```
102|( BASEBALL MASTER LOADER SCREEN    DATE [redacted] )
103|( MASTER LOADER SCREEN #2 )
104|( MASTER LOADER SCREEN #3 )
105|( MASTER LOADER SCREEN #4 )
106|( MASTER LOADER SCREEN #5 )
107|( BB patterns )
108|( BB patterns )
109|( BB patterns )
110|( BB patterns )
111|( BB patterns )
112|( BB, patterns )
113|( BB pattern equates  -pattern index # & magic- )
114|( BB sentry definitions   CHKSCRTIME , BASETABLE )
115|( BB sentry definitions    RESETRUNNERS )
116|( BB sentry defined routines    CHKCAUGHT )
117|( BB sentry defined routines )
118|( BB sentry defined routines )
119|( BB sentry defined routines     THROWANIM )
120|( BB sentry defined routines )
121|( BB sentry defined routines   CREDARROW MLINN )
122|( BB sentry defined routines     THROWBALL , PTTBL )
123|( BB sentry defined routines )
124|( BB sentry defined routines    BATHIT , STARTMUSIC , SETOFTBL )
125|( BB sentry definitions     BUTTONCHECK )
126|( BB sentry definitions ) ( pitch control )
127|( BB sentry definitions     INSCN , PP )
128|( BB sentry definitions     DOGOVER )
129|( BB sentry definitions     TF )
130|( BB sentry definitions     TF )
131|( BB sentry definitions     TFIELD , STRTBBALL , PLRSELCT )
132|( BB sentry loop      SENTRY , TERSECHK )
133|( BB gamestart call        GS )
134|( BASEBALL SCORES 4-3, 10-6, 5-4, etc. ) BASE @ HEX
135|( FOUL,CHEERS,BAT CRACK ) HEX BTABLE FOULSCORE 0 VIBS
136|( CROWD CHEERS & TAKE ME O-T-T-B-G.) HEX
138|( BASEBALL SOUNDS , SIREN CANNON ) HEX
140|( SAFE , OUT , ) HEX
141|( MUSIC PROCESSOR COMANDS )  BASE@ HEX
142|( NOTE CONSTANTS )
143|( MUSIC PROCESSOR IN ASSEMBLY )
144|( MUSIC PROCESSOR AS A CODED SUBROUTINE )
145|( MUSIC PROCESSOR CALLS )
146|( BB music calls )
147|( BB hit sector constants )
148|( BB constants )
149|( BB variables)
150|( BB variables)
151|( BB variables)
152|( BB variables)
153|( infield logic loop      LDINFLDPA , )
154|( BB infield logic    HITDST , FHTBL , GHTBL , HTBL , HMHTBL )
155|( BB infield action by sector )
156|( BB infield action by sector   GRNDRSTABLE )
157|( BB infield action by sector    GRNDRACTION )
158|( BB infield action     GRNDRHIT , CHKCN )
159|( BB  fence hit check does homer set up    HOMERCHK )
160|( BB infield action )
161|( BB infield action     OUTFLDHIT )
162|( BB infield action     INLOG )
163|( BB hit run and throw for outfield fly )
164|( BB hit run and throw for outfield fly   TAKEOFF )
165|( BB hit run and throw for infield grounder )
166|( BB hit run and throw for infield grounder )
167|( BB hit run and throw for infield grounder )
168|( BB who's on which base    WHONBASEODDS )
```

Baseball
Test Version
11/20/78

```
169|( BB who's on which base    WHBODDS )
170|( BB hit logic )
171|( BB hit logic   SWINGTABLE )
173|( BB outfield computer control     CMPOF )
174|( BB tractor ball logic     TBALLPRC )
175|( BB outfield tractor ball running algorithm )
176|( BB out fielder animation logic )
177|( BB pitching algorithm     TBALLPITCH )
178|( BB short subroutines string routines )
179|( BB short subroutines string routines )
180|( BB    BALLERASE , DOHOMER , DOEHOMER )
181|( BB string routines    OUTTIME )
182|( BB short subroutines   STRIKETIME , BALLTIME , FOULTIME )
183|( BB short subroutines SCOREME , STTBL , STETBL )
184|( BB string routines  STRINGGO , STRINGPRC )
186|( BB pattern tables )
187|( BB pattern tables )
188|( BB pattern tables )
189|( BB pattern table matrix   PATTERNS )
190|( BB op codes for playaction defined   BSRTBL , DEACTIVATE )
191|( BB op codes for playaction defined   THWANMSET )
192|( BB op codes for playaction defined     BLDST )
193|( BB op codes for playaction defined )
194|( BB op codes for playaction defined     FLDDST )
195|( BB op codes for playaction defined    OFFDST )
196|( BB op codes for playaction defined     RUNDST )
197|( BB op codes for playaction defined )
198|( BB op codes for playaction defined )
199|( BB playaction op code  OFMOTION , WAITTHRW , BLMOTION , OPTBL)
200|( BB playaction process   OPCODECHK , LOADANM )
201|( BB playaction process    ANMSEQLOAD , PACTLOAD )
202|( BB playaction process     PATFETCH )
203|( BB playaction process     PLAYACT )
211|( BB inning player intialization   SETFDST )
212|( BB inning player intialization    LSETTFCOOR )
213|( BB inning player intialization RETPA , SETTFPA )
214|( BB inning player intialization ZERORAM , SPEC0RAM , LSETTF )
215|( BB playaction loader )
216|( BB vectors )
225|( BB play action tables )
226|( BB play action tables -runners- )
227|( BB play action tables fieldlogic -infielders- )
228|( BB play action tables fieldlogic -infielders- )
229|( BB play action tables  single player pitches )
230|( BB destination delta calculation   DSTCALC )
231|( BB destination delta calculation )
232|( BB set destination registers from vector )
233|( BB short subroutines COMPHL COMPDE TIMEDCR BONE BZERO WUPGO )
234|( BB short subroutines DIVHLBY4 INDEXW EX  )
235|( BB short subroutines AUTOR1 ect. , LVRSTAT , WALKOVER )
236|( BB short subroutines  MULTHLBY4 , KILLOF , WAIT )
237|( BB short subroutines INFLDACT OUTFLDACT ALLFLDACT CMTALL )
238|( BB short subroutines     BLERASE , WUPWRT , CHGS , DWAIT )
239|( BB short subroutines FLSHTON FLSHTOFF FLSHWUP DOCHGS )
240|( BB short subroutines     CHKFLSHSTAY )
241|( VGS write routines    relabs , magic equates )
242|( VGS write routines    reloff )
243|( VGS write routines      write )
244|( VGS write routines   writep , WRITE )
245|( VGS write routines    WRITER )
246|( VGS character routines     cpost  )
247|( VGS character routines     cpost con't. )
248|( VGS character routines     CPOST , SPOST , 3DROP )
249|( VGS character routines     NPOST )
250|( VGS character routines     BCD+ , BCD+! )
251|( BB vector write     VWRITE )
```

```
252|( BB interupt vector erase      VERASE )
253|( BB interrupt    CHKGROUNDER )
254|( BB interrupt      TIMER )
255|( BB interrupt      PERSPECTIVE )
257|( BB interrupt     VECTOR )
258|( BB interrupt       BATHITCHK )
259|( BB interrupt     BATWRITE , BATSWING )
260|( BB interrupt )
261|( BB interrupt        OFBLCHK )
262|( BB interrupt        BLPOSCHK )
263|( BB interrupt )
264|( BB interupt main     WINTBL , INTERRUPT )
265|( BB interupt main )
266|( main vectoring, does 3 vectors given IX-starting vect )
267|( BB interupt ball and bat process )
268|( BB interrupt )
269|( BB interrupt call      INTERRUPTS0 , HMRINT , SI0 , SI1 )
270|( BB field table  dugout pattern     cnline )
271|( BB line vector routines    CNLINE , DLINE )
272|( BB line vector routines    RECTAN , OUTLINE )
273|( BB field write main       FIELDWRT , SUP )
274|( BB field write main       FL )
275|( BB field write )
280|( PIXEL TABLES ) BASE@ HEX
281|( VECTOR GENERATOR )
282|( VECTOR GENERATOR ) BASE@ DECIMAL
283|( VECTOR GENERATOR )
284|( VECTOR GEN )
285|( VECTOR GEN - WRITE POINT AND TEST )
286|( COIN READING ROUTINE ) HEX
287|( BB coin routine         CHKCOIN1 )
288|( BB coin routine         CHKCOIN1 )
290|( I/O PORT DEFINES ) BASE@ HEX
291|( INTERRUPT ROUTINES ) HEX
292|( Interrupt routines )
293|( VGS screen handeling verbs    INTCOMMERCIAL , FILL , SCRERASE )
294|( VGS        NDUP )
295|( HIGH SPEED RANDOM NUMBER ROUTINE )
296|( NUMBER TABLE FOR STRING DISPLAY ROUTINES )
297|( CHARACTER PATTERN TABLE FOR DISPLAY )
298|( CHARACTER PATTERN TABLE CONT. )
299|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
300|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
301|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
302|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
303|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
304|( system verbs )
305|( BB sentry string routines  DPCN CHKGMCNT ) HEX
306|( BB sentry string routines  INSTRC )
```

```
 +--------Block    102--------
 0|( BASEBALL MASTER LOADER SCREEN    DATE 11/19/79 )
 1|  { : BASE! } BASE ! { ; } { : BASE@ } BASE @ { ; }
 2|CR ." system verbs"          304 LOAD
 3|CR ." vgs "                 290 LOAD
 4|CR ." patterns "            107 LOAD
 5|-->
 6|CR ." HERE-" HERE H. HEX 8241 DP ! DECIMAL
 7|CR ." BASE- " BASE? CR ." sysave at 310 load 103  " CR .
 8|;S
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    103--------
 0|( MASTER LOADER SCREEN #2 )
 1|CR ." music "                 146 LOAD
 2|CR ." variables " 147 LOAD   216 LOAD
 3|CR ." pattern matrix "       186 LOAD
 4|CR ." parameter control"     276 LOAD
 5|CR ." vgs write routines"    241 LOAD
 6|CR ." playaction tables "    225 LOAD
 7|CR ." short subroutines "    233 LOAD
 8|CR ." dest calculation "     230 LOAD
 9|CR ." cmp outfld control"    173 LOAD
10|CR ." line vector "          280 LOAD
11|CR ." field generartor "     270 LOAD
12|CR ." who's on base "        168 LOAD
13|CR ." run throw logic "      163 LOAD
14|CR ." hit logic "            170 LOAD
15|-->
 +--------Block    104--------
 0|( MASTER LOADER SCREEN #3 )
 1|CR ." string process "       178 LOAD
 2|-->
 3|CR ." BASE- " BASE? CR ." sysave at 400 load 105  " CR .
