First Steps in

Computer Chess

Programming

Kathe and Dan Spracklen 10832 Macouba Pl San Diego CA 92124

The fascination of chess gains a new dimension with microcomputer chess. No longer are the struggles confined to giant machines. With the advent of the Chess Mate, Chess Challenger, Boris, and Compuchess, as well as some custom software packages, the day of microcomputer chess has dawned. Writing a program to play chess on a small system is no small matter, though. Consider just for a start the challenge of meaningfully representing the board and its pieces in computer memory: there are 64 squares, 32 pieces, 6 piece types and 2 piece colors. Since the machine is a microcomputer, storage requirements must be kept to a minimum. Next comes the job of moving the pieces. Only when these first problems of piece representation and move generation have been solved can the chess programmer go on to consider strategy.



Sargon, a chess playing program we developed for Z-80 machines, solves the representation problem through the use of a board array. Move generation is accomplished through a network of routines diagrammed in figure 1. The functions of the routines are as follows:

GENMOV	Generate move routine. Generates the move set
	for all of the pieces of a
MPIECE	given color. Piece mover routine.
MITECE	Generates the move set
	for a given piece.
INCHK	Check routine.
	Determines whether or
	not the King is in check.
PATH	Path routine.
	Generates a single pos-
	sible move for a given
	piece along its current
	path of motion.
ADMOVE	Admove routine.
	Adds a move to the
	move list.
CASTLE	Castle routine.
	Determines whether
	castling is legal and
	adds it to the move list if it is.
ENPSNT	
LINFSINT	En passant routine.
	Tests for an en pas- sant pawn capture and
	adds it to the move
	lists if it is legal.
ATTACK	Attack routine.
ATTACK	Finds all the attackers
	on a given square.
ADJPTR	Adjust move list point-
	er.
	Links around the second
	move in a double move
	(ie: castle or en passant
	pawn capture).
ATKSAV	Attack save routine.
	Saves attacking piece
	value in the attack list
	and increments the



Figure 1: Block structure of the move generation routine of Sargon, the authors' chess playing program written for Z-80 assembler language.

and increments the attack count for that color piece. Pin check routine. Checks to see if an attacking piece is in the pinned piece list.

Several of the routines involved are multipurpose routines. Their involvement in move generation is incidental to a main function elsewhere in the move selection logic. The key routines in move generation are MPIECE, PATH, CASTLE and ENPSNT. Of these,

PNCK

110	111	112	113	114	115	116	117	118	119	6E	6F	70	71	72	73	74	75	76	77
100	101	102	103	104	105	106	107	108	109	64	65	66	67	68	69	6A	6B	6C	6D
90	91	92	93	94	95	96	97	98	99	5A	5B	5C	5D	5E	5F	60	61	62	63
80	81	82	83	84	85	86	87	88	89	50	51	52	53	54	55	56	57	58	59
70	71	72	73	74	75	76	77	78	79	46	47	48	49	4A	4B	4C	4D	4E	4F
60	61	62	63	64	65	66	67	68	69	3C	3D	3E	3F	40	41	42	43	44	45
50	51	52	53	54	55	56	57	58	59	32	33	34	35	36	37	38	39	3A	3B
40	41	42	43	44	45	46	47	48	49	28	29	2A	2B	2C	2D	2E	2F	30	31
30	31	32	33	34	35	36	37	38	39	1E	1F	20	21	22	23	24	25	26	27
20	21	22	23	24	25	26	27	28	29	14	15	16	17	18	19	1A	1B	10	1D
10	11	12	13	14	15	16	17	18	19	A	В	С	D	E	F	10	11	12	13
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9

Figure 2: Decimal (a) and hexadecimal (b) representations of the chessboard used in the Sargon program. Each square of the board is represented by a single byte in memory. Border squares are assigned a flag value of hexadecimal FF. The use of the border simplifies move generation, since it becomes easy to determine when a piece moves off the board.

MPIECE and PATH will be discussed here. The routines will be described in a language independent narrative. The Z-80 assembler code in which they are implemented will also be presented and exhaustively commented.

The chessboard in memory is an array of 120 bytes that can be visualized as in figure

2. Each square of the board is represented in memory by a single byte. Border bytes

The Board in Memory

board. If a board square is empty, it has the value 00. Thus the board set up for play would be as shown in figure 3.

