCF rated computers are published right along with the humans. We list them as though the last name is Computer and the first name is the true name. Below is a partial list of the ratings you've requested. Most of them are provisional (that is, have played less than 25 games).

Computer, Blitz	1518
Computer, 4.5	1935
Computer, Dutchess	1351
Computer, Patzer 451	1006
Computer, Sneaky Pete	1209
Computer, Worcester	1295
Computer, Xenabor	1244
Computer, Zap	1057

HEATH
Schlumberger
HEATH COMPANY
BENTON HARBOR, MICHIGAN 49022
PHONE: 616-982-3344 TLX 72-9421

Dear Mr. Penrod:

I am enclosing a copy of our computer catalog as you requested. I hope you like what you see. And I hope these products live up to your expectations.

As for software, it comes with the computers only. It is not available separately to non-Heathkit computer owners. We do plan to offer applications software in the near future. We would like to do a chess program, but do not presently have the in-house expertise to do it. I'd like to find a program we could adapt to our machines and resell. Any suggestions?

Thanks for your interest in our products.

Sincerely,

Louis E. Frenzel (pr)

Louis E. Frenzel
Director, Computer Products

Thomas A. Fallis
P.O. Box 76242
Sandy Springs, GA 30328

Dear Mr. Penrod,

When I received the first issue of the Computer Chess Newsletter, I was surprised to learn how many other people were also interested in Computer Chess. While I do not own a "hobby" computer, I have been working on a design which should be easily adapted to micros and minis. My current project is to convert all known master games from the early nineteenth century to the present into a database. Once this is done, it will be available to anyone interested in analyzing actual play with computer generated play.

Enclosed is my check for $2.25 to cover the cost of the first three issues. I really enjoyed the first issue and hope the next issues will be even better. I would be interested in corresponding with other people with similar interests. I am currently using an IBM 360/30 with 128K bytes for most of my work. It would be beneficial to have the Newsletter provide a service to allow its readers to correspond with those who share or want to explore similar paths.

David Levy
104, Hamilton Terrace,
London, NW8 9UP
England

Dear Mr. Penrod,

Thank you for sending me the first copy of your newsletter. I am very pleased that at last someone has had the courage to start something like this.

I would very much like to receive the newsletter by airmail, and would be willing to write occasional articles, e.g. on the 2nd World Computer Championship in Toronto (August 7-10) and the ACM tournament in Seattle (October 17-19).

Whether or not you can find some individual or university to take over the running of the newsletter will probably depend on how many subscribers you get and how much people are willing to pay. So long as the income is sufficient to pay for mailing, printing and typing I would think that it should not be too difficult to find a permanent editor.

As to the matter of content, I think that it should be possible to persuade chess programmers to sent you short technical papers from time to time. Hopefully some of those who are on your list will do so without any special request from you. If you like I can try to get some support from the people who participate in Toronto and Seattle. Will you be able to attend either or both of these meetings?

Can we see the Fischer games in your next number?

This is from Donald S. Tork (WB6YJY), 3484 Hill Canyon Ave, Thousand Oaks, CA 91360:

Please add me to your list of computer fanatics. I have an Altair 8800 & disk and I'm in the process of writing an operating system. (I'm a programmer in real life). I have a USCF rating of 1457 and I'm interested in seeing hi-level designs for chess programs; especially the ones that participate in the ACM tourney. (If you can persuade the authors to write them)

The following is from a letter from Kevin McLoughlin, Caltech 1-58, Pasadena, CA 91126:

Howdy Computer Chess Newsletter

I just finished typing in, translating, and debugging Randy Miller's Altair Basic chess program for use on the PDP-10 system here, and have a number of comments, expletives deleted:

1) I was almost able to type it in translating as I went, but there was one section of code, 25 lines in 1-increment statement numbers, packed with multi-statement lines, which could only be rewritten from scratch. Our BASIC is bare-bones ANSI standard and doesn't permit such things as multi-statement lines, logical AND, OR, and NOT, ON-GOSUB, etc., with which Miller's program abounds. He should have at least spaced out his statement numbers a bit more, to allow for changes.