 4|
 5|
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
```

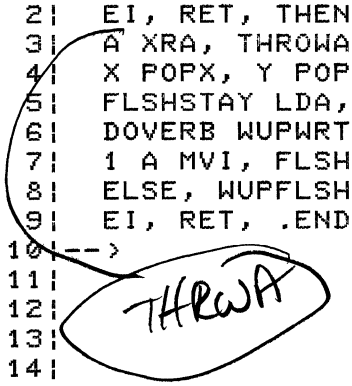
```
 +--------Block    105--------
 0|( MASTER LOADER SCREEN #4 )
 1|CR ." tractor ball logic"  174 LOAD
 2|CR ." infield action "     153 LOAD
 3|CR ." playaction process " 215 LOAD
 4|-->
 5|CR ." BASE- " BASE? CR ." sysave at 450 load 106    " CR .
 6|;S
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    106--------
 0|( MASTER LOADER SCREEN #5 )
 1|CR ." coin routine"         286 LOAD
 2|CR ." interupt "            253 LOAD
 3|HEX 8000 DP ! DECIMAL
 4|CR ." sentry "              114 LOAD
 5|;S
 6|CR ." BASE- " BASE? CR ." sysave at 500 "
 7|CR ."                      ***** all done *****    "
 8|CR ." HERE- "  HERE H. CR . ;S
 9|;S
10|
11|
12|
13|
14|
15|
 +--------Block    107--------
 0|( BB patterns )
 1|BASE@  HEX
 2|{  : 8STF  } B, B, B, B, B, B, B, B, {  ;  }
 3|{  : 12STF  } 8STF B, B, B, B, {  ;  }
 4|{  : 14STF  } B, B, 12STF {  ;  }
 5|{  : 16STF  } B, B, 14STF {  ;  }
 6|{  : 18STF  } B, B, 16STF {  ;  }
 7|{  : 24STF  } 12STF 12STF {  ;  }
 8|{  : 26STF  } 12STF  14STF {  ;  }
 9|{  : 28STF  } 14STF 14STF {  ;  }
10|{  : 32STF  } 18STF 14STF {  ;  }
11|LABLE NOBOD 0 0 2 1 0 0 B, B, B, B, B, B,
12|LABLE BALLPAT 0 C0 0 C0 2 2 0 0 B, B, B, B, B, B, B, B,
13|LABLE UPTRI 80 C0 E0 F0 F8 B, B, B, B, B,
14|LABLE DNTRI F8 78 38 18 8 B, B, B, B, B,
15|LABLE IMRK F8 B, 70 B, 20 B, -->
```

```
 +--------Block    108--------
 0|( BB patterns )
 1|LABLE TUP1M C0 1 80 1D 80 D 80 D 80 F 90 F A0 F C0 F
 2|       80 FF 80 F 0 7 0 7 C 2 6 7 28STF
 3|LABLE TUP2M 0 3 0 32 0 12 0 1E 80 1E 0 1F 0 FE 0 1E
 4|       0 C 0 C A 2 5 6 24STF
 5|LABLE TUP3M 0 6 0 34 0 1C 0 1C 0 1B 0 FE 0 1C 0 8
 6|       8 2 4 5 18STF B, B,
 7|LABLE TUP4M 0 6C 0 38 0 38 0 FC 0 38 0 10 6 2 3 4 16STF
 8|LABLE TUP1F 0 1C 0 98 0 D8 0 F8 0 78 0 78 0 79 0 7A
 9|       0 7C 0 78 0 B0 0 B0 0 80 0 80 E 2 7 1 32STF
10|LABLE TUP2F 0 18 0 90 0 D0 0 70 0 70 0 74 0 78 0 70
11|       0 B0 0 B0 0 80 B 2 5 1 26STF
12|LABLE TUP3F 0 30 0 A0 0 E0 0 60 0 60 0 68 0 F0
13|       0 A0 0 80 9 2 4 1 18STF B, B, B, B,
14|LABLE TUP4F 0 30 0 A0 0 E0 0 6 0 F0 0 A0 0 80
15|       7 2 3 1 18STF -->
 +--------Block    109--------
 0|( BB patterns )
 1|{ : 20STF 18STF } B, B, { ; }
 2|LABLE RUP3M 0 30 0 20 0 30 0 30 0 34 0 78 0 30 0 10
 3|          8 2 4 3 20STF
 4|LABLE RUP4M 0 30 0 20 0 B8 0 70 0 38 0 10 6 2 3 3 16STF
 5|LABLE RUP0M 0 0E 0 0C 0 0F 0 0E 0 0E 0 0E 0 0E 80 1E 80 1E
 6|     0 1F 0 1F 0 0E 0 6 0 6 E 2 7 6 32STF
 7|LABLE RUP1M 0 1C 0 18 0 1E 0 1C 0 1C 0 1C 0 3C 0 3D 0 3F
 8|  0 1E 0 C 0 C C 2 6 4 28STF
 9|LABLE RUP2M 0 1C 0 18 0 1E 0 1C 0 1C 0 3D 0 3F 0 1E
10|     0 C 0 C A 2 5 4 24STF
11|LABLE RUP0B C0 80 80 F9 80 F9 80 19 80 1F 0 0F 0 0E
12|    0 8E 0 4E E0 2F 0 1E 0 6 0 6 0D 2 7 6 28STF B, B,
13|LABLE ONBASE2 0 33 0 19 0 0F 0 7 40 8F 80 5F 0 3F 0 1E
14|       0 0C 0 0C 0A 2 5 0A 24STF
15|-->
 +--------Block    110--------
 0|( BB patterns )
 1|LABLE RUP1B 80 83 0 F3 0 F3 0 1F 0 1E 0 9C 0 5C 80 3F
 2|     0 1C 0 C 0 C B 2 6 4 24STF B, B,
 3|LABLE RUP2B 80 83 0 F3 0 FF 0 1E 0 9C 0 5C 80 3F
 4|     0 C 0 C 9 2 5 4 20STF B, B,
 5|LABLE RUP3B 0 8C 0 E8 0 38 0 30 0 B0 0 7C 0 10
 6|    7 2 4 3 18STF
 7|LABLE RUP4B 0 4C 0 68 0 38 0 B0 0 7C 0 10 6 2 3 3 16STF
 8|LABLE CBUP1 0 70 0 E0 0 E0 0 E0 0 E0 0 E0 0 F8 0 E4
 9|       0 E4 0 E0 0 60 0 60 C 2 5 1 28STF
10|LABLE CBUP2 0 70 0 E0 0 E0 0 E0 0 E0 0 F8 0 E4
11|       0 E4 0 E0 0 60 0 60 0B 2 5 1 26STF
12|LABLE PTNID 0 0E 0 6 0 6 0 7F 0 7F 0 4F 0 0F 0 7F 0 8F 0 7F
13|       0 9 0 6 C 2 6 6 28STF
14|LABLE ONBASE1 80 31 80 18 80 0C 80 7 80 7 80 8F A0 5F C0 3F
15|       0 1F 0 0C 0 0C 0B 2 6 7 26STF -->
```

```
 +--------Block    111--------
 0|( BB patterns )
 1|LABLE FUP1B 0 7 0 6 0 86 0 FE 0 FE 10 1F A0 0F C0 7 80 3
 2|       80 1 80 1 B 2 5 6 24STF B, B,
 3|LABLE FUP2B 0 0E 0 8C 0 FC 20 FC 40 1E 80 F 0 7
 4|       0 3 0 3 9 2 4 5 18STF B, B, B, B,
 5|LABLE FUP1M 0 38 0 30 0 3C 0 38 0 3C 20 3E 40 1F
 6|       80 F 0 7 0 3 0 3 B 2 5 3 24STF B, B,
 7|LABLE FUP2M 0 38 0 30 0 38 20 3C 40 1E 80 F
 8|       0 7 0 3 0 3 9 2 4 3 18STF B, B, B, B,
 9|LABLE BATMID FF FF FF FF 2 2 0 0 8STF
10|LABLE BATU45 0 C0 0 E0 0 70 0 38 0 1C 0 E 0 7 80 3 C0 1 E0 0
11|       70 0 38 0 18 0 D 2 C 0 28STF B, B,
12|LABLE BATU30 0 C0 0 F0 0 7C 0 1F C0 7 F0 1 7C 0 1E 0 6 0
13|       9 2 8 0 18STF B, B, B, B,
14|LABLE BATU90 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0
15| 0 C0 B 2 D 0 24STF B, B,  -->
 +--------Block    112--------
 0|( BB, patterns )
 1|LABLE BATD30 6 0 1E 0 7C 0 F0 1 C0 7 0 1F 0 7C 0 F0 0 C0
 2|        9 2 0 0 18STF B, B, B, B,
 3|LABLE BATD45 18 0 38 0 70 0 E0 0 C0 1 80 3 0 7 0 E 0 1C
 4|       0 38 0 70 0 E0 0 C0 D 2 0 0 28STF B, B,
 5|LABLE BATD90 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0 0 C0
 6| 0 C0 0 C0 0 C0 B 2 0 0 24STF B, B,
 7|LABLE HOMEPLATE 0 10 0 38 0 7C  0 FE 0 FE 0 FE 0 FE
 8|       7 2 0 4 18STF
 9|LABLE BASEPAT 0 F8 0 F8 0 F8 0 F8 4 2 0 4 12STF
10|LABLE BASE2PAT 0 F0 0 F0 0 F0 3 2 0 4 8STF B, B,
11|LABLE STN1 0 33 0 12 0 12 0 12 0 1E 40 9E 40 9E 40 9E
12|       80 7F 0 3F 0 C 0 C C 2 6 4 28STF
13|LABLE STN2 0 33 0 12 0 12 0 1E 40 9E 40 9E
14|     80 7F 0 3F 0 C 0 C A 2 5 4 24STF
15|  -->
 +--------Block    113--------
 0|( BB pattern equates  -pattern index # & magic- )
 1|LABLE STN3 0 6C 0 28 0 28 0 38 0 BA 0 BA 0 7C 0 10
 2|     8 2 4 3 20STF
 3|LABLE STN4 0 6C 0 28 0 BA 0 BA 0 7C 0 10 6 2 3 3 16STF
 4|LABLE STNPT C0 1D 80 D 80 D 80 D 80 F 90 4F 90 4F 90 4F
 5|       E0 3F C0 1F 0 7 0 7 C 2 6 6 28STF
 6| 28 CONSTANT RRUN 68 CONSTANT LRUN
 7|128 CONSTANT RFLD 168 CONSTANT LFLD
 8|228 CONSTANT RTRW 268 CONSTANT LTRW ( 2 )
 9|328 CONSTANT RCB  368 CONSTANT LCB ( 3 )
10|428 CONSTANT STND 528 CONSTANT NBD
11|628 CONSTANT BAL 728 CONSTANT SCB
12|828 CONSTANT STPT 928 C= PTTHW
13|0A28 C= RONBS 0A68 C= LONBS
14|BASE!