Piece Mover Data Base

In order to generate moves for the pieces on the board, data must be maintained to describe the possibilities for each piece. This is accomplished through the use of three tables. Values for the tables are given in table 1.

are assigned a fla The border sin	ag value of hexadecimal FF. mplifies move generation, es easy to determine when f the board.		D	IREC	т	U d	lsed irect		eterr	nine over	1.200
			D	POIN	Т	D	irect	ion T	Table	Poin	nter.
The Pieces in Me	mory							to det		n in	the
	represented in memory by					d	irect	ion 1	table	for	
	a. The meaning and function						1.000	piece			
of the bits are as	follows:		D	COU	NT			ion 7	Table	:	
D:+ 7	and the state of t						ount	25			
Bit 7 —	color of the piece. 1 – Black					0.000	CT. CT. CT. CT.	120-22010-220	100	nine irecti	1 1 2 1 7
a station like	0 – White					0	f m	ovem	ent	for	any
Bit 6 —	not used.					gi	iven	piece			
Bit 5 –	not used.										
Bit 4 –	castle flag for Kings only.	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
	Set if the King has	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
	castled.	FF	84	82	83	85	86	83	82	84	FF
Bit 3 -	moved flag.	FF	81	81	81	81	81	81	81	81	FF
	Set if the piece has	FF	00	00	00	00	00	00	00	00	FF
	moved.	FF	00	00	00	00	00	00	00	00	FF
Bits 2-0 -	Piece type.	FF	00	00	00	00	00	00	00	00	FF
	1 Pawn	FF	00	00	00	00	00	00	00	00	FF
	2 Knight	FF	01	01	01	01	01	01	01	01	FF
	3 Bishop	FF	04 FF	02 FF	03 FF	05 FF	06 FF	03 FF	02 FF	04 FF	FF
	4 Rook	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
	1 HOON										

About the Authors

Dan and Kathe Spracklen are the creators of Sargon, the microcomputer chess program that won the microcomputer chess tournament at the 1978 West Coast Computer Faire. Dan Spracklen is a 13 year programming veteran. His experience ranges from scientific simulation programs to real time commercial applications. He is currently a senior applications analyst for Sperry-Univac. Kathe Spracklen is a graduate student in computer science at San Diego State University. An experienced tournament player, Kathe provided the chess background for Sargon.

Bit 3 -Bits 2-0 -5 Queen 6 King

The pieces in play occupy squares of the

Figure 3: Representation of the pieces on their home squares. Pieces are identified by means of unique byte values.

Listing 1: The Sargon move generation routine, written in Z-80 assembly language.

******************************** ; EQUATES ******************************** PAWN = 1 KNIGHT = 2 BISHOP = 3 ROOK = 4 QUEEN = 5 KING = 6 = 0 WHITE 80H BLACK = BLACK+PAWN BPAWN = ******* TABLES SECTION ******************************* START: .LOC START+80H TBASE = START+100H ***** DIRECT - DIRECTION TABLE ******************************* .-TBASE DIRECT = .BYTE + 09, + 11, -11, -09 .BYTE + 10,-10,+01,-01 .BYTE -21,-12,+08,+19 .BYTE + 21,+ 12,-08,-19 .BYTE + 10,+ 10,+ 11,+ 09 .BYTE -10,-10,-11,-09 ********************************** DPOINT-DIRECTION TABLE POINTER ****** DPOINT = .-TBASE .BYTE 20,16,8,0,4,0,0 ******** DCOUNT-DIRECTION TABLE COUNTER *********************************** DCOUNT = .-TBASE .BYTE 4,4,8,4,4,8,8 ******************************

BOARD - BOARD ARRAY ******* BOARD = -TBASE BOARDA .BLKB 120 ****************************** TABLE INDICES SECTION ****** START+0 .LOC WORD TBASE M1: WORD TBASE M2: WORD TBASE M3: .WORD TBASE M4: .WORD TBASE T1: .WORD TBASE T2:

Equate statements supply symbolic equivalents for the piece types and colors.

Start is the first address in Sargon and should lie on an even 256 byte page boundary.

Indexing in the Z-80 makes use of an address, contained in either the IX or IY index registers, plus a displacement. The displacement is a signed number +127 to -128. Thus a 256 byte area of memory centered on the index address is accessible. For this reason TBASE is placed in the middle of the tables section.

The value of "." is the current program counter. Direct is now the displacement of the direction table from the table base. So if the value of TBASE is loaded in the IY index register, "DIRECT(Y)" will reference the first element in the direction table.

Diagonal directions used for Bishop, Queen, and King. Rank and file directions used for Rook, Queen, and King. Knight move directions.

White pawn directions including two forward moves and two diagonal moves for captures. Black pawn directions.

Displacement from table base. Starting point in direction table. In the order BP, WP, N, B, R, Q, K.

