MICROCHESS NOTES

MICROCHESS is now available on cassette tape. Each cassette costs $3.00 with the purchase of a MICROCHESS manual at the regular price. Cassettes available are 1200 baud CUTS (SOLOS), 300 baud Kansas City, POLY 88 (origin 2000H), and Tarbell.

Register yourself as a MICROCHESS user to receive notice of further patches and improvements. Please include details of the system you are running.

BUGS

Known bugs in the hex dump, and early paper tapes:
MICROCHESS pawn promotion: Change 03C7 from 85 to 86.
Challenger pawn promotion: Change 035F from 85 to 8C.
Autodisplay after promotion: Change 0392 from D7 to D4.
Autodisplay after castling: Change 042A from D7 to D4.
Autodisplay after castling: Change 0478 from D7 to D4.
Printed dump error from 027C: OF OF OF 83.
All copies shipped after May 1, 1977 are correct versions.

RESTART WITHOUT INITIALIZATION

If no capture has occurred, you can use the M suffix to move pieces back to the correct squares after an error, but if a piece must be replaced it is more difficult. Corrections may be made, or positions set up in the following manner. When the input prompt (:) appears, interrupt the program. Use your system monitor or front panel to put the appropriate data in the piece table (See Appendix C). Execute address 0072H.
MICROCHESS can be relocated using a relocation program such as the MACHINE CODE RELOCATOR by Leor Zolman in the July '77 issue of BYTE. First, move the entire program with relocation. Then, copy the tables over the new version. Finally, correct the LXI instructions which were mistakenly adjusted.

Pass 1. Relocate the entire program, adjusting all references.

a = 0000 Start of program.
b = OFFF End of program.
c = New start address. (E.g. 2000)
d = New start address. (E.g. 2000)
e = New end address. (E.g. 2FFF)
f = 01 Fix and move option.

Pass 2. Copy the tables into place without adjustment.

a = 09DD Start of tables.
b = 0DE5 End of tables.
c = New start address + 9DD (E.g. 29DD)
d = 0000 (Not used)
e = 0000 (Not used)
f = 02 Move block only option.

Fix data mistakenly adjusted by the relocator. The following locations will contain your start address. Change them to 00 00. (Your start address +) 0306, 086E, 0774, 08A1.

Thanks to F. Gerlach, P.O. Box 9305, Ft. Lauderdale, FL, 33310 for suggesting this relocator and the COMPUCOLOR patches below.

------------------

COMPUCOLOR users will have to use the above method to relocate MICROCHESS. Then add the following: To replace 
ctrl-X with ctrl-K, 0330: 0B; eliminate echo after input, 
0334: 00 00 00; modify input, 09DA: C3 00 0E, 0E00: 
CD 03 01 78 C9; modify output, 09D7: C3 05 0E and 
0E05: 47 cd 09 01 C9.

COORDINATES can be added to the board display by making the modifications shown at the right:
MICROCHESS FOR SOL SYSTEMS

MICROCHESS is available on CUTS cassette for SOL systems and CUTER and VDM users. The cassette is prepared for the SOLOS operating system. Each cassette costs $3.00 with the purchase of the MICROCHESS manual.

SOLOS patches to the original hex dump:
00D4 CD 00 0E, 0120 CD OB 0E, 01D7 C3 01 C0
0258 00 00 00, 028A 7F CD 20 0E CD 20 0E
09D7 C3 54 C0, 0DF1 DB FA E6 01 C2 F1 0D DB FC
0E00 3E 01 32 09 C8 CD 42 02 C3 E5 C0 CD 42 02 CD F4
0E10 CO CD CB C0 CD F4 C0 CD CB C0 CD 04 C1 C3 04 C1
0E20 E5 D5 C5 F5 CD 1C C1 CD 98 C0 F1 C1 D1 E1 C9
0330 7F CA 30 0E, 0E30 06 5F CD 19 C0 0D 2B C3 25 03

Use the DEL key to correct errors. DO NOT use the AUTO or NO AUTO commands.