15|;S
```

```
 +--------Block    114--------
 0|( BB sentry definitions   CHKSCRTIME , BASETABLE )
 1|305 LOAD ( stings )
 2|BASE@ HEX
 3|LABLE BASETABLE 6000 , 4000 , 4400 , 2800 , 6000 , 1000 ,
 4| AA00 , 2800 , ( base coor )
 5|CODE CHKSCRTIME   ( score process ) DI,
 6| X PUSHX, HITOF LDA, A ANA, 0< >, IF, PLAYON LHLD, H PUSH,
 7|  X POPX, ( player play is on ) VMAGIC R4 D LXI, E A MOV,
 8|  L CMP, ( r4? ) < >, IF, DBLFLAG LDA, A ANA, 0=, IF,
 9|  VLENGTH D LXI, D DAD, PLAYON SHLD, THEN, THEN,
10|  ELSE, VMAGIC R1 X LXIX, ( infield hit ) THEN,
11|  A XRA, A VWBASE X STX, SCRTIME STA, X POPX, EI,
12|  PLYR1UP LDA, A ANA, 0< >, IF, SCORE1 H LXI, 1 D MVI,
13|  ELSE, SCORE2 H LXI, 2 D MVI, THEN,
14|  M A MOV, 1 ADI, DAA, A M MOV, D A MOV, SCORESHOW STA, RET,
15|-->
 +--------Block    115--------
 0|( BB sentry definitions    RESETRUNNERS )
 1|  FORWARD RSR2WB FORWARD RSR3WB FORWARD RSR4WB
 2|CODE RESETRUNNERS ( reset runners to order of on base )
 3|.ASSEMBLE 4 A MVI, BEGIN, PSW PUSH,
 4|  VWBASE R1 LDA, A ANA, 0< >, IF, VWBASE R2 LDA, A ANA,
 5|  RSR2WB JZ, VWBASE R3 LDA, A ANA, RSR3WB JZ, RSR4WB JMP, THEN,
 6|  VMAGIC R1 D LXI, VMAGIC R2 H LXI, VLENGTH B LXI, LDIR,
 7| LABEL RSR2WB VMAGIC R2 D LXI, VMAGIC R3 H LXI, VLENGTH B LXI,
 8| LDIR, LABEL RSR3WB VMAGIC R3 D LXI, VMAGIC R4 H LXI,
 9| VLENGTH B LXI, LDIR, LABEL RSR4WB A XRA, VWBASE R4 STA,
10|  VOFW R4 STA, 2800 H LXI, VX R4 SHLD,
11|  A H MOV, A L MOV, VDX R4 SHLD, VDY R4 SHLD,
12|  AC00 H LXI, VY R4 SHLD, NOBOD H LXI, VPAT R4 SHLD, 1 A MVI,
13|  VSTATUS R1 STA, VSTATUS R3 STA, VSTATUS R3 STA,
14|  VSTATUS R4 STA, RUNBPA H LXI, VPLAYACTPC R4 SHLD, PSW POP,
15|  A DCR, 0=, END, ( do 4 tms )    RET, .END -->
 +--------Block    116--------
 0|( BB sentry defined routines    CHKCAUGHT )
 1|   FORWARD OUTOFFF FORWARD CCLEAVE
 2|CODE CHKCAUGHT    ( caught set in interupt )
 3|  .ASSEMBLE WALK LDA, A ANA, RNZ,   Y PUSHX, X PUSHX,
 4|  THROW STA, CAUGHT STA, TOOFF STA, TOOF STA,
 5|  1 A MVI, NORUN STA,
 6|  THROWAROUND LDA, A ANA, 0=, IF, 30 A MVI, STRIKES STA,
 7|  BALLS STA, OFCATCH LDA, A ANA, DI, 0< >, IF,
 8|  THROWANM STA, WHOSUP LIYD, OUTOFFF JMP, THEN,
 9|  PLAYON LIYD, OFOUT LDA, A ANA, 0< >, IF, A XRA, OFOUT STA,
10|  ELSE, CSAFE A MVI, STRING STA, THEN, BASEBLAT LDA, A C MOV,
11|  VATBS VSTATUS Y BITX, 0=, IF, VOFW Y A LDX, A ANA, 0< >, IF,
12|  VHW VRSTAT Y BITX, 0< >, IF, 2 A MVI, LVRSTAT CALL, ELSE,
13|  VWBASE Y A LDX, C CMP, ( runner at this base )   =, IF,
14|LABEL OUTOFFF ( out )  OFFFPA H LXI, A XRA,
15|-->
```

```
  +--------Block    117--------
 0|( BB sentry defined routines )
 1|   A VWBASE Y STX, H VPLAYACTPCH Y STX, ( run off field )
 2|   L VPLAYACTPCL Y STX, FLDCLR H LXI, M INR,   COUT A MVI,
 3|   STRING STA,   VGO VSTATUS Y RESX, OUTS LDA, 32 CPI,
 4|   =, IF, A XRA, THROWANM  STA,
 5|   X POPX, Y POPX, EI, RET, ( ret if 3rd out ) THEN,
 6|   THEN, THEN, THEN, THEN,
 7|   OFCATCH LDA, A ANA, 0<>, IF, OFOUT STA,
 8|   CCLEAVE JMP, THEN, C A MOV, ( baseblat ) 3 CPI, <>, IF,
 9|   FLDON1ST H LXI, ' INDEXW CALL, ( who has it ) D PUSH, X POPX,
10|   VPT VSTATUS X BITX, 0=, IF, X PUSHX, H POP, WHOTHROWS SHLD,
11|   1 A MVI, THROWANM STA, INAIR STA, DBLPLAY LDA, A ANA,
12|   0<>, IF,  ( double play ) A XRA, DBLPLAY STA, VLENGTH D LXI,
13|   Y PUSHX, H POP, D DAD, PLAYON SHLD, 1 A MVI, DBLFLAG STA,
14|   ELSE, ( no dblplay ) 1 A MVI, THROWAROUND STA, THEN,
15|-->
  +--------Block    118--------
 0|( BB sentry defined routines )
 1|LABEL CCLEAVE A XRA, OFCATCH STA, X POPX, Y POPX,
 2|   EI, RET, THEN, THEN, THEN, ( throwaround catch )
 3|   A XRA, THROWAROUND STA,
 4|   X POPX, Y POPX, DI, DOVERB STFLD (  reset fielders )
 5|   FLSHSTAY LDA, A ANA, 0=, IF,
 6|   DOVERB WUPWRT ' WUPWRT H LXI, FLSHWHO SHLD,
 7|   1 A MVI, FLSHON STA, FLSHTIME STA,
 8|   ELSE, WUPFLSH STA, THEN,
 9|   EI, RET, .END
10|-->
11|
12|
13|
14|
15|
  +--------Block    119--------
 0|( BB sentry defined routines      THROWANIM )
 1|SUBR CHKATBS  ( check if active player at base in- A vstatus )
 2|   VACT A BIT, RZ, VATBS A BIT, RNZ, E INR, RET,
 3|CODE THROWANIM  ( sets up BLDST and player anim )
 4|   X PUSHX, Y PUSHX, DI, A XRA, WHOTHROWS LIXD, GRNDR STA,
 5|   THROWANM STA, A VANM# X STX, ( zero plyr deltas )
 6|   A VDXL X STX, A VDXH X STX, A VDYL X STX, A VDYH X STX,
 7|   VXH X H LDX, H INX, H INX,
 8|   VX BL SHLD, VYH X H LDX, VY BL SHLD, ( give ball plyr coor. )
 9|   THROWAROUND LDA, A ANA, 0<>, IF, ( thrown to pt after play )
10|   VY PT LHLD, VDSTY BL SHLD, VX PT LDED, 4 B MVI, ( bl vel )
11|   ELSE, ( base throw )
12|   PLAYON LIYD, HITOF LDA, A ANA, VWBASE Y A LDX, 0<>, IF,
13|   ( hit outfield ) A ANA, 0=, IF, ( run to 1st ) A INR, THEN,
14|-->
15|
```



```
 +--------Block    120--------
 0|( BB sentry defined routines )
 1|  A D MOV, 0 E MVI, VSTATUS R1 LDA, CHKATBS CALL,
 2|  VSTATUS R2 LDA, CHKATBS CALL, VSTATUS R3 LDA, CHKATBS CALL,
 3|  VSTATUS R4 LDA, CHKATBS CALL, A XRA, E ORA,
 4|  0=, IF, ( everybody on base ) 1 D MVI, THEN, D A MOV,
 5|  2 B MVI, ELSE, 5 B MVI, ( throw vel )
 6|  THEN, BASEBLAT STA, ( # of base to throw to ) A SLAR,
 7| BASETABLE  H LXI, ' INDEXW CALL,
 8|  VDSTY BL SDED, H INX, H INX,  M E MOV, H INX, M D MOV,
 9|  THEN, B A MOV, VVEL BL STA, ( set bl vel )
10|  VDSTX BL SDED, VXH X A LDX, D CMP, <, IF, RTHWPA H LXI,
11|  ELSE, LTHWPA H LXI, THEN, H VPLAYACTPCH X STX,
12|  L VPLAYACTPCL X STX, VACT VSTATUS X SETX,
13|  VGO VSTATUS X RESX, 10 A MVI,   THROWTIMER STA,
14|  THROW STA, Y POPX, X POPX, EI, RET, -->
15|
 +--------Block    121--------
 0|( BB sentry defined routines   CREDARROW MLINN )
 1|: MLINN ( mark last inning ) DI PX0 2A A5 4D 3 RECTAN LINN B@
 2|  DUP DUP IF 0A < IF 8F0 SWAP 0 DO 200 + LOOP ELSE DROP 1CF0
 3|  THEN A500 IMRK 301 828 WRITE ELSE DROP DROP THEN EI ;
 4|: CREDARROW   ( update last inning arrow ) DI UPCRED BZERO
 5|  STRT1 B@ IF SELECT BONE ELSE CREDITS B@ CREDITS BZERO
 6|    DUP IF CNSW1 B@ IF GAMEOVER B@ IF CREDITS B!
 7|    CHKPLRS B@ IF ELSE SELECT BONE THEN 0
 8| ELSE CMPLR B@ IF LINN B@ IF 2* ELSE SELECT BONE 1- 2* 1+ THEN
 9|    ELSE ( not cmplr ) DUP 1 AND CREDITS B! 0FE AND
10|      LINN B@ IF ELSE SELECT BONE 1- THEN THEN THEN
11|  ELSE ( not cnsw1 ) LINN B@ IF 2* ELSE SELECT BONE 1- 2* 1+
12|  THEN THEN LINN B@ + LINN B! MLINN
13|  CHKFLSHSTAY ELSE DROP THEN THEN EI ;
14|-->
15|
 +--------Block    122--------
 0|( BB sentry defined routines     THROWBALL , PTTBL )
 1|LABLE PTTBL    ( computer pitching table )
 2| PSB , PCO , PICO , PFB , PFB , PCO , PSU , PSD ,
 3| PICI , PSD ,
 4|
 5|CODE THROWBALL ( wait for timer release ball )
 6|  A XRA, THROWTIME STA,
 7|  BAL H LXI,  PITCHTIME LDA, A ANA, 0<>, IF, TBLPA SHLD,
 8|  DI, ( pitch ) VX PT LHLD, H DCX, H DCX, H DCX,
 9|  H DCX, VX BL SHLD, VY PT LHLD, H DCX, H DCX, H DCX, H DCX,
10|  H DCX, H DCX, H DCX, VY BL SHLD, ( give bl pt coor.)