Number of directions to use from table. In the same order as DPOINT.

The board array consists of 120 bytes in memory.

Uses the area of memory between START and START+80H. These indices are used to index into the various tables. Since TBASE is on an even boundary, its address is of the form XX00, where XX depends on the load address. The table address needed for a particular routine is formed by storing a one byte value in the 00 portion. Since addresses are stored in memory with the low order byte first, XX00 would be stored as 00XX. Then changing the 00 portion is simply a

.WORD TBASE

P3: .BYTE 0

T3:

ANI	87H	
CPI	BPAWN	

matter of storing a one byte value in the index.

Working storage area to hold the contents of the board array for a given square.

Gets the piece to be moved into register A. In GENMOV, the routine which calls MPIECE, the piece value in register A, had been exclusive ORed with COLOR, the color of the piece to determine whether or not to call MPIECE. Another exclusive OR restores the piece.

This clears all the flag bits and leaves just piece type and color.

Is it a Black pawn?

Listing 1, continued (Listing 1 is concluded on page 95):

Accessory)	JRNZ DCR	MP2 A	No-Skip. Decrement, making piece type a 0 for a Black pawn.
MP2:	ANI STA	7 T1	Clears color bit and leaves just the piece type. This is the first step in forming the index into DPOINT and DCOUNT. T1 contains the value of TBASE (XX00) stored
	LIVE	m 1	in low-high order (00XX). After storing the piece type $(0-6)$ in T1, it contains the address of TBASE + TYPE.
	LIYD	Т1	This operation loads the entire TBASE + TYPE address into the IY index register.
	MOV	B,DCOUNT(Y)	DCOUNT is the displacement from TBASE to the start of the direction count table. So DCOUNT + TBASE is the starting address of the direction count table. Then DCOUNT(Y) is: DCOUNT + CONTENTS IY Register
			= DCOUNT + TBASE + TYPE (0-6)
			= START OF TABLE + TYPE $(0-6)$
			This move instruction pulls the direction count for the given
	MOV	A,DPOINT(Y)	piece type and places it in register B. Similarly, this instruction pulls the direction table pointer for
	IVIOV	A,DFOINT(1)	the given piece type and places it in register A.
	STA	INDX2	The direction table pointer will be used to index into the
100	LIYD	INDX2	direction table.
MP5:	MOV LDA	C,DIRECT(Y) M1	Gets the direction and places it in register C. Gets the "from" position which was stored in M1 in GENMOV.
	STA	M2	Save in M2 to form the address of the current position.
MP10:	CALL	PATH	Generate a single move in the given direction.
	CPI	2	Did the moving piece encounter a piece of the same color,
	JRNC	MP15	or is new position off the board? Jump if yes to either question. No move to add to move list. Ready for new direction.
	ANA	A	Was the square moved to empty?
	EXAF		Save the answer to this question by swapping flag register
	LDA	T1	for alternate flag register. Get type of moving piece.
	CPI	PAWN+1	Is it a pawn?
	JRC	MP20	If so, jump to special pawn handling logic. PAWN+1 is equal to the number 2. A White pawn would be of type 1 while a Black pawn would have type set to 0. In either case the carry flag would be set upon a comparison to a value of 2.
	CALL	ADMOVE	Valid move, so add it to the move list.
	EXAF		Restore the answer to the empty square question.
	JRNZ	MP15	If it is not empty, go get ready for next direction. No further moves are possible in this direction.
	LDA	T1	Get piece type. Some pieces may only make one move in a given direction.
	CPI	KING	The King is such a piece. Is this piece a King?
	JRZ	MP15	If so, go get ready for a new direction.
	CPI	BISHOP	Compare piece type to a Bishop.
	JRNC	MP10	If piece type is bishop or greater (ie: Bishop, Rook, or Queen) go make another move in this same direction.
MP15:	INX	Y	Increment direction index for next direction in the direction table.
	DJNZ	MP5	Decrement the direction count (in register B). If count is not yet 0, go back and repeat this process for the new direc- tion. Otherwise all of the directions have been considered.
	LDA	Tl	Fetch piece type again.
	CPI	KING	Is it a King?
	CZ RET	CASTLE	If so, call castle to add it to the move list if legal. Return to GENMOV.
*****		N LOGIC**********	Return to GENNOV.
MP20:	MOV	A,B	Get the number of move directions left to consider. If this
		and the point of the short of the	is the first direction, register A=4.
	CDI	7	And there three directions left to look at?

Are there three directions left to look at?