CUTER patches to a SOLOS cassette. Thanks, to Barry Watzman, 2330 Millenium Ln., Reston, VA, 22091. (CUTER at F000H)

ODE6 CD 19 F0 C9, ODF1 DB 06 E6 01 C2 F1 0D DB 05 47 C9
01D7 C3 01 F0, 09D7 C3 77 F0, 0330 7F CA 30 0E
0E00 3E 01 32 09 F8 CD 42 02 C3 08 F1 CD 42 02 CD 1C
0E10 F1 CD EE F0 CD 1C F1 CD EE F0 CD 2C F1 C3 2C F1
0E20 E5 D5 C5 F5 CD 44 F1 CD BB F0 F1 C1 D1 E1 C9 00
0E30 06 5F CD 19 F0 0D 2B C3 25 03

Patches for CONSOL:

ODF1 DB FA E6 01 C2 F1 0D DB FC, 09D7 C3 4B C0
0258 00 00 00, 01D7 C3 B4 C1, 0120 CD OB 0E
00D4 CD 00 0E, 028A 7F CD 1D 0E CD 1D 0E
0E00 3E 01 32 01 C8 CD 42 02 C3 F3 C0 CD 42 02 CD E3
0E10 CO CD FA C0 CD E3 C0 CD FA C0 C3 04 C1 E5 D5 C5
0E20 F5 CD 21 C1 78 CD 87 C0 F1 C1 D1 E1 C9

0E40 20 20 20 20 20 30 20 20 31 20 20 32 20 20 33 20
0E50 20 34 20 20 35 20 20 35 20 20 37 0D 00 00 00 00
0E60 CD BF 09 04 78 E6 07 32 2F 0E 06 20 C3 BF 09 0E
0E70 06 20 CD BF 09 C3 BF 09 CD DA 01 CD 70 0E 21 B6
0E80 0C C3 DA 01 CD 70 0E C3 DA 01
024A 21 40 0E CD 78 0E 3A 2F 0E C6 30 47 CD 60 0E 00
025A 00
02B1 CD 84 0E
The paper tape version of MICROCHESS is in a binary format. Each frame represents one byte. The first non-blank frame is the data for address 0000. The last non-blank frame is the data for address 0DFD. The linear checksum for the entire tape is exactly equal to 00.

The following loader has been provided to load MICROCHESS. Manually load the bootstrap at FO00H. Load the tape into the reader. Execute from FO00H. Start the tape reader. If no checksum error occurs, the computer will HALT. If a checksum error occurs, the INTE light will light and the program will loop at address 0F33.

```
0F00  31 FF 0F
0F03  21 00 00
0F06  45
0F07  4D
0F08  11 00 0E
0F0B  CD 22 0F
0F0E  B9
0F0F  CA 0B 0F
0F12  77
0F13  80
0F14  47
0F15  23
0F16  1B
0F17  7A
0F18  B3
0F19  CA 2C 0F
0F1C  CD 22 0F
0F1F  C3 12 0F
0F22  DB 00
0F24  E6 40
0F26  CA 22 0F
0F29  DB 01
0F2B  C9
0F2C  AF
0F2D  B8
0F2E  C2 32 0F
0F31  76
0F32  FB
0F33  C3 33 0F
```

```
LOADR
LXI SP, OFFFH
LXI H, 0
MOV B, L
MOV C, L
LXI D, 0E00H
INTL
CALL INPUT
CMP C
JZ INTL
LOOP
MOV M, A
ADD B
MOV B, A
INX H
DCX D
MOV A, D
ORA E
JZ DONE
CALL INPUT
JMP LOOP
INPUT
IN 0
ANI 40H
JZ INPUT
IN 1
RET
DONE
XRA A
CMP B
JNZ ERROR
HLT
ERROR
EI
JMP ERROR+1
```
# TABLE OF CONTENTS