11|  EI, ' RESETRUNNERS CALL, DOVERB WHBODDS DI,
12|  ( reset runners ) PITCHBLPA H LXI, 1 A MVI, TBALLYSR STA,
13|  SWINGGO STA, -->
14|
15|
```

```
 +--------Block    123--------
 0|( BB sentry defined routines )
 1|  CMPT LDA, A ANA, 0<>, IF, ( computer pitch )
 2|  PTRND# LDA, PTTBL H LXI, ' INDEXW CALL, XCHG, ( rnd pitch )
 3|  THEN, ELSE, ( hit or throw ) THWPA SHLD, THWBLPA H LXI, THEN,
 4|  VPLAYACTPC BL SHLD,   VSTATUS BL H LXI, VACT M SET,
 5|  VGO M RES, EI, RET,
 6|
 7|BTABLE IODTBL   ( inning odds table )
 8|  5 B, 3 B, 3 B, 3 B, 3 B, 2 B, 2 B, 1 B, 1 B,
 9|: INNODDS
10|  INN# B@ CMSW B@ IF DUP 6 > IF DROP 1 ELSE 1 > IF 2 ELSE 3
11|  THEN THEN ( cmp player hit ratio ) SPBON B@ IF DROP 4 THEN
12|  ELSE ( player ) SPBON B@ ( bonous inning ) IF 3 > IF 1 ELSE 2
13|  THEN ELSE 1 - IODTBL B@ THEN THEN IODDS B! ;
14|-->
15|
 +--------Block    124--------
 0|( BB sentry defined routines    BATHIT , STARTMUSIC , SETOFTBL )
 1|CODE BATHIT  ( he hit it time to get em running )
 2|  HITYET STA, DI, DOVERB INLOG
 3|  A XRA, HITTIME STA, PITCHTIME STA, ( tractor ball outfld)
 4|  SWINGGO STA, EI, RET,
 5|
 6|CODE STARTMUSIC A XRA, STMUSIC STA,
 7|  CHEERME LDA, A ANA, 0<>, IF, DOVERB MRAH A XRA, CHEERME STA,
 8|  DOVERB SI0 7 A MVI, 1 OUT, 2 OUT, 3 OUT, A XRA, 0 OUT, 4 OUT,
 9|  ELSE, HOMERUN LDA, A ANA, 0=, IF, DOVERB MFENCE
10|  ELSE, DOVERB MCANNON DOVERB SI1 THEN, THEN, RET,
11|
12|CODE OFLDTBL   OFTBLPA H LXI, A XRA, OFTOTBL STA,
13|DI, VPLAYACTPC RF SHLD, VPLAYACTPC CF SHLD, VPLAYACTPC LF SHLD,
14|  ' OUTFLDACT CALL, EI, RET,
15|-->
 +--------Block    125--------
 0|( BB sentry definitions     BUTTONCHECK )
 1|CODE BUTTONCHK   ( reads buttons for swing run & pitch it )
 2|  HITYET LDA, A ANA, 0<>, IF,
 3|( runner control )
 4|  CMSW LDA, A ANA, 0<>, IF,  ( computer control ) DI,
 5|  X PUSHX, Y PUSHX, 0E D MVI, OFBLCHK CALL, ( chk if close )
 6|  Y POPX, X POPX, EI, 1 XRI,
 7|  ELSE, 14 IN, 2 ANI, 2 XRI, THEN,   FORW STA,
 8|-->
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    126--------
 0|( BB sentry definitions ) ( pitch control )
 1|  PITCHIT LDA, A ANA, RZ, FLDCLR LDA, A ANA, RNZ,
 2|  TIMEOUT LDA, A ANA, 0<>, IF, 14 IN, 1 ANI, RNZ, THEN, DI,
 3|  A XRA, STRIKE STA, HITYET STA, FLSHCNT LDA, A ANA, 0=, IF,
 4|  DOVERB WUPWRT THEN, FLSHSTAY LDA, A ANA, 0=, IF,
 5|  FLSHON STA, FLSHTIME STA, FLSHCNT STA,
 6|  ELSE, A XRA, DOWUP STA, WUPFLSH STA, THEN,
 7|  PITCHPA H LXI, VPLAYACTPC PT SHLD, A XRA,
 8|  PITCHIT STA, VUPDATE# PT STA, 4A A MVI, THROWTIMER STA,
 9|  PITCHTIME STA,   VMAGIC PT H LXI, WHOTHROWS SHLD,
10|  VSTATUS PT H LXI, VACT M SET, ' KILLOF CALL, EI,
11|  ELSE, ( swing control ) CMSW LDA, A ANA, 0<>, IF,
12|  SWRND# LDA, ( set up in whonbase ) 98 ADI, A D MOV,
13|  VYH BL LDA, D CMP, RC, VXH BL LDA, 26 SBI, 4 CPI, RNC,
14|  ELSE, 14 IN, 2 ANI, RNZ, THEN, SWINGGO LDA, A ANA, RZ,
15|  SWING STA, THEN, RET, -->
 +--------Block    127--------
 0|( BB sentry definitions     INSCN , PP )
 1|: ISCN 1C00 1000 828 A" INSERT COINS" SPOST ;
 2|: INSCN ( insert coin call )
 3|  SGO ' ISCN FLSHTON
 4|  CNSW1 B@ IF 0C00 2200 828 A" 1 COIN PER PLAYER 1ST INNING"
 5|  SPOST 1700 3400 828 A" 1 COIN PER PLAYER" SPOST 0F00 4600 828
 6|  A" EACH ADDITIONAL 2 INNINGS" SPOST
 7|  ELSE 1800 2600 828 A" 1 COIN 1ST INNING " SPOST
 8|  800 3C00 828 A" 1 COIN EACH ADDITIONAL 2 INNINGS " SPOST
 9|  THEN 2880 4C00 BASE2PAT 428 WRITR ;
10|: CONGR 700 1000 828 A" CONGRATULATIONS YOU ARE VERY GOOD"
11|  SPOST 0E00 2800 828 A" LETS PLAY AGAIN I WILL BUY" SPOST ;
12|: SORRY 1B00 1000 828 A" TOO BAD I WON" SPOST DP1CN ;
13|: DP1 1A00 2C00 828 A" DEPOSIT 1 COIN" SPOST ;
14|-->
15|
 +--------Block    128--------
 0|( BB sentry definitions     DOGOVER )
 1|: TCNT 700 1800 828 A" TO CONTINUE GAME AT END OF INNING" SPOST
 2|  CNSW12 B@ IF CREDITS B@ IF DP1 ELSE 0F00 2C00 828
 3|  A" DEPOSIT 1 COIN PER PLAYER" SPOST THEN ELSE DP1 THEN ;
 4|: SPLR12 1400 2000 828 A" SELECT 1 OR 2 PLAYERS" SPOST ;
 5|: SPLR CNSW1 B@ IF CREDITS B@ 1 <> IF SPLR12 ELSE 0E00 2000 828
 6|  A" SELECT 1 PLAYER OR DEPOSIT" SPOST
 7|  0F00 3000 828   A" 1 MORE COIN FOR 2 PLAYERS"
 8|  SPOST STRT1 BONE THEN ELSE SPLR12 THEN ;
 9|: DOGOVER ( do game over )
10|  SGO 32 OUTS B! CNSW BONE CMPT BONE PX0 2A B6 4D 8 RECTAN EIDI
11|  8C0 INNX ! INN# BZERO GAMEOVER BZERO  9 LINN B! MLINN DI
12|  GAMEOVER BONE LINN BZERO SCORE1 ZERO PX1 SCRS CMPLR BZERO
13|  FLSHSTAY B@ IF FLSHSTAY BZERO FLSHOFF THEN 10 TWAIT INSCN
14|  80 TWAIT INSCN CREDITS B@ IF SGO THEN SHILL BONE
15|  FLSHOFF DROP 0 PLYR1UP BZERO ( plyr1up 0 ) EIDI ; -->
```

```
  +--------Block    129--------
 0|( BB sentry definitions      TF )
 1|:  TF   ( take the field )
 2|  DI ZERORAM EI 30 DUP DUP OUTS B! BALLS B! STRIKES B!
 3|  15 INP 1 AND CNSW1 B!