JRC	MP35	A carry on this compare indicates a diagonal move. If so, branch to diagonal logic.
JRZ	MP30	Equality on this compare indicates a forward move of two squares.
		Branch to check for legality.
EXA	F	Otherwise this is a forward move of one square. Restore the answer to the empty square question.
JRN2	Z MP15	If the square is not empty, this is not a valid move. Go check the next direction.
LDA	M2	Get the "to" position of the move.
CPI	91	Is it on the last rank and therefore a promotion of a White pawn?
JRNC	MP25	If so, go set promotion flag.
CPI	29	Otherwise, is it on the first rank and therefore a promotion of a Black pawn?
JRNC	2 MP26	If no, skip setting flag.
MP25: LXI	H,P2	Load the address of the promotion flag.
SET	5,M	Set the flag (bit 5 of P2).

CP

Based Reason Alway Comp Enterp	s Choo uter	ose	the a on th micro sets o eleme routir	nctual mo ne indexir processor. of indices nts of th	es in Sarg ve generat ng capabilit For this are main e tables. I especially	ion rely ties of tl purpose tained to The piece	heavil he Z-8 severa o acces e move
 Quality Merc Lowest Prices NEW! No-Risk 		ntee	group	M1-M4 T1-T3	Working index into Working	o the boa	rd array
Take A Lo	ok At				index into	o directio	n coun
					value tab		
Our Latest Line From MICR				INDX1 INDX2	General Used for	working	
From MICK			Varia	oles and co	onstants use	ed in the	routin
	Credit	Cash	PATH	and MPIE	CE include	:	
ALL ASSEMBLED: METAFLOPPY DISK SUBSYSTEMS FOR THE	Card Price S-100 BUS.	Discount Price		PAWN KNIGHT		entificatio ce types	
053 Mod II Two-disk system vith 630,000 bytes (formatted) 043 Mod II One-disk system	1,774	1,706		BISHOP	= 3 three	ough us	e
with 315,000 bytes (formatted)	1,072	1,031		QUEEN		iate sta mbers ar	
METAFLOPPY ADD-ON STORAGE MODU 1023 Mod II One-disk 315,000 byte add-on storage module with enclosure and power sup-	LES			KING WHITE	= 6 dec = 0	imal.)	
ly. Requires daisy chain cable.	604	581	300	BLACK	= 80		
MACROFLOPPY DISK SUBSYSTEMS FOR T 1042 Mod I One-disk system	744	716	in an and an	BPAWN P1-P3	= Black + = Workin		hold t
with 143,000 bytes (formatted) MACROFLOPPY ADD-ON STORAGE MOI						ts of the	
1022 Mod I One-disk 143,000 byte add-on storage module	Section Contra	Contra Pilling	alesa Partis		array fo	or a given	square
with enclosure and power sup- bly. Requires daisy chain cable.	510	491					
DISKETTES 1081-05 Package of 5 Micropolis diskettes (5-¼″) for			DPOIN	т	(a)		
use with both Mod I and Mod II drives.	33	32	0	+09	+11	-11	-09
DAISY CHAIN CABLES AND ACCESSORIES 1083-02 Daisy chain interface cable B, with 3 connectors for	s - vi	stem, on	4 8 12 16	+10 -21 +21 +10	-10 -12 +12 +10	+01 +08 -08 +11	-0 +1 -19 +0
ise with 2 storage modules at- ached to controller.	33	32	20	-10	-10	-11	-09
1083-04 Daisy chain interface cable D, with 5 connectors for use with 4 storage modules at- tached to controller.	61	59	NORUSE MOM-U	- CF	(b)		
nething lanotosing-io	2910 (11		Piec	е Туре	DPOINT	D	COUNT
NO-RISK GUAR * Cancellation With No Obligation Days * Full Satisfaction Or You May Retu * Plus All Manufacturers' Factory O	If WE Don't Deli urn Product	iver In 90		ор	20 16 8 0 4		4 4 8 4 4

Shipping charges: \$10 per CPU on larger units; \$1.50 per kit. \$2.00 min. per order.

Delivery is stock to 30 days on most items. Shipment is immediate for payment by cashier's check, money order or charge card. Allow 3 weeks for personal checks to clear. N.Y. State residents add approp. sales tax. Availability, prices and specs may change without notice.



P. O. Box 71 Fayetteville, N.Y. 13066 **Phone (315) 637-6208** Today!

Queen 0 8 0 8 King

Table 1: Direction table (a) and direction table pointer and counter (b). In order to generate moves for the chess pieces, data describing the possibilities for each piece is kept in table 1a. Table 1b shows the direction table pointer, which tells where to start in the table for a given piece, and the direction table counter, which determines the number of directions of movement for a given piece.