2) The programming itself is redundant and incompetent. Comment lines are absent where they are most needed; an entire subroutine is repeated statement for statement; blocks of code are repeated with only small, computable variations; the move generation and legality check routines are clumsy and slow. There is a statement inserted which bypasses the entire evaluation routine, which you are apparently supposed to delete if you want the program to play decent (non-random) chess. The board display routine must be intended for a 9200 baud line printer--it would take until next year to print the board on a TTY. Miller could have saved the core space or used it for a routine to accept P/K2-K4 type notation.

3) Once debugged, the program does indeed play chess, though it is a bad joke to have included a statement to make it easier. With this removed, the program plays at the level of a seven-year old who learned the game last month. Enclosed is a listing of a game it played against the Greenblatt chess program, which resides (protected against examination to the nth degree) on the system library here. As you can see, the Greenblatt program won in 15 moves, but the Miller program put up a good fight; its moves were mostly consistent, considering the one-ply lookahead. The evaluation routine could be rewritten to make better play possible.

4) A listing of the program, rewritten in ANSI standard BASIC (so it should run on most machines), renumbered, commented, relieved of some of its redundancies, is enclosed. You probably can't publish it because of copyright rules, but ask Randy Miller (or MITS, or whoever owns it) anyway; perhaps they will let you print it in a future issue. In this form it is a basic, usable, understandable, modifiable chess program, and might be of use to your readers.

5) The one thing the distributors of this program did right was to include with the program a report explaining the steps involved in writing a chess program, in general high-level language terms. With the problem defined and set before me, I intend as soon as I have the time and energy to write my own program. It will be written in PASCAL, a highly structured language, whose best feature is perhaps its near total lack of need for a GO TO statement (though it has one, for those who have not yet learned how to think). Once I have it working in basic form, I will translate it into BASIC and send you a copy (a process considerably easier than translating BASIC into PASCAL). Grunging through the Miller program did two good things for me; it showed me the advantages of good programming style, and it showed me, laid out step-by-step, exactly what is needed (and not needed) in a chess program.

The following is from North Star Chess, edited by Paul D. Shannon, Box 371, Osseo, Minnesota 55369. This was from Volume II, #II.

Computors in Chess
Editorial Opinion by Paul Shannon

Computors do not yet deserve to play tournament chess. Sportsmanship and fighting spirit simply do not compute. Playing a computor appears to be like a boxer fighting a wrestler; the aim is the same (to put the opponent down on the canvas) but the style is very different, and the result is novelty not true sport.

I was playing in the Minnesota Open when Chess 4.5 scored its remarkable 5-1 triumph. Most of its opponents 4 of 6 were formerly high rated competitors but had not played much for the past several years. Ronning and Fenner had not played at all for at least three years, while Stenberg and Armagost have played only once or twice each year recently. Quite frankly I think that just about any good A class player could have had a master performance against that group on that weekend.

One week later I was tournament director for the Minnesota Championship playoff. All of these players (1) were in practice, and (2) had prepared to face the machine. As a result, although the average rating of the opponents was about the same, the program scored only one win and one draw. The difference had to be one of the two factors above or more probably a combination.

Tournament promotors cannot do much about keeping their players in top form except to hold many events and hope that everyone will play in them. However, we can do something about the second factor, the computor's advantage of surprise.

When a player prepares to play in a Swiss System event, he anticipates human opponents with human weaknesses. Specifically, a good tactical player can prepare to force complicated variations and thus cause errors. Against a computor this same tactical player feels the opponent has "nerves of steel" and "x-ray eyes" which make it a true "superman" due to the ability to "see" all of the variations and never get frustrated. Thus preparation for humans, or practice against them will not help much against the machine.