 4|  PLYR1UP DUP B@ 1 XOR DUP ROT B! CMPLR B@
 5|  IF DUP IF CMSW BONE ELSE CMPT BONE THEN THEN
 6|  DI SHILL B@ IF SGO DOGOVER ELSE
 7|  GAMEOVER B@ IF SGO CMPLR B@ IF SCORE1 B@ SCORE2 B@ < IF
 8|  CONGR 50 TWAIT LINN 1+B! MLINN DI CONGR
 9|  GAMEOVER BZERO SPBON BONE
10|  ELSE SORRY C0 TWAIT SORRY CREDITS B@ IF DOCHGS
11|  ELSE DOGOVER THEN THEN
12|  ELSE CNSW1 B@ IF CNSW12 BZERO CREDITS B@ 1 = IF DPCN ELSE
13|  CHKGMCNT DUP IF 1 = IF DPCN ELSE DOCHGS THEN
14|-->
15|
  +--------Block    130--------
 0|( BB sentry definitions      TF )
 1|  ELSE DROP DOGOVER THEN THEN CNSW12 BONE
 2|  ELSE ( no cnsw1 ) CHKGMCNT IF DOCHGS
 3|  ELSE DOGOVER THEN THEN THEN ELSE 1STINN B@ IF ELSE
 4|  DOCHGS THEN THEN THEN
 5|  ( plyr1up ) IF INNX @ B700 DNTRI INN# B@ LINN B@ = IF
 6|  TCNT 30 TWAIT TCNT THEN ELSE
 7|  INN# DUP 1+B! B@ 0A = IF INN# BONE
 8|  LINN DUP B@ 9 - SB! ( new pointer ) MLINN DI
 9|  PX0 2A B6 4D 8 RECTAN 8C0 INNX ! THEN
10|  INNX DUP @ 200 + DUP ROT ! B600 UPTRI THEN 501 828 WRITE
11|  FLSHSTAY B@ IF WUPFLSH BONE ELSE ' WUPWRT WUPWRT FLSHTON
12|  EIDI THEN SETFDST SETTF SB0 HMRFLSH BZERO 3 OFTFCNT B! INNODDS
13|  GAMEOVER B@ IF ELSE F00 SUP 3F00 SUP ( sets "up" ) THEN
14|  HITYET BONE MYAH SI0 ;
15|-->
  +--------Block    131--------
 0|( BB sentry definitions      TFIELD , STRTBBALL , PLRSELCT )
 1|: PLRSELECT  ( select singel player or 2 player )
 2|  DI FL   CHKPLRS BONE
 3|  SPLR SELECT BZERO MLINN ;
 4|: STRTBBALL DI INSTRC FL F00 SUP 3F00 SUP
 5|  PLYR1UP BONE SCORE1 ZERO INN# BZERO STRT1 BZERO GAMEOVER BZERO
 6|  CNSW1 B@ IF CREDARROW ELSE MLINN DI THEN SHILL BZERO
 7|  CMPLR B@ IF YOU1ST 50 DWAIT YOU1ST THEN
 8|  8C0 INNX ! 1STINN BONE TF 1STINN BZERO MOPENERS ;
 9|: PLRCHK 14 INP DUP 4 AND 4 XOR IF DROP CMPLR BONE STRTBBALL
10|  DI 800 7000 8828 A" YOU ARE" SPOST EI
11|  ELSE STRT1 B@ IF DROP ELSE 8 AND 8 XOR IF CNSW1 B@ IF
12|  CNSW12 BONE THEN STRTBBALL THEN THEN THEN ;
13|-->
14|
15|
```

```
 +--------Block    132--------
 0|( BB sentry loop       SENTRY , TERSECHK )
 1|: TERSECHK   ( look at terse routine flags ) 0 RND DROP
 2|  TFTIME B@ IF TF THEN SELECT B@ IF PLRSELECT THEN
 3|  CHKPLRS B@ IF PLRCHK THEN UPCRED B@ IF CREDARROW THEN ;
 4|CODE SENTRY    ( game loop ) B PUSH,
 5|  TAKEFIELD LDA, A ANA, ' BUTTONCHK CZ,
 6|  PITCHTIME LDA, A ANA, ' TBALLPITCH CNZ,
 7|  OFTOTBL   LDA, A ANA, ' OFLDTBL CNZ,
 8|  THROWTIME LDA, A ANA, ' THROWBALL CNZ,
 9|  THROWANM  LDA, A ANA, ' THROWANIM CNZ,
10|  HITTIME   LDA, A ANA, ' BATHIT CNZ,
11|  CAUGHT    LDA, A ANA, ' CHKCAUGHT CNZ,
12|  SCRTIME   LDA, A ANA, ' CHKSCRTIME CNZ,
13|  STMUSIC   LDA, A ANA, ' STARTMUSIC CNZ,
14|  DOVERB TERSECHK ( chk terse flags )
15|  B POP, ' SENTRY JMP, NEXT -->
 +--------Block    133--------
 0|( BB gamestart call         GS )
 1|: PUP  ( power up routine )
 2|  INTHIGHRES 0 0 OUTP 0 4 OUTP 7 DUP DUP 2 OUTP 1 OUTP 3 OUTP
 3|  DI 0 RND# @ 1 RND# @ SCRERASE 1 RND# ! 0 RND# ! EMUSIC SI0
 4|  SHILL BONE
 5|  DI FL SGO EI GAMEOVER BONE TF SENTRY ;
 6|BASE! ;S
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    134--------
 0|( BASEBALL SCORES 4-3, 10-6, 5-4, etc. ) BASE @ HEX
 1|BTABLE STRIKESCORE
 2|23 MASTER AB ABVOLS 0B MCVOLS 0 VIBS 1 #E2 CNOTE #F2 BTONE #C2
 3|ATONE 1 #G2 CNOTE #FS2 BTONE 1 #A2 CNOTE #AS2 BTONE 1 #C3 CNOTE
 4|#CS3 BTONE 1 #DS3 CNOTE #E3 BTONE 1 #F3 CNOTE #FS3 BTONE 1 #G2
 5|CNOTE #GS3 BTONE 08 #B3 CNOTE #C4 BTONE QUIET
 6|BTABLE BALLSCORE
 7|28 MASTER AB ABVOLS 0A MCVOLS 0 VIBS 2 #G2 #F2 #D2 NOTES
 8|2 #G2 #E2 #C2 NOTES 2 #G2 #D2 #B1 NOTES 5 #G2 #C2 #A1 NOTES
 9|QUIET
10|BTABLE COINSCORE
11|23 MASTER FF ABVOLS 2F MCVOLS  E8 NOISE 52 VIBS
12|12 #C2 #D2 #E2 NOTES ( 6 #B1 ANOTE 6 #A1 BNOTE ) QUIET
13|-->
14|
15|
```

```
  +--------Block    135--------
 0|( FOUL,CHEERS,BAT CRACK ) HEX BTABLE FOULSCORE 0 VIBS
 1|46 MASTER 0F ABVOLS 0 MCVOLS 4 #A2 ANOTE
 2|5 #G2 ANOTE 5 #F2 ANOTE 6 #E2 ANOTE
 3|6 #D2 ANOTE 7 #C2 ANOTE 7 #B1 ANOTE 8 #A1 ANOTE
 4|8 #G1 ANOTE 9 #F1 ANOTE A #E1 ANOTE B #D1 ANOTE QUIET
 5|( CROWD CHEER ) BTABLE CHEERSSCORE 35 MASTER 33 NOISE
 6|AA ABVOLS 3A MCVOLS 7C 50 40 2E NOTES 4 53 42 2F NOTES
 7|88 ABVOLS 38 MCVOLS 4 56 44 30 NOTES
 8|66 ABVOLS 36 MCVOLS 4 59 46 31 NOTES
 9|44 ABVOLS 34 MCVOLS 5 DURATION
10|22 ABVOLS 32 MCVOLS 7 DURATION QUIET
11|BTABLE CRACKSCORE 50 MASTER 3D NOISE
12|FF ABVOLS 1F MCVOLS #F4 BTONE #E4 ATONE
13|6 #G4 CNOTE QUIET
14|BTABLE FENCESCORE 48 MASTER 38 NOISE 4 CRACKSCORE LDPCC
15|-->
  +--------Block    136--------
 0|( CROWD CHEERS & TAKE ME O-T-T-B-G.) HEX
 1|{ : CMAJOR } 1C MASTER 0 NOISE EF ABVOLS F MCVOLS { ; }
 2|{ : REST } 0 ABVOLS 0 MCVOLS A DURATION { ; }
 3|{ : ^ } EF ABVOLS F MCVOLS { ; } BTABLE TAKE-ME CMAJOR 0 VIBS
 4|14 #C1 #E1 #C2 NOTES A #E1 #G1 #C3 NOTES A #C1 #F1 #A2 NOTES
 5|A #C1 #E1 #G2 NOTES A #C1 #G1 #E2 NOTES
 6|1E #B0 #D1 #G2 NOTES 1E #G0 #F1 #D2 NOTES 14 #C1 #E1 #C2 NOTES
 7|A #E1 #G1 #C3 NOTES A #F1 #C2 #A2 NOTES A #E1 #C2 #G2 NOTES
 8|A #C1 #G1 #E2 NOTES 3C #B0 #D1 #G2 NOTES 3C #C1 #E2 #C3 NOTES
 9|0 CHEERSSCORE LDPCC -->
10|A #C1 #G1 #E2 NOTES 14 #B0 #D1 #G2 NOTES A #C1 #E1 #G2 NOTES
11|A #D1 #F1 #G2 NOTES A #DS1 #C2 #FS2 NOTES A #E1 #C2 #G2 NOTES
12|A #F1 #C2 #A2 NOTES A #E1 #B1 3A NOTES A #F1 #C2 #A2 NOTES
13|A #G1 #C2 #E2 NOTES A #A1 #C2 #F2 NOTES A #E1 #CS2 #G2 NOTES
14|A #F1 #D2 #A2 NOTES REST ^ A #D1 #A1 #F2 NOTES -->
15|
  +--------Block    138--------
 0|( BASEBALL SOUNDS , SIREN CANNON ) HEX
 1|BTABLE SIRENSCORE AB ABVOLS 0B MCVOLS
 2|90 5E 11 RDRNDNTE 80 5E 13 RDRNDNTE 70 5E 12 RDRNDNTE
 3|0 VIBS 10 MASTER 1 DURATION 0E MASTER 1 DURATION 0C MASTER
 4|1 DURATION 0A MASTER 1 DURATION 8 MASTER 1 DURATION 6 MASTER
 5|1 DURATION 4 MASTER 1 DURATION 2 MASTER 1 DURATION
 6|12 SIRENSCORE LDPCC
 7|( HOMER CONT. ) BTABLE CANNONSCORE
 8|FF ABVOLS 1F MCVOLS 2 MASTER 8 NOISE
 9|#D2 CTONE #FS2 BTONE 6 #GS2 ANOTE
10|#C2 CTONE 6 #E2 ANOTE #B1 CTONE 6 #D2 BNOTE
11|DD ABVOLS 1A MCVOLS  #A1 CTONE 6 #C2 ANOTE
12|AA ABVOLS 17 MCVOLS  #G1 CTONE 6 #B1 BNOTE
13|0 SIRENSCORE LDPCC
14|-->
15|
```

```
 +--------Block    140--------
 0|( SAFE , OUT , ) HEX
 1|BTABLE SAFESCORE
 2|34 MASTER AB ABVOLS 0B MCVOLS 0 VIBS 5 #C2 #E2 #G2