Circle 70 on inquiry card.



Figure 4: Generating all the possible Knight moves from the Queen Bishop 3 (QB3) square. The Knight is piece type 2 (see text) and has a DPOINT (direction table pointer) value of 8 and a DCOUNT (direction table counter) value of 8 also. So in generating the Knight's moves, DIRECT+8 will be the starting point in the direction table, and 8 values will be used: -21, -12, +08, +19, +21, +12, -08 and -19. The Knight starts at White's QB3 square, which is square 43 (see figure 2a, decimal representation). Thus the first possible Knight move is 43 - 21 = 22 (QN1), and so on.

Sample Move Generation

Suppose a Knight occupies the QB3 square. A Knight is piece type 2 and has a DPOINT of 8 and a DCOUNT of 8 (see table 1b). So in generating the Knight's moves, DIRECT + 8 will be the starting point in the direction table and 8 values will be used. Those values are -21, -12, +08, +19, +21, +12, -08, and -19. The Knight starts at White's QB3, which is



- SWTPC 6800 Plug Compatible
- STATIC No refresh required
- Allows SWTPC 6800 expansion to 48K
- Low Power
- Uses Single +8 Volt Supply

The M-16A STATIC random access memory system, with a total storage capacity of 16834 words of 8 bits each, is switch selectable to any 4K starting address, and a hardware write protect switch is also included. The system's storage elements are 4K by 1 STATIC memory chips which store 4 times as much in only 12% more space than the low power 2102's. Typical access time is fast enough to work with a 6800 based

square 43 (see figure 2a, decimal representation). Thus the first possible Knight move is 43 - 21 = 22. Now 22 is QN1, so the first Knight move returns the Knight to its starting square. Figure 4 summarizes all possible Knight moves from QB3.

Move Generation-The Algorithms Explained

Move generation is controlled by GEN-MOV, which scans the board array and calls MPIECE for each piece encountered. computer operating at 2 MHz and all systems are factory tested at 2 MHz.

Send for FREE NEW **Computer Products Catalog** includes our complete line of 6800 Super Software products.



(213) 462-5652

Circle 320 on inquiry card.

Then MPIECE, the piece mover routine, generates all possible legal moves for that piece (moves that place the King in check are eliminated later in the program). The piece is brought in from memory. It is a one byte data value, as previously discussed, which contains piece type, flags and color. The flags are deleted from the piece before checking for type. Basic piece types are indicated by values from 1 to 6. Except for pawns, White and Black pieces move alike. So a special case is needed for the Black pawn. If the given piece is a Black pawn, the piece type is decremented, making it type 0.

The type of the piece, now one from 0 to 6, is used as an index into the DCOUNT, direction table count, and DPOINT, direction table pointer arrays. The values for the given piece are fetched. The direction table pointer is then used as an index into DIRECT, the direction table, and the first move direction is fetched. The "from" position of the piece is the square on which the piece currently stands. This "from" board index and the direction table value are passed as parameters to the routine PATH.

PATH generates the move indicated and returns a flag which describes the status of the "to" position of the piece. Flag values are:

- 0 "to" position is empty.
- 1 "to" position contains a piece of the opposite color.
- "to" position contains a piece 2 of the same color.
- "to" position is off the board. 3

PATH accomplishes its task by fetching the "from" position, adding the direction counter, and storing the result as the "to" position. It then uses the "to" position to form an index into the board array. The current contents of the square are fetched. If the square contains hexadecimal FF, it is off the board. The off board flag is set and control is returned to MPIECE.

If the square is on the board, the contents of the square are saved in memory location P2. The color and flag bits are then cleared and the remaining piece type is saved in T2. If the square is empty, control is returned to MPIECE with the flag value still set to 0. Otherwise the color of the piece on the "to" square is compared with that of the moving piece. The appropriate flag is set to indicate whether or not the pieces are of the same color, and control is returned to MPIECE.