Having watched several of the games of Chess 4.5, it seems that one must avoid tactics and play variations where the issue is in using superior planning to create long term weaknesses, which may then be exploited by further long term strategies and maneuvers. Since this is contrary to the wildly tactical style which I cultivate for 30/1 Swiss play, I'm not sure I could adapt to a positional style or the typical ten minute pairing notice of a weekend Swiss.

For this reason I feel it is unfair to play computors in Swiss System events except perhaps in a one round per day event (club tourney or the U.S. Open) where significant notice (several hours at least) can be given to the computor's opponents.

Another problem area for all concerned when a computor plays is finishing the game. Chess 4.5 was unable to resign or to accept a draw offer or offer one itself. Thus theoretically all games must be played on to mate or perhaps the 50 move rule, both of which it is good enough to avoid for hours. In the Minnesota Championship we had to resort to adjudicated draw in bishops-of-opposite-color ending. Elsewhere the operators have used their human judgement to make the decision, but neither procedure is good. It is not fair for a player to have to

continue and thus lose rest time before the next round when the result is obvious to all present.

The final question for this article is: Should computor Chess 4.5 have been allowed to play for a state championship title? At the risk of being accused of advocating discrimination against all machine intelligance, I have to disagree with that decision of our worth (?) state board. The publicity the event received had some small value as did the novelty of seeing it, but not enough to offset the cheapening a great sporting event, the Minnesota Championship. There is no way computor capabilities can be compared with human players except perhaps with a human of total recall, no fatigue, interchangeable brain cells and no emotions. It is not the mechanical moves which make chess great, it is the sport of mental battle with another human. The human spirit and sporting values are intangibles which no machine may ever comprehend.

Hopefully, various local and national groups will soon devise some specific policy statements on these issues. The very minimum statements should include (1) adequate notice of the pairing to the opponent (or the right to refuse the pairing!), (2) an agreement on a workable system of evaluating draw offers and a program for resignation by the machine when warranted, and (3) restriction on participation where titles or large cash prizes are at stake.

Of course this is just one opinion and we would be happy to print other comments or opposing viewpoints. Adress all correspondence to Editor, North Star Chess, Box 371, Osseo, MN 55369.

Robert Semko
8188 Sherwood Place
Riverside, Calif. 92504

Dear Doug,

Thanks for sending me a copy of your newsletter. I'm enclosing 2 dollars for the first issue and the second issue. Put me down for a subscription when they're available.

I need some advice. I'm a beginner with no hardware as of yet. I'm planning to buy an assembled system by the end of the year and I'd like to keep the price at about $1500. The programming will mainly be games, especially chess. I'd like to find out about graphics, where I can display the board and pieces directly on the CRT. I know a good chess program requires a lot of memory so I'd want the kind of hardware where I can add more memory as I can it.

I'm a quadriplegic, paralyzed from the neck down, and although I can handle a keyboard, any kind of kits are out of the question. If you have any ideas as to what kind of computer system will be the best for me, I'd appreciate hearing from you.

From Jonathon Steer, 21 Berkeley St., Nashua, NH 03060:

I was interested to read your letter in the June KILOBAUD as chess computing has been a hobby of mine for several years. I got interested in computers through chess and chess programming. As a USCF member, having a good program around to play against is great. Having helped write a program running on the Univ. of New Hampshire DEC-10, I appreciate all the information one can get. The present version, however, only plays in the 1000-1100 level with openings being the strongpoint.

I hope that this newsletter works out because up until now as you know only academic sources for help and information.