 3|NOTES 5 #G2 #D3 #B2 NOTES 5 #A2 #D2 #C3 NOTES
 4|12 #B2 #D3 #G3 NOTES  0 CHEERSSCORE LDPCC
 5|BTABLE OUTSCORE
 6|23 MASTER AB ABVOLS 0B MCVOLS 0 VIBS 5 #G2 #D2 #F2
 7|NOTES 5 #E2 #B1 #D2 NOTES 5 #B1 #GS1 #F1 NOTES
 8| 5 #E1 #F1 #B0 NOTES
 9|12 #G0 #F1 #A1 NOTES QUIET
10|BASE ! ;S
11|
12|
13|
14|
15|
 +--------Block    141--------
 0|( MUSIC PROCESSOR COMANDS )  BASE@ HEX
 1|0 VARIABLE MUSPC { : MASTER } 10 B, B, { ; } { : ATONE } 11 B,
 2|B, { ; } { : BTONE } 12 B, B, { ; } { : CTONE } 13 B, B, { ; }
 3|{ : VIBS } 14 B, B, { ; }        { : ABVOLS } 16 B, B, { ; }
 4|{ : MCVOLS } 15 B, B, { ; }     { : NOISE } 17 B, B, { ; }
 5|{ : DURATION } 1 B, B, { ; }
 6|{ : LDPC } 3 B, , { ; } { : QUIT } 4 B, HERE LDPC { ; }
 7|{ : LDPCC } 2 B, , { ; }       { : QUIET } 0 ABVOLS
 8|0 MCVOLS 0 ATONE 0 BTONE 0 CTONE QUIT { ; }
 9|{ : ANOTE } ATONE DURATION { ; } { : BNOTE } BTONE DURATION
10|{ ; } { : CNOTE } CTONE DURATION { ; }
11|{ : RDRNDNTE } 0 B, B, B, B, { ; }
12|{ : RRNDNTE } 0 B, B, 0 B, B, { ; }
13|{ : RNDNTE } 0 B, B, 0 B, FF B, { ; }
14|{ : NOTES } ATONE BTONE CNOTE { ; }
15|BTABLE ENDMUSIC QUIET -->
 +--------Block    142--------
 0|( NOTE CONSTANTS )
 1|FD C= #G0  EE C= #GS0  E1 C= #A0  D4 C= #AS0  C8 C= #B0
 2|BD C= #C1  B2 C= #CS1  A8 C= #D1  9F C= #DS1  96 C= #E1
 3|8D C= #F1  85 C= #FS1  7E C= #G1  77 C= #GS1  70 C= #A1
 4|6A C= #AS1  64 C= #B1  5E C= #C2  59 C= #CS2  54 C= #D2
 5|4F C= #DS2  4A C= #E2  46 C= #F2  42 C= #FS2  3E C= #G2
 6|3B C= #GS2  37 C= #A2  34 C= #AS2  31 C= #B2  2E C= #C3
 7|2C C= #CS3  29 C= #D3  27 C= #DS3  25 C= #E3  22 C= #F3
 8|20 C= #FS3  1F C= #G3  1D C= #GS3  1B C= #A3  1A C= #AS3
 9|18 C= #B3  17 C= #C4  15 C= #CS4  14 C= #D4  13 C= #DS4
10|12 C= #E4  11 C= #F4  10 C= #FS4  0F C= #G4  0E C= #GS4
11|0D C= #A4  0B C= #C5  0A C= #CS5  09 C= #DS5 08 C= #F5
12|07 C= #G5  06 C= #A5  05 C= #C6  04 C= #DS6  03 C= #G6
13|02 C= #C7  01 C= #G7  00 C= #G8
14|{ : V } VARIABLE { ; } { : BV } BVARIABLE { ; }
15|-->
```

```
  +--------Block    143--------
 0|( MUSIC PROCESSOR IN ASSEMBLY )
 1|0 BV MULTIPLE 0 V STARTPC 0 BV NOTETIMER 0 BV PRIORITY
 2|SUBR FETCH MUSPC LHLD, M A MOV, RET,
 3|SUBR INCPC MUSPC LHLD, H INX, MUSPC SHLD, RET,
 4|SUBR HLOAD FETCH CALL, INCPC CALL, 0 H MVI, A L MOV, RET,
 5|SUBR PCJUMP MUSPC LHLD, M E MOV, H INX, M D MOV,
 6|MUSPC SDED, RET,
 7|( MUSIC PROCESSOR IN ICODE )
 8|FORWARD PROCESS FORWARD M1CASE FORWARD M2CASE FORWARD M3CASE
 9|FORWARD M4CASE FORWARD PORTOUT FORWARD MULTDN FORWARD MUSEND
10|CODE MUSCPU .ASSEMBLE
11|B PUSH, NOTETIMER LDA, A ORA, PROCESS JZ,
12|A DCR, NOTETIMER STA, MUSEND JNZ,
13|LABEL PROCESS FETCH CALL, INCPC CALL, A ORA,
14|M1CASE JNZ, HLOAD CALL, H PUSH, HLOAD CALL,
15|H PUSH, HLOAD CALL, H PUSH, DOVERB RND -->
  +--------Block    144--------
 0|( MUSIC PROCESSOR AS A CODED SUBROUTINE )
 1|H POP, D POP, D DAD, B POP, L OUTP, A XRA, MUSEND JMP,
 2|LABEL M1CASE A DCR, M2CASE JNZ, FETCH CALL,
 3|NOTETIMER STA, INCPC CALL, 1 ORI, MUSEND JMP,
 4|LABEL M2CASE A DCR, M3CASE JNZ, PCJUMP CALL,
 5|A XRA, MUSEND JMP,
 6|LABEL M3CASE A DCR, M4CASE JNZ, PCJUMP CALL,
 7|1 ORI, MUSEND JMP,
 8|LABEL M4CASE A DCR, PORTOUT JNZ, MULTIPLE LDA, A DCR,
 9|MULTIPLE STA, MULTDN JZ, STARTPC LHLD, MUSPC SHLD, MUSEND JMP,
10| LABEL MULTDN PRIORITY STA, MUSEND JMP,
11|LABEL PORTOUT 4 ADI, A C MOV, FETCH CALL, A OUTP,
12|INCPC CALL, A XRA,
13|LABEL MUSEND A ORA, PROCESS JZ, B POP, RET,
14|.END
15|-->
  +--------Block    145--------
 0|( MUSIC PROCESSOR CALLS )
 1|  SUBR loadpc MUSPC SHLD, STARTPC SHLD, RET,
 2|CODE BMUSIC PRIORITY LDA, H POP, A ORA, 0=, IF, loadpc CALL,
 3|  NOTETIMER STA, A INR, MULTIPLE STA, THEN, NEXT
 4|CODE EMUSIC 0 ENDMUSIC H LXI, loadpc CALL, 1 A MVI, NOTETIMER
 5|  STA, MULTIPLE STA, NEXT ( CALL EMUSIC AS AN INIT IN PROGRAM )
 6|CODE PMUSIC H POP, loadpc CALL, 1 A MVI, MULTIPLE STA,
 7|  PRIORITY STA, NOTETIMER STA, NEXT
 8|CODE MMUSIC H POP, PRIORITY LDA, A ORA, 0=, IF, loadpc CALL,
 9|  NOTETIMER STA, H POP, L A MOV, MULTIPLE STA,
10| ELSE, H POP, THEN, NEXT
11|CODE MPMUSIC 1 A MVI, PRIORITY STA, NOTETIMER STA,
12|  H POP, loadpc CALL, H POP, L A MOV, MULTIPLE STA, NEXT
13|BASE! ;S
14|
15|
```

```
 +--------Block    146--------
 0|( BB music calls )
 1|DECIMAL 141 LOAD 134 LOAD
 2|: MCANNON 0 CANNONSCORE BMUSIC ;
 3|: MBOMB MCANNON ;
 4|: MYAH 0 CHEERSSCORE BMUSIC ; : MRAH MYAH ;
 5|: MHIT 0 CRACKSCORE BMUSIC ; : MFENCE 0 FENCESCORE BMUSIC ;
 6|: MSTRIKE  0 STRIKESCORE BMUSIC ; : MBALL  0 BALLSCORE BMUSIC ;
 7|: MFOUL  0 FOULSCORE BMUSIC ;
 8|: MSAFE 0 SAFESCORE BMUSIC ; : MOUT 0 OUTSCORE BMUSIC ;
 9|: MOPENERS 0 TAKE-ME PMUSIC ;
10|: MCOIN 0 COINSCORE PMUSIC ;
11|;S
12|
13|
14|
15|
 +--------Block    147--------
 0|( BB hit sector constants )
 1|DECIMAL
 2|1 C= FLYB 0 C= FG
 3|0 C= FBL 1 C= HLL 2 C= F3RDR 3 C= F3RDL
 4|4 C= HL 5 C= FSSR 6 C= FSSL 7 C= HM
 5|8 C= F2NDR 9 C= F2NDL 10 C= HR 11 C= F1STR 12 C= F1STL
 6|13 C= HRL 14 C= FBR
 7|( bit equates ) 7 C= VACT 6 C= VATBS 5 C= VOF
 8|  4 C= VPT 3 C= VGO 2 C= VHW ( half way to next base )
 9|  1 C= VBL 0 C= VRUN 1 C= VAUTO 0 C= VFORW
10|-->
11|
12|
13|
14|
15|
 +--------Block    148--------
 0|( BB constants )
 1|  DECIMAL 0 C= VMAGIC 1 C= VANM# 2 C= VANML VANML C= VANM
 2|  3 C= VANMH 4 C= VANMSEQL VANMSEQL C= VANMSEQ 5 C= VANMSEQH
 3|  6 C= VPLAYACTPCL VPLAYACTPCL C= VPLAYACTPC 7 C= VPLAYACTPCH
 4|  8 C= VUPDATE# 10 C= VDXH 9 C= VDXL VDXL C= VDX 12 C= VXH
 5|  11 C= VXL VXL C= VX 14 C= VDYH 5 C= LDFLAG
 6|  13 C= VDYL VDYL C= VDY 16 C= VYH 15 C= VYL VYL C= VY
 7|  17 C= VSTATUS 18 C= VWBASE 20 C= VPATH
 8|  19 C= VPATL VPATL C= VPAT 21 C= VPERS#
 9|  22 C= VSCRADRL VSCRADRL C= VSCRADR 23 C= VSCRADRH
10|  24 C= VXPAND 25 C= VDSTXL VDSTXL C= VDSTX
11|  26 C= VDSTXH 27 C= VDSTYL VDSTYL C= VDSTY
12|  28 C= VDSTYH 29 C= VOFW 29 C= VVEL 31 C= VLENGTH
13|  30 C= VRSTAT
14|-->
15|
```

```
 +--------Block    149--------
 0|( BB variables)
 1|BV= TMP3 V= TMP2 BV= CMPLR BV= SWRND# BV= LINN BV= CNSW1
 2|  BV= PLYR1UP  BV= INN# BV= SCORE1 BV= SCORE2 BV= HRAT
 3|  BV= GAMEOVER BV= CREDITS BV= COINS V= INNX BV= UPCRED
 4|  BV= CNTM1 BV= CNTM2 BV= 1STINN BV= SHILL BV= CNSW12
 5|   ." VPTR SPEC- " VPTR @ H.