Upon return from PATH, piece mover



program development by **Peter Jennings**, author of the famous 1K byte chess program for the KIM-1. MICROCHESS 2.0 for 8K PETs TRS-80 without TBUG). MICROCHESS checks every move for and 16K APPLEs, in 6502 machine language, offers 8 levels of play legality and displays the current position on a graphic chessboard. to suit everyone from the beginner learning chess to the serious player. It examines positions as many as 6 moves ahead, and includes a chess clock for tournament play. MICROCHESS 1.5 for BRIDGE CHALLENGER by George Duisman for 8K PETs, Level II 16K TRS-80s, and 16K APPLEs: You and the dummy play 4 person Contract Bridge against the computer. The program will deal hands at random or according to your criterion for high card points. You can review tricks, swap sides or replay hands when the cards are known. No longer do you need 4 people to play! \$14.95 ORDERS: Check, money order or VISA/Master Charge accepted; programs and cassettes guaranteed. If you have questions, please call us at 617-783-0694. If you know what you want and have your VISA/MC card ready, you can DIAL TOLL FREE 1-800-325-6400

Level I and Level II versions are included and can be loaded on any You can play White or Black, set up and play from special board positions, or even watch the computer play against itself! Available now at a special introductory price of only \$19.95 STIMULATING SIMULATIONS by Dr. C.W. Engel for 8K PETs, 4K Level I and II TRS-80s, and APPLEs with Applesoft II: Ten original simulation games such as Forest Fire, Lost Treasure, Gone Fishing and Diamond Thief, progressing from elementary to guite complex with most suitable for schoolchildren. Includes a 64 page book giving flowcharts, listings and suggested modifications ... \$14.95 (24 hours, 7 days; in Missouri, dial 1-800-342-6600). Or you can mail your order to the address below. Personal Software" products are now AVAILABLE NATIONWIDE FROM COMPUTER STORES. Look for the Personal Software[™] display in your local store!

Personal Software[™] Cambridge, MA 02138 P.O. Box 136-B10

Circle 302 on inquiry card.

94 October 1978 © BYTE Publications Inc checks to see if the square is occupied by a piece of the same color or is off the board. If so, this cannot be a legal move, so a check for further moves must follow a new direction. Otherwise a check is made to see if the square is empty. The answer is saved. A check is made to see if the piece being moved is a pawn. If so, control is passed to the special pawn logic. Otherwise the move generated must be added to the move list. ADMOVE is called for the job. After the move has been added to the move list,

Listing 1, continued:

MP26:	CALL	ADMOVE
	INX	Y
	DCR	В
	LXI	H,P1
	BIT	3,M
	JRZ	MP10
	JMP	MP15
		F 2 SQUARES*******
MP30:	EXAF	MP15
MP31:	CALL	ADMOVE
	JMP	MP15
		NAL MOVE********
MP35:	EXAF	
	JRZ	MP36
	LDA	M2
	CPI	91
	JRNC	MP37
	CPI	29
	1000 A 1000	1110
	JRNC	MP31
MP37:	LXI	H,P2
	SET	5,M
	JMPR	MP31
		SQUARE EMPTY *****
MP36:	CALL	ENPSNT
	IMP	MP15
******	JMP *******	MP15 ************************************
PATH R	OUTINE	**********************
PATH R	OUTINE	
PATH R ******	OUTINE ********* LXI	**************************************
PATH R ******	OUTINE CUTINE LXI MOV	**************************************
PATH R ******	OUTINE ********* LXI	**************************************
PATH R ******	OUTINE CUTINE LXI MOV ADD	**************************************
PATH R ******	OUTINE CUTINE LXI MOV ADD MOV	**************************************
PATH R ******	MOV LIXD	**************************************
PATH R ******	MOV LIXD MOV	**************************************
PATH R ******	MOV LIXD MOV CPI	**************************************
PATH R ******	MOV LIXD MOV LIXD MOV LIXD MOV	**************************************
PATH R ******	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA	**************************************
PATH R ******	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA ANI	**************************************
PATH R ******	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA	**************************************
PATH R ******	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA ANI STA RZ	**************************************
PATH R ******	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA ANI STA RZ LDA	**************************************
PATH R ******	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA ANI STA RZ LDA LXI	**************************************
PATH R ******* PATH:	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA ANI STA RZ LDA LXI XRA	**************************************
PATH R ******* PATH:	MOV LIXD MOV LIXD MOV LIXD MOV CPI JRZ STA ANI STA RZ LDA LXI	**************************************
PATH R ******* PATH:	OUTINE CUTINE	**************************************
PATH R ******* PATH:	OUTINE CUTINE	**************************************
PATH R ******* PATH:	OUTINE CUTINE CUTINE CUTINE CUTINE CUTINE CUTINE MOV ADD MOV LIXD MOV CPI JRZ STA ANI STA RZ LDA LXI XRA BIT JRZ MVI	**************************************
PATH R ******* PATH:	OUTINE CUTINE	**************************************
PATH R ******* PATH:	OUTINE CUTINE	**************************************
PATH R ******* PATH:	OUTINE CUTINE	**************************************
PATH R ******* PATH: *******	OUTINE CUTINE	**************************************
PATH R ******* PATH: *******	OUTINE CUTINE	**************************************

the answer to the empty square question is recovered. If the square is empty and the piece is a Bishop, Rook, or Queen, it is possible to continue moving in the same direction. In this case control passes back to the call to PATH for another move in that direction. Kings and Knights may make only one move in a given direction.