© 1977 Bobby Fischer

Greenblatt-White
Fischer-Black

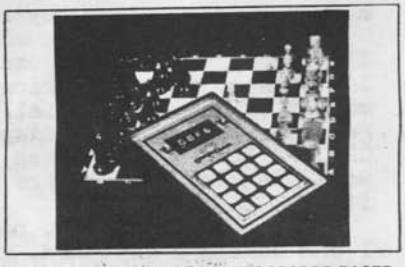
1.P-K4	P-QB4	21.QxQRP	N-K6
2.N-KB3	P-Q3	22.BxP	Q-N4
3.P-Q4	PxP	23.P-KN3	R-R1
4.NxP	N-KB3	24.B-R7	P-R4
5.N-QB3	P-QR3	25.Q-N7	P-R5
6.B-K2	P-K4	26.K-B2	PxP
7.N-QN3	B-K2	27.PxP	P-B4
8.B-K3	O-O	28.PxP	RxPch
9.Q-Q3	B-K3	29.K-K1	QR-KB1
10.O-O	QN-Q2	30.K-Q2	N-B5db1 ch
11.N-Q5	R-B1	31.K-B2	Q-N3
12.NxBch	QxN	32.Q-K4	N-Q3
13.P-KB3	P-Q4	33.Q-B6	R-B7
14.N-Q2	Q-N5	34.K-Q1	B-N5
15.N-QN3	PxP	35.BxR	Q-Q6ch
16.Q-Q1	N-Q4	36.K-B1	BxB
17.B-QR7	P-QN3	37.N-Q2	RxB
18.P-QB3	Q-K2	38.QxN/Q7	R-B8ch
19.PxP	N-K6	39.NxR	Q-Q8 Mate
20.Q-Q3	NxR		

Fischer-White
Greenblatt-Black

1.P-K4	P-K4
2.P-KB4	PxP
3.B-QB4	P-Q4
4.BxP	N-KB3
5.N-QB3	B-QN5
6.N-B3	O-O
7.O-O	NxB
8.NxN	B-Q3
9.P-Q4	P-KN4
10.NxKNP	QxN
11.P-K5	B-KR6
12.R-B2	BxKP
13.PxB	P-QB3
14.BxP	Q-N2
15.N-B6ch	K-R1
16.Q-R5	R-Q1
17.QxB	N-QR3
18.R-B3	Q-N3
19.R-QB1	K-N2
20.R-N3	R-KR1
21.Q-R6 MATE	

Greenblatt-White
Fischer-Black

1.P-K4	P-QB4	25.BxR	BxB
2.N-KB3	P-KN3	26.R-KB1	B-B4
3.P-Q4	B-N2	27.R-B2	P-KR4
4.N-QB3	PxP	28.R-K2	K-B2
5.NxP	N-QB3	29.R-K3	B-Q5
6.B-K3	N-KB3	30.R-KB3	K-K3
7.NxN	NPxN	31.P-B3	B-K4
8.P-K5	N-N1	32.R-K3	P-Q5
9.P-KB4	P-KB3	33.PxP	PxP
10.PxP	NxP	34.R-K1	P-Q6
11.B-QB4	P-Q4	35.P-KR4	P-Q7
12.B-K2	R-QN1	36.R-Q1	B-B6
13.P-QN3	N-N5	37.K-B2	B-KN5
14.B-Q4	P-K4	38.R-KR1	B-Q5ch
15.PxP	O-O	39.K-N2	K-Q4
16.BxN	Q-R5ch	40.P-R3	K-K5
17.P-KN3	QxB	41.R-KB1	K-Q6
18.QxQ	BxQ	42.K-R2	K-K7
19.R-KB1	RxR	43.K-N2	B-R6ch
20.KxR	P-QB4	44.KxB	KxR
21.B-B2	BxP	45.P-QN4	P-Q8(Q)
22.B-K1	R-KB1ch	46.K-R2	Q-K7ch
23.K-N2	R-B6	47.K-R3	Q-KN7 MATE
24.P-KR3	RxN		



COMPUCHESS I IS A MICROPROCESSOR BASED chess opponent with six selectable levels of intelligence, randomized strategy, programmable game situation capability, automatic or selectable pawn promotion, castling, and en pessant. Levels 1 and 2 are excellent teaching levels for children while 3 and higher levels are for practiced players. Level 5, for instance, will solve the most complicated of mate-in-two problems. Stock to 4 weeks at $159.95 from CompuChess, P.O. Box 65, Largo, Florida 33540.