 6|   HEX 7C30 VPTR ! DECIMAL ( seperate ram )
 7|  BV= CMPT BV= CMSW BV= LBASE BV= WALK
 8|  BV= HITTYPE  BV= SECTNM BV= OFPU BV= HITGOING
 9|  BV= BASESTATUS  BV= OFTFCNT BV= THROW BV= THROWAROUND
10|  BV= OLDSTRING  BV= NEGDX  BV= SWINGTYPE BV= BASEBLAT
11|  BV= THRWBASE  BV= OFFLDANM BV= HITOF V= PLAYON
12|  BV= STRINGERASE  BV= DBLPLAY BV= OFTBLGO BV= INTFLAG
13|  BV= OUTS  BV= TAKEFIELD BV= OLDFORW V= WHOSUP
14|  BV= TOTIMER BV= TOOF BV= TOOFF BV= TOOF# BV= PTRND#
15|  BV= HOMER BV= FENCE  V= WHOTHROWS  BV= THROWANM -->
 +--------Block    150--------
 0|( BB variables)
 1|  BV= OFANM#  BV= OTBALLY  BV= TBALLDY
 2|  BV= TBALLYSR  BV= OTBALLX  BV= TBALLDX
 3|  V= THWPA  BV= THWUPDATE#  V= THWDY
 4|  V= THWDX  BV= THWOPCODE
 5|  V= RUNPA  BV= RUNUPDATE#  V= RUNDY
 6|  V= RUNDX  BV= RUNOPCODE
 7|  V= FLDPA  BV= FLDUPDATE#  V= FLDDY
 8|  V= FLDDX  BV= FLDOPCODE
 9|  V= OFPA  BV= OFUPDATE#  V= OFDY
10|  V= OFDX  BV= OFOPCODE
11|  V= TBLPA  BV= TBLUPDATE#  V= TBLDY
12|  V= TBLDX  BV= TBLOPCODE
13|  V= OFFPA  BV= OFFUPDATE#  V= OFFDY
14|  V= OFFDX  BV= OFFOPCODE
15|-->
 +--------Block    151--------
 0|( BB variables)
 1|V= TMPPITCH BV= STRT1 BV= DOWUP
 2|  V= VECTPC BV= TFTIMER V= FNCX V= FNCY
 3|   V= VECTIX  BV= STRIKES BV= BALLS BV= HOMERUN
 4|  BV= SCORESHOW V= CMFLDR BV= OFCRUN BV= FLSHSTAY BV= WUPFLSH
 5|  BV= NOCATCH BV= WAITTHROW BV= STRING BV= STRINGOFFTIMER
 6|  BV= STRIKE BV= WLKCNT BV= SPBON BV= OFTOTBL
 7|  BV= SAMEDLT BV= FLDCLR BV= TFTIME BV= ICPTYP V= FLSHWHO
 8|  BV= TEMPV BV= IODDS BV= NORUN BV= INSOUT
 9|  V= BASEX  V= BASEY BV= DBLFLAG
10|  BV= NOBLWRT BV= OFCATCH BV= OFOUT
11|  BV= FLSHCNT  BV= FLSHON BV= FLSHTIME
12|  BV= SWINGGO BV= STMUSIC BV= CHEERME BV= HMRFLSH
13|  BV= TIMEOUT  BV= HMRCNT BV= HMFLSHCNT
14|  BV= GOSEQCNT BV= GOSEQ
15|-->
```

```
 +--------Block    152--------
 0|( BB variables)
 1| BV= CMOFTIMER BV= CHKPLRS BV= SELECT
 2|  BV= WHICHINT BV= BLWAIT
 3| BV= BLTIME  BV= RDST
 4| V= THROWTIMER  V= THROWTIME
 5| BV= INAIR
 6| BV= ATBASE  V= FLDON1ST  V= FLDON2ND
 7| V= FLDON3RD   BV= SCRTIME
 8| BV= GRNDR BV= GRNDRVALUE
 9| BV= DPHIT BV= HLSIDE BV= HITDEEP BV= FOUL
10| BV= HITTIME  BV= COMMITED
11| BV= HITYET  BV= CAUGHT
12| BV= SWING  BV= SWINGTIME
13| BV= FORW  BV= PITCHIT
14| BV= PITCHTIME
15|;S
 +--------Block    153--------
 0|( infield logic loop      LDINFLDPA , )
 1|BASE@ HEX
 2|BV= HTMP BV= HMTMP BV= SWINGTMP BV= WTMP BV= DPTMP
 3|: LDINFLDPA VPLAYACTPC 3RD ! VPLAYACTPC SS ! VPLAYACTPC 2ND !
 4|    VPLAYACTPC 1ST ! VPLAYACTPC PT ! ;
 5|-->
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    154--------
 0|( BB infield logic    HITDST , FHTBL , GHTBL , HTBL , HMHTBL )
 1|: HITDST ( chooses hit location ) VUPDATE# BL BZERO
 2|  THWBLPA VPLAYACTPC BL ! 82 VSTATUS BL B! ( vact )
 3|  BAL THWPA ! 2800 VX BL ! AD00 VY BL ! ;
 4|TABLE FHTBL      ( fly ball hit table )
 5|  4000 , 3000 , 2000 , 1000 , 200 ,
 6|TABLE GHTBL   ( grounder hit table )
 7|  200 , 800 , 1000 , 1600 , 1800 , 2400 , 2800 , 2E00 ,
 8|  3700 , 3A00 , 3F00 , 4700 , 4E00 ,
 9|BTABLE HMHTBL  ( fence = 1 homer = 2 normal = 0 )
10|  2 1 1 1 1 1 0 0 0 0 &STF B, B,
11|-->
12|
13|
14|
15|
```

```
 +--------Block    155--------
 0|( BB infield action by sector )
 1|: FBLLS NOACTPA FBLL3RD
 2|   1 THROWAROUND BONE FOUL BONE ;
 3|: HLLS HITOF BONE NOACTPA HLL3RD 1 ;
 4|: F3RDRS NOACTPA F3RDR3RD 1 ;
 5|: F3RDLS F3RDLSS F3RDL3RD 1 ;
 6|: HLS HITOF BONE HLSS HL3RD 1 ;
 7|: FSSRS DPHIT BONE FSSRSS HR3RD 1 ;
 8|: FSSLS DPHIT BONE FSSLSS HR3RD 1 ;
 9|: HMS HITOF BONE HMSS HR3RD 1 ;
10|: F2NDRS DPHIT BONE HRPT HL1ST F2NDR2ND 0 ;
11|: F2NDLS DPHIT BONE HRPT HL1ST F2NDL2ND 0 ;
12|-->
13|
14|
15|
 +--------Block    156--------
 0|( BB infield action by sector   GRNDRSTABLE )
 1|: HRS HITOF BONE HRPT HR1ST HR2ND 0 ;
 2|: F1STRS F1STPT F1STR1ST NOACTPA 0 ;
 3|: F1STLS F1STPT F1STL1ST NOACTPA 0 ;
 4|: HRLS HITOF BONE HRPT HRL1ST NOACTPA 0 ;
 5|: FBLRS FOUL BONE THROWAROUND BONE NOACTPA
 6|   FBLR1ST NOACTPA 0 ;
 7|TABLE GSTBL  ( grnder fielder action by sector )
 8|   ' FBLLS , ' HLLS , ' F3RDRS , ' F3RDLS ,
 9|   ' HLS , ' FSSRS , ' FSSLS , ' HMS ,
10|   ' F2NDRS , ' F2NDLS , ' HRS , ' F1STRS ,
11|   ' F1STLS , ' HRLS , ' FBLRS ,
12|-->
13|
14|
15|
 +--------Block    157--------
 0|( BB infield action by sector    GRNDRACTION )
 1|: GRNDRACTION ( given sectnm set up infield action )
 2|   ( default values ) HITOF BZERO DPHIT BZERO
 3|   FOUL BZERO
 4|     GSTBL @ EX ( ret hlside and playact )
 5|   DUP HLSIDE B! IF VPLAYACTPC 3RD ! VPLAYACTPC SS !
 6|   HLPT VPLAYACTPC PT ! HL1ST VPLAYACTPC 1ST ! HL2ND
 7|   VPLAYACTPC 2ND ! ELSE ( hit right side )
 8|   HRSS HR3RD LDINFLDPA THEN ;
 9|-->
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    158--------
 0|( BB infield action      GRNDRHIT , CHKCN )
 1|: GRNDRHIT  ( grounder ) GRNDR BONE 2 VVEL BL B!
 2|3 RND ( waited hit constant ) VDX BL 1+ B@ (  1 , 0 -1 )
 3|  SWINGTYPE B@ SWINGTABLE @ EX DUP DUP DUP
 4|  SECTNM B! GRNDRACTION
 5|  FOUL B@ IF MFOUL 1 THROWAROUND B! CFOUL STRING B!
 6|  HLSIDE B@ IF 800 ELSE 4800 THEN VDSTX BL !
 7|  6400 VDSTY BL ! DROP DROP
 8|  ELSE ( fair ) WUPGO HITOF B@ IF 10 TOTIMER B!
 9|  TOOF BONE 3 TOOF# B!
10|  CMPT B@ IF 35 CMOFTIMER B! THEN ELSE HITRUN LBASE BONE THEN
11|  1- GHTBL @ VDSTX BL ! HM = IF 2C00 ELSE 3C00 THEN
12|    VDSTY BL ! THEN HITDST ;
13| LABLE CNTBL  2800 , 2000 , 300 ,
14|CODE CHKCN   ( set up corner type )
15|  C A MOV, CNTBL H LXI, ' INDEXW CALL, RET, )
 +--------Block    159--------
 0|( BB  fence hit check does homer set up    HOMERCHK )
 1|CODE HOMERCHK   ( chk for homer or fence hit )
 2|  EXX, PSW POP, 5 ADI, FNCY ( 1+ ) STA, ( 5 rnd +5 for fncy )
 3|  PSW POP, 4 ADI, A B MOV, ( 5 rnd +5 for fncx )
 4|  H POP, L A MOV, ( hit type ) H POP, ( x ) D POP, ( y )
 5|  A ANA, 0< >, IF, 1 CPI, =, IF, FENCE STA, 500 D LXI, ELSE,
 6|  HOMER STA, 100 D LXI, THEN, THEN,
 7|  A C MOV, D A MOV, 28 CPI, <, IF, H A MOV, 0A CPI, ( lcn )
 8|  <, IF, CNTBL H LXI, C A MOV, ' INDEXW CALL,
 9|        B H MOV, FNCX SHLD, 1C0 H LXI, ( lft corner )
10|  ELSE, 46 CPI, >=, IF, CNTBL H LXI, C A MOV, ' INDEXW CALL,
11|         B A MOV, NEG, FNCX ( 1+ ) STA,
12|         4E00 H LXI, ( hit rt corner )
13|-->
14|
15|
 +--------Block    160--------
 0|( BB infield action )
 1|  ELSE, SWINGTYPE LDA, A DCR, 0=, IF, B A MOV, NEG,
 2|  ELSE, 4 CPI, =, IF, B A MOV,
 3|  ELSE, B A MOV,
 4|          0 L BIT, 0=, IF, NEG, THEN, THEN, THEN,
 5|  FNCX ( 1+ ) STA, THEN,
 6|  THEN, THEN, VDSTX BL SHLD, VDSTY BL SDED, EXX, NEXT -->
 7|-->
 8|
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    161--------
 0|( BB infield action      OUTFLDHIT )
 1|: OUTFLDHIT ( fly ball ) 18 TOTIMER B! HITOF BONE 3 VVEL BL B!