## PLAYER'S MANUAL

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOTATION</td>
<td>3</td>
</tr>
<tr>
<td>PROGRAM EXECUTION</td>
<td>4</td>
</tr>
<tr>
<td>MICROCHESS COMMANDS</td>
<td>4</td>
</tr>
<tr>
<td>THE DISPLAY COMMAND</td>
<td>5</td>
</tr>
<tr>
<td>THE GO COMMAND</td>
<td>5</td>
</tr>
<tr>
<td>THE SPEED COMMAND</td>
<td>6</td>
</tr>
<tr>
<td>THE RESIGN COMMAND</td>
<td>6</td>
</tr>
<tr>
<td>THE EXCHANGE COMMAND</td>
<td>6</td>
</tr>
<tr>
<td>ENTERING YOUR MOVE</td>
<td>7</td>
</tr>
<tr>
<td>SPECIAL MOVES</td>
<td>7</td>
</tr>
<tr>
<td>CASTLING</td>
<td>8</td>
</tr>
<tr>
<td>PAWN PROMOTION</td>
<td>8</td>
</tr>
<tr>
<td>EN PASSANT</td>
<td>8</td>
</tr>
<tr>
<td>THE COMPUTER MOVE</td>
<td>9</td>
</tr>
<tr>
<td>NOTES</td>
<td>9</td>
</tr>
</tbody>
</table>

## APPENDICES

<table>
<thead>
<tr>
<th>Appendix</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>THE PROGRAM</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>THE CONTROL AND INPUT/OUTPUT</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>MOVE GENERATION</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>DATA COLLECTION</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>STRATEGY</td>
<td>12</td>
</tr>
<tr>
<td>B</td>
<td>INPUT AND OUTPUT SUBROUTINES</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>DISPLAY OPTIONS</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>CRT DISPLAY</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>CUSTOM BOARD DISPLAY</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>PIECE ADDRESSES FOR BOARD DISPLAY</td>
<td>16</td>
</tr>
<tr>
<td>D</td>
<td>RETURN TO YOUR OPERATING SYSTEM</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>HEX DUMP OF MICROCHESS</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>TYPICAL OUTPUT FROM MICROCHESS</td>
<td>22</td>
</tr>
</tbody>
</table>
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:

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

The MICROCHESS program is executed from address 0000. 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

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

<table>
<thead>
<tr>
<th>ENTER</th>
<th>SPEED</th>
<th>TIME PER MOVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>SUPERBLITZ</td>
<td>5 TO 10 SECONDS</td>
</tr>
<tr>
<td>B</td>
<td>BLITZ</td>
<td>10 TO 30 SECONDS</td>
</tr>
<tr>
<td>N</td>
<td>NORMAL</td>
<td>30 TO 300 SECONDS</td>
</tr>
</tbody>
</table>

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 0-0 to castle on the king's side (short), and 0-0-0 to castle on the queen's side (long). The letter 0 is used, not the numeral Ø.

:0-0

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:

: 13-03=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 following information for use by the strategy algorithms.

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

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

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

CAPTURE ($\psi$). 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.
\[ \text{VALUE} = 4.00\psi_1 + 1.25\psi_2 + 0.75\alpha_1 + 0.75\alpha_1 + 0.25\mu_1 \]
\[ + 0.25\psi_3 - 3.00\alpha_1 - 2.00\alpha_1 - 1.25\psi_1 - 0.25\rho_0 \]
\[ - 0.25\alpha_0 - 0.25\psi_2 - 0.25\psi_1 - 0.25\mu_0 \]

(\(\hat{\cdot}\)) signifies the challenger's value.

\((n)\) subscript signifies the position at time \(n\).
\((t)\) signifies the current board position.

\[ \text{VALUE} = \text{VALUE} + 02 \text{ if a piece is moved from the back rank.} \]
\[ \text{VALUE} = \text{VALUE} + 02 \text{ if a piece is moved to the centre.} \]
\[ \text{VALUE} = \text{FF} \text{ if the challenger is checkmated.} \]