When the time comes to consider a new direction of movement for the piece, the index into the direction table is incremented. DCOUNT, the number of directions to con-

Add this move to the move list.

Increment direction index for two square move direction. Decrement the direction count.

Load the address where the piece was saved.

Check the flag in the piece which tells whether it has moved before.

If the pawn has never moved, go generate a second forward move. (The pawn can move two squares on the first move.) Otherwise go get new direction, skipping second forward move.

Restore the answer to the empty square question. If the square is not empty, this is not a valid move. Go check the next direction.

Otherwise add this move to the move list. Go check the next direction.

Restore the answer to the empty square question. If the square is empty, it is not a normal pawn capture. Go try en passant. Get the "to" position of the move.

If the board index is 91 or greater, this is the last rank and a White pawn promotion.

If so, go set promote flag.

Otherwise, if the board index is less than 29, this is the first rank and a Black pawn promotion.

If not, just go add the move to the move list.

Load the address of the promotion flag.

Set the flag (bit 5 of P2), and go add the move to the move list.

Check for possible en passant capture and add it to the move list if legal.

Go check the next direction.

Get the address of the location where the "from" position was stored. Get the "from" position from that memory location. Add in the direction from the direction table, giving the "to" position. Use "to" position to form an index into the board array. Load the board index. Get the contents of the board at the "to" square. Is the "to" position off the board? If so, go set off-board flag. Save contents of the board at "to" square. Isolate piece type. Save piece type. Return if the square is empty. The flag value is returned in the A register and it is already 0. Get piece again. Load the address of the moving piece. Compare the pieces. Check to see if the colors match. If so, after the exclusive OR the color bit will be 0. If they match, go set match flag. Otherwise, set different color flag and return.

Set same color flag and return.

Set off-board flag and return.

sider, is decremented. When DCOUNT reaches zero, all the moves for the piece have been generated. If the piece involved is the King, a call to castle will add any





Note: The authors' complete Sargon chess playing program is available in book form. The documentation includes a complete Z-80 listing with detailed comments. The price is \$15 from Dan and Kathe Spracklen, 10832 Macouba Pl, San Diego CA 92124, or from local computer stores for \$17.95.

legal castling moves. Then control is returned

All that remains is to discuss the special

pawn logic. Pawns are peculiar in that they

to GENMOV.

96 October 1978 © BYTE Publications Inc

pied by an enemy piece. It may also be possible to move to the eighth rank in capturing, so pawn promotion must be considered here as well. Another type of diagonal pawn move is the *en passant* capture. It must be considered by a call to ENPSNT. Finally it is time to consider a new direction, as is done for the other piece types.

If, however, the direction of movement is forward, the "to" square must be empty. Pawn promotion must be checked for on forward moves. If the piece has never moved before, another move in the same direction is a possibility. Otherwise it is time to consider a new direction. Figures 5 and 6 are flowcharts of MPIECE and PATH, respectively.

The Other Move Generation Routines

The move generation driver is GENMOV, the generate move routine. The basic function of GENMOV is to generate the move set for all pieces of a given color. It scans the board checking for a piece of the same color and calls MPIECE, the piece mover routine.

CASTLE and ENPSNT are also key routines in move generation. CASTLE checks the legality of both King side and Queen side castling. It adds them to the move list if legal. Basic checks must include:

> Has King moved? Is King in check? Has Rook moved? Are the intervening squares empty? Are any squares that the King passes through under attack?

ENPSNT checks for any en passant pawn captures and adds them to the move list if legal. The tests must include:

> On the fifth rank? Was previous move the first move for the enemy pawn? Is the enemy pawn on an adjacent file?

INCK, the check routine, performs the



U.S. Robotics now combines the price/performance leader in 300 Baud Modems with the price/performance leader in hardcopy terminals to bring you teleprinter capability at incredible package prices.

43 KSR	= \$1195
	= \$1365
pe Model 43 RO	= \$1215
Direct ¹ Connect Style S324 S299 ia standard extens	
-	pe Model 43 RO Direct ¹ Connect Style S324 S299

(²Connection to phone lines via CBS-1001F DAA which can be leased from phone company for approximately \$5.00/mo. plus installation fee.)