The above appeared in a recent issue of Electronic Engineering Times

A number of Issue #1 were returned because the addressee had moved. If you know the current address of any, please let me know:

William B. Adams	Hyattsville, MD
Miki Alexander	Sperry-Univac, Cinnaminson, NJ
Luis Ayala	IBM, Mexico City
Jeffrey M. Bachman	Milwaukee, WI
Craig Barnes	UC, Berkeley, CA
David Barton	UC, Berkeley, CA
Steven Bellovin	Columbia U, New York City, NY
Victor Berman	Sperry-Univac, Cinnaminson, NJ
Paul E. Black	Cedar City, UT
W.W. Bledsoe	AI Lab, Cambridge, MA
Burton H. Bloom	Lexington, MA
Ted Brown	Boston, MA
Capt. Franklin Ceruti	Alexandria, VA
Anton Chernoff	Philadelphia, PA
Ben Cohen	Annapolis, MD
Robert W. Enden, Jr.	Omaha, NB
Charles H. Fisher	Highland Falls, NY
Jack Fox	Austin, TX
Lawrence Futrell	Georgia Tech, Atlanta, GA
John Gaydos	Moorestown, NJ
arry Goldstein	Belmont, MA
Stan Kugell	AI Lab, Cambridge, MA
Daniel S. Marcus	Flushing, NY
James P. O'Donnell	Forest Park, IL
C. J. Orton	U. of Ottawa, Ottawa, Ontario
Walter Perkins	Danville, CA
Robert Polansky	Boston, MA
Capt. Herbert Raymond	San Diego, CA
Prof. Helmut Richter	West Germany
Peter Rowe	UC, Berkeley, CA
Ira Ruben	Sperry-Univac, Cinnaminson, NJ
Aaron Samson	Chicago, IL
Robert Schuman	Winston-Salem, NC
Capt. Rolf Smith	Richard-Gebaur AFB, MS
Joel Stutman	Brooklyn, NY
William Toikka	Sperry-Univac, Cinnaminson, NJ
Bob Uliss	Winthrop, MA
John P. Walsh	Salem, NH
Albert Waltner	Huntsville, AL
David W. Zacharias	Fort Lauderdale, FL

As most of you have heard by now, Chess 4.6, by David Slate and Larry Atkin, running on the fast Cyber 176, won the World Computer Chess Tournament in Toronto last month. And although it was not matched against the former champion, Kaissa, by Michail Donskly and Vladimir Arlazarov of Moscow, an exhibition match was arranged, which Chess 4.6 won. I hope to print more about it next time, with an article by someone who attended the affair.

This issue is being mailed Third Class, except for overseas, which is being sent as Printed Matter by Air. The cost of air postage overseas results in a higher charge for the overseas subscribers.

You can tell how many more issues you will receive by the total amount which you have sent in thus far:

I expect to put out Issue #3 as soon as I am finished with this issue, so I'll be free to attend the ACM meeting if I'm otherwise able to.

-- Doug Penrod, editor

$

David Bryant
4371 Rigel Ave.
Lompoc, CA 93436

Hi Doug.

Sorry it took me so long to write, but between Digital Group and high school I'm kept pretty busy.

The chess playing program we have at school is written in PDP-8 assembly (I think he means machine code--ed.) by John Youngquist. It takes about a minute a move, but plays very good chess - (only a handful of students here at the high school can beat it!) I will have it play itself for a while and get you a copy of some of its interesting games.

I'm still trying to get that Basic chess program working on our PDP-8s. It looks like it will work on one of our compiler Basics except for two minor problems: This Basic doesn't know anything about line numbers bigger than 4095 or two-letter variables. I'm working on some Basic programs to re-sequence and convert multiple letter variables to regular Basic variables (i.e. XY to X1).