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.
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:

```
0DE6 4130 ************************************************************
0DE6 4140 * TELETYPTE INPUT/OUTPUT ROUTINES SUPPLIED. *
0DE6 4150 ***************************************************************
0DE6 4160 * OUTPUT ROUTINE *
0DE6 4170 ************************************************************
0DE6 DB 00
0DE8 E6 80
0DEA CA E6 0D
0DED 78
0DEE D3 01
0DF8 C9
0DF1
0DF1 DB 00
0DF3 E6 40
0DF5 CA F1 0D
0DF8 DB 01
0DFA E6 7F
0DFC 47
0DFD C9
0DE6

4180 TTYO IN 0
4190 ANI 80H
4200 JZ TTYO
4210 MOV A, B
4220 OUT 1
4230 RET
4240 ************************************************************
4250 * INPUT ROUTINE *
4260 ************************************************************
4270 TTYI IN 0
4280 ANI 40H
4290 JZ TTYI
4300 IN 1
4310 ANI 7FH
4320 MOV B, A
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.
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 $0120$ ($00 00 00$) with a call to the display subroutine (CD $42 02$) 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 $00D4$ with the same subroutine call (CD $42 02$).

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 ($00 00 00$) at address $0258$. This replaces the CD DA $01$ 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 $0242$ (CD AC $09$) 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 09ED. 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

<table>
<thead>
<tr>
<th>PIECE</th>
<th>MICROCHESS</th>
<th>CHALLENGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>King</td>
<td>09ED 03</td>
<td>09FD 73</td>
</tr>
<tr>
<td>Queen</td>
<td>09EE 04</td>
<td>09FE 74</td>
</tr>
<tr>
<td>King Rook</td>
<td>09EF 00</td>
<td>09FF 70</td>
</tr>
<tr>
<td>Queen Rook</td>
<td>09F0 07</td>
<td>0A00 77</td>
</tr>
<tr>
<td>King Bishop</td>
<td>09F1 02</td>
<td>0A01 72</td>
</tr>
<tr>
<td>Queen Bishop</td>
<td>09F2 05</td>
<td>0A02 75</td>
</tr>
<tr>
<td>King Knight</td>
<td>09F3 01</td>
<td>0A03 71</td>
</tr>
<tr>
<td>Queen Knight</td>
<td>09F4 06</td>
<td>0A04 76</td>
</tr>
<tr>
<td>KR Pawn</td>
<td>09F5 10</td>
<td>0A05 60</td>
</tr>
<tr>
<td>QR Pawn</td>
<td>09F6 17</td>
<td>0A06 67</td>
</tr>
<tr>
<td>KN Pawn</td>
<td>09F7 11</td>
<td>0A07 61</td>
</tr>
<tr>
<td>QN Pawn</td>
<td>09F8 16</td>
<td>0A08 66</td>
</tr>
<tr>
<td>KB Pawn</td>
<td>09F9 12</td>
<td>0A09 62</td>
</tr>
<tr>
<td>QB Pawn</td>
<td>09FA 15</td>
<td>0A0A 65</td>
</tr>
<tr>
<td>Q Pawn</td>
<td>09FB 14</td>
<td>0A0B 64</td>
</tr>
<tr>
<td>K Pawn</td>
<td>09FC 13</td>
<td>0A0C 63</td>
</tr>
</tbody>
</table>
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 01D7 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.
APPÉ_NÍX

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

DO YOU WANT WHITE ? (Y,N) N

+----- MICROCHESS ------+
| WR WN WB WK WQ WB WN WR |
| WP WP WP WP WP WP WP WP |
| I   I   I   I   I   I |
| I   I   I   I   I   I |
| I   I   I   I   I   I |
| I   I   I   I   I   I |
| BP BP BP BP BP BP BP BP |
| BR BN BB BK BQ BB BN BR |
+----- CHALLENGER ------+

THE USER DECIDES TO PLAY BLACK.

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: 61-22
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

THE BOARD IS NOW DISPLAYED AFTER EVERY MOVE.
THE USER TURNS OFF THE AUTOMATIC DISPLAY.

... B-K2

6 Q-K2 N-Q3

7 B X N

THE USER REQUESTS A BOARD DISPLAY.

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

EXCHANGE BACK TO THE ORIGINAL POSITION.

TURN ON THE AUTO DISPLAY.

... P X B
MC : 34-43

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

54-66

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

... N-N2
MICROCHESS

PLAY AGAIN ? (Y,N) N

THANKS FOR THE GAME... MICROCHESS.