Interfaces for stand alone modems: USR-310 — RS232C only USR-320 and USR-330 — RS232C and 20 ma. (Specify with order. If both interfaces are required, add S10 to unit price.)

All products include a 90 day warranty and optional annual maintenance package. Add 1% shipping and handling in the continental U.S. Illinois residents add 5% Sales Tax.

U.S. ROBOTICS, INC. 2440 N. Lincoln/Chicago, IL 60614/(312) 528-9045

	-11		
540			
	And Lines Street		
-	Chinal Contract of the first state	inter terrinet, benned	
\bigcirc	and a second second second	addaunia samara	
	Links Same	TT TTN. Print	
2			
		the former and the former	

It's **TIME** you brought your LSI-11 up to **DATE**. **TIME** and **DATE**, two important parameters in the computer world, are available to your LSI-11 on one **DUAL SIZE BOARD**. When requested, the TCU-50D will present you with the date (month and day), time (hour and minutes), and seconds. Turn your computer off and forget about the time — your battery supported TCU-50D won't, not for 3 months anyway. The correct date and time will be there when you power up.

function of determining whether or not the King is in check. The basic method used is to scan outward from the King looking for attackers, by calling ATTACK.

The attack routine finds all attackers on a given square by scanning outward from the square until one of the following occurs:

> A piece is found that attacks this square. A piece is found that doesn't attack

this square. The edge of the board is reached. The TCU-50D is shipped preset to your local time, but can be set to any time you want by a simple software routine.

AT \$295 YOU CAN'T AFFORD TO IGNORE TIME

Time is only one way we can help you upgrade your LSI-11 or PDP-11 system. We'd also like to tell you about the others. So contact Digital Pathways if you're into -11's. We are too.



DIGITAL PATHWAYS INC. 4151 Middlefield Road • Palo Alto, California 94306 • Telephone (415) 493-5544

Circle 89 on inquiry card.

Note: Next month the authors discuss their Sargon Exchange Evaluator.

Figure 6: Flowchart of the PATH routine, which performs the actual move of the piece.



2

In the case where this routine is called by CASTLE or INCHK, the routine is terminated as soon as an attacker of the opposite color is encountered.

ADMOVE adds a move to the move list. The move list is a linked list. Each move in the move list is stored in a 6 byte area. The meaning of each byte is as follows:

- 0&1 MLPTR - Move list pointer. A pointer to the next move in the move list. Used to facilitate sorting the list.
 - MLFRP Move list from position.

The board position from which

MLTOP - Move list to posi-3 tion.

The board position to which the piece is moving.

- MLFLG Move list flags. 4
- 5 MLVAL – Move list value.

Contains the score assigned to the move in evaluation.

It is hoped that this introductory discussion will assist potential chess programmers in getting started. With the essentials of move generation out of the way, the fun part of evaluation can begin.

BIBLIOGRAPHY

In writing Sargon, it was our original intention to put together the first version without any research into the attempts made by others. In this respect Sargon is a unique creation. After competing in the Second West Coast Computer Faire, we began to investigate some of the literature. This bibliography presents some of the references we found most helpful, together with our evaluations.

1. Michie, Donald, On Machine Intelligence, Edinburg University Press, 1974.

Michie's book provides an excellent treatment of exchange evaluation. He uses the concept of an exchange polynomial for accurately determining the outcome of battles engaged on the board. The basic approach we used in XCHNG, the Sargon exchange evaluator, turned out to be surprisingly similar. Sargon's approach, however, is far less computationally complex. We highly recommend this reference to anyone planning to write a chess program without look-ahead.

- 2. Samuel, A L, "Some Studies in Machine Learning Using the Game of Checkers. 11-Recent Progress," IBM Journal, November 1967. Samuel provides a complete though sometimes difficult treatment of alpha-beta pruning. One of the few articles we encountered before writing Sargon, Samuel's article is the basis for the tree search used in the Sargon program.
- 3. Fine, Reubin, Ideas Behind the Chess Openings, David McKay, New York, 1943. Fine's book makes a great starting point for

anyone contemplating the addition of an opening book. Although Fine does not present enough lines of play for a complete book, it does provide a good orientation to other



references.

4. Chernev, Irving, Practical Chess Endings, Dover Publications, New York, 1961.

Perhaps the greatest weakness we've seen in microcomputer chess programs is the play of the endgame. Chernev's book presents a marvelously readable introduction to this phase of the game.

5. Kmoch, Hans, Pawn Power in Chess, D McKay Company, New York, 1959.

> Alas, too many microcomputer chess programs shoot out pawns like photon torpedoes. Kmoch provides an excellent introduction to what constitutes good pawn structure.