I'd like to compare the Basic and assembly chess programs, for both speed and strength, but I'm sure that assembly is the only way to make a good and fast chess program, at least on micros and minis.

I'd love to write a chess program myself, but I doubt if I'll have time for quite a while. I'll be writing Digital Group's disk monitor soon, and this summer I may write an RPG for the 8080. However, I might rewrite John Youngquist's PDP-8 assembly program for the 8080.

Excerpts from 2 letters from Daniel Grieser, 4326 Kenny Road, Columbus, Ohio 43220:

I have just gotten a Microchess (8080 version) up and running on my Z-80/Digital Group system. I'm planning to pit it against an HP-21MX minicomputer (at work) running an early version of MAC/6. .....

Our first computer vs. computer chess contest:

WHITE: HP-21MX minicomputer running Minitech (in Algol)
BLACK: Digital Group (Z-80) running 8080 Microchess
Result: Draw; White=47 minutes, Black=32 minutes

Doug Penrod, editor
COMPUTER CHESS NEWSLETTER
1445 La Cima Road
Santa Barbara
California 93101

P 3.00 B 1.50
PETER R. JENNINGS
MICRO WARE LTD.
27 FIRSTBROOKE ROAD
TORONTO, ONTARIO M4E 2L2
CANADA

THIRD CLASS MAIL

You have now received 2 issues, which cost $1.50 altogether.
Your "subscription" status is shown by the address label code.
The meaning of the code is shown by these 4 unrelated examples:

P10.00 means you have paid a total of $10.00 so far.
0 0.50 " you owe $0.50 more for the 2 issues.
B 4.00 " you have a balance of $4.00 toward future issues.
RR " Renew: To receive future issues, send 75¢ apiece.

file Chess

Join Canada's largest and most active Chess Club!

TORONTO CHESS CLUB

109 Vaughan Road, Toronto 656-9054

Hours: Monday - Friday 6:30 p.m. to midnight
Saturday - Sunday 1:00 p.m. to midnight

SPECIAL ACTIVITIES

Casual Chess anytime.

Tuesdays: Novice Tournament 7:45 - 11:00 p.m.
Open to players rated under 1500 or unrated
15 minutes per player per game (using chess clocks)
5 round "Swiss" system (all players play all five games)
50 cents entry fee -- money returned as prizes
An excellent introduction to "casual-tournament" chess and a great way to improve!

Sundays: Junior Tournament 1:00 - 4:30 p.m.
For players under 14 years of age (free cookies!)
5 round tournament, 20 minutes per player per game
Entry fee 25 cents; cash prizes and book prizes (this tournament is subsidized by the club)
Tournament Director: Erik Malmsten (call Erik at 699-2618 or at the club on Sunday afternoons for more information)

Mondays:
Wednesdays:
Fridays: Rapid Transit Tournament 7:45 - 11:00 p.m.
Entry fee 75 cents, 92% of entries returned as prizes
15 minutes per player per game -- 5 round Swiss system
All players will be rated in our club rating system

Thursdays: Speedchess Tournament 7:45 - 11:00 p.m.
6 double rounds (12 games) - 5 minutes per player per game
Entry fee 75 cents

Saturdays: Speed Tournament 1:45 - 5:30 p.m.
Entry fee $1.00
7 double rounds -- 5 minutes per player per game

MEMBERSHIP RATES		
Adult	$30 (year)	$18 (6 months)
Senior Citizen	20	12
Student (full-time)	20	12
Junior (under 20)	12	7
Junior (under 16)	6	
Junior (under 13)	FREE	

Visitors to the Club pay $1.25 per day (50 cents for juniors).

OCTOBER 1978 VOLUME 3, Number 10

$2.00 in USA
$2.40 in CANADA

BYTE

the small systems journal