 2|  TOOF BONE 3 TOOF# B! INAIR BONE 3000 RND 700 + ( y )
 3|  SWINGTYPE B@ 1- FHTBL @ F00 RND + ( x )
 4|  0A RND HMHTBL B@ ( hit type ) 5FF RND 5FF RND
 5|  HOMERCHK WUPGO
 6|  HITDST HRPT HL1ST VDSTX BL @ 2800 < IF ( hit left )
 7|  HL2ND HLOUTSSPA ELSE HROUT2NDPA HRSS THEN HR3RD LDINFLDPA ;
 8|
 9|-->
10|
11|
12|
13|
14|
15|
 +--------Block    162--------
 0|( BB infield action      INLOG )
 1|: INLOG ( infield master logic ) NOCATCH BZERO LBASE BZERO
 2|  OFTBLGO BONE WLKCNT BZERO NORUN BZERO
 3|  MHIT VMAGIC R1 PLAYON ! GRNDR BZERO HOMER BZERO DBLFLAG BZERO
 4|  OFPU BZERO FENCE BZERO INAIR BZERO SWINGTYPE B@ 6 = IF 0
 5|  ELSE HRAT B@ 0A RND < IF 0 ( flyb ) ELSE 1 ( grndr ) THEN
 6|  DUP HITTYPE B! THEN
 7|  IF OUTFLDHIT CMPT B@ IF 30 CMOFTIMER B! THEN
 8|  ELSE GRNDRHIT THEN ALLFLDACT ;
 9|BASE! ;S
10|
11|
12|
13|
14|
15|
 +--------Block    163--------
 0|( BB hit run and throw for outfield fly )
 1|SUBR FNBB RET,
 2|SUBR FR01  AUTOR1 CALL, RET,
 3|SUBR FR02 VSTATUS R1 STA, RET,
 4|SUBR FR03 VSTATUS R1 STA, RET,
 5|SUBR FR012  AUTOR1 CALL,  AUTOR2 CALL,  RET,
 6|SUBR FR023 VSTATUS R1 STA, VSTATUS R2 STA, RET,
 7|SUBR FR0123 AUTOR1 CALL, AUTOR2 CALL, AUTOR3 CALL, RET,
 8|SUBR FR013 VSTATUS R1 STA, AUTOR2 CALL, RET,
 9|LABLE WESFTBL
10|  FNBB , FR01 , FR02 , FR03 , FR012 , FR0123 , FR013 , FR023 ,
11|-->
12|
13|
14|
15|
```

```
 +--------Block    164--------
 0|( BB hit run and throw for outfield fly   TAKEOFF )
 1|HEX
 2|F= TORET
 3|SUBR TAKEOFF   ( turn proper runners on ) .ASSEMBLE
 4|  BASESTATUS LDA, WBSFTBL H LXI, ' INDEXW CALL, TORET H LXI,
 5|  81 A MVI, ( vact & vrun ) H PUSH, ( ret add.) XCHG, PCHL,
 6|LABEL TORET RET, .END
 7|-->
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    165--------
 0|( BB hit run and throw for infield grounder )
 1|SUBR SETDBLPLAY ( sets up double play HL-guy on 1st )
 2| 1 A MVI, DBLPLAY STA, RET,
 3|SUBR NBB  RET,
 4|SUBR R01  AUTOR1 CALL,  DPHIT LDA, A ANA,
 5|  SETDBLPLAY JNZ, VMAGIC R2 B LXI, RET,
 6|SUBR R02 HLSIDE LDA, A ANA, 0=, IF,
 7|    AUTOR1 CALL, THEN, VMAGIC R2 B LXI, RET,
 8|SUBR R03  VMAGIC R2 B LXI, DPHIT LDA, A ANA,
 9|   AUTOR1 CNZ, RET,
10|SUBR R012  AUTOR1 CALL,  AUTOR2 CALL,
11|  VMAGIC R2 B LXI, DPHIT LDA, A ANA, SETDBLPLAY JNZ,
12|  VMAGIC R3 B LXI, RET,
13|-->
14|
15|
 +--------Block    166--------
 0|( BB hit run and throw for infield grounder )
 1|SUBR R023  VMAGIC R3 B LXI, DPHIT LDA, A ANA,
 2|  0< >, IF,  AUTOR1 CALL, HLSIDE LDA, A ANA,  AUTOR2 CZ, THEN,
 3|  RET,
 4|SUBR R0123 AUTOR1 CALL,  AUTOR2 CALL,  AUTOR3 CALL,
 5|   VMAGIC R3 B LXI, DPHIT LDA, A ANA,
 6|   SETDBLPLAY JNZ, VMAGIC R1 B LXI, RET,
 7|SUBR R013  AUTOR2 CALL,  DPHIT LDA, A ANA,
 8|  VMAGIC R2 B LXI,
 9|  0< >, IF,  AUTOR1 CALL,  SETDBLPLAY JMP, THEN,
10|  VMAGIC R3 B LXI, RET,
11|LABLE WBASETABLE
12|  NBB , R01 , R02 , R03 , R012 , R0123 , R013 , R023 ,
13|-->
14|
15|
```

```
 +--------Block    167--------
 0|( BB hit run and throw for infield grounder )
 1|FORWARD HRRET
 2|CODE HITRUN  ( activate proper runners and set trwbase )
 3|    A XRA, OFTBLGO STA,  ( runners wont advance 2 bases )
 4|  B PUSH, ( set playon by which runners go defaults to r1 )
 5|.ASSEMBLE  BASESTATUS  LDA, VMAGIC R1 B LXI, ( default value )
 6|  WBASETABLE H LXI, ' INDEXW CALL, HRRET H LXI,
 7|  SECTNM LDA, H PUSH,  ( ret add. ) XCHG, PCHL,
 8|LABEL HRRET PLAYON SBCD, B POP, NEXT .END
 9|DECIMAL ;S
10|
11|
12|
13|
14|
15|
 +--------Block    168--------
 0|( BB who's on which base    WHONBASEODDS )
 1|: RUN3CHK VWBASE R3 B@ 0 = IF 7 VMAGIC R3 ELSE 5 VMAGIC R4
 2|  THEN ;
 3|: RUN2CHK1ST VWBASE R2 B@ 1 = IF 6 VMAGIC R3
 4|     ELSE RUN3CHK THEN ;
 5|: RUN2CHK VWBASE R2 B@ 0 = IF 3 VMAGIC R2
 6|     ELSE RUN2CHK1ST THEN ;
 7|: RUN2CHK12ND VWBASE R2 B@ 0 = IF 2 VMAGIC R2
 8| ELSE 4 VMAGIC R3 THEN ;
 9|: RUN1CHK2ND VWBASE R1 B@ 2 = IF RUN2CHK12ND
10|     ELSE RUN2CHK THEN ;
11|: RUN1CHK1ST VWBASE R1 B@ 1 = IF 1 VMAGIC R2
12|     ELSE RUN1CHK2ND THEN ;
13|-->
14|
15|
 +--------Block    169--------
 0|( BB who's on which base    WHBODDS )
 1|: HRBY2 HRAT B@ 2 / HRAT B! ;
 2|: WHBODDS   ( sets who is on base calculates play odds )
 3|  VWBASE R1 B@ 0 = IF 0 VMAGIC R1
 4|  ELSE RUN1CHK1ST THEN WHOSUP ! BASESTATUS B!
 5|  36 RND SWRND# B! 10 RND PTRND# B!
 6|   ( odds logic )
 7|  IODDS B@ HRAT B! SCORE2 B@ SCORE1 B@
 8|  PLYR1UP B@ IF SWAP THEN -
 9|  DUP 0 < IF -3 > IF 7 ELSE 8 THEN HRAT B!
10|  ELSE  ( player up ahead )
11|  DUP 2 > IF HRBY2 DUP 4 > IF HRBY2 DUP 6 > IF HRBY2
12|  THEN THEN THEN DROP THEN ;
13|;S
14|
15|
```

```
 +--------Block    170--------
 0|( BB hit logic )
 1|( given dx and rnd waited constant )
 2|: SW-90 DROP DROP FBR ;
 3|: SW-45 DUP IF 1 = IF ( x=1 ) DROP FBR
 4|   ELSE ( x = -1 ) IF HRL ELSE F1STL THEN THEN
 5|   ELSE ( X=0 ) DROP DUP IF 1- IF F1STL ELSE F1STR THEN
 6|                 ELSE DROP HRL THEN THEN ;
 7|: SW-30 DUP IF 1 = IF ( x=1 ) IF F1STR ELSE HR THEN
 8|   ELSE ( x=-1 ) IF HR ELSE F2NDL THEN THEN
 9|   ELSE ( x= 0 ) DROP DUP IF 1- IF F2NDL ELSE F2NDR THEN
10|                  ELSE DROP HR THEN THEN ;
11|: SW0 DUP IF 1 = IF ( x=1 ) IF F2NDR ELSE HM THEN
12|   ELSE ( x=-1 )  IF FSSL ELSE HM THEN THEN
13|   ELSE ( x=0 ) DROP IF HM ELSE F2NDR THEN THEN ;
14|-->
15|
 +--------Block    171--------
 0|( BB hit logic   SWINGTABLE )
 1|: SW+30 DUP IF 1 = IF ( x=1 ) IF HL ELSE FSSR THEN
 2|   ELSE ( x= -1 ) IF F3RDL ELSE HL THEN THEN
 3|   ELSE ( x=0 ) DROP DUP IF 1- IF FSSR ELSE FSSL THEN
 4|                 ELSE DROP HL THEN THEN ;
 5|: SW+45 DUP IF 1 = IF ( x=1 ) IF HLL ELSE F3RDR THEN
 6|   ELSE ( x= -1 ) DROP FBL THEN
 7|   ELSE ( x=0 ) DROP DUP IF 1- IF F3RDR ELSE F3RDL THEN
 8|                 ELSE DROP HLL THEN THEN ;
 9|: SW+90 DROP DROP FBL ;
10|TABLE SWINGTABLE
11|' SW-90 , ' SW-45 , ' SW-30 , ' SW0 ,
12|' SW+30 , ' SW+45 , ' SW+90 ,
13|;S
14|
15|
 +--------Block    173--------
 0|( BB outfield computer control     CMPOF )
 1|HEX
 2|SUBR CMPOF   ( calculates proper deltas and patterns )
 3|   DI, OFTBLGO LDA, A ANA, RZ, X PUSHX,
 4|   VDSTXH BL LDA, 1C CPI, >=, IF, 34 CPI, >=, IF, 0 RF X LXIX,
 5|   ELSE, 0 CF X LXIX, THEN, ELSE, 0 LF X LXIX, THEN,
 6|   CMFLDR SIXD,
 7|   VDSTX BL LHLD, VDSTY BL LDED, 2 A MVI, VGO VSTATUS X RESX,
 8|   ' DSTCALC CALL, OFDX SDED, OFDY SHLD, OFFA SBCD,
 9|   ' OUTFLDACT CALL, EI, X POPX, RET,
10|SUBR TOOFCK     ( time out outfield check )
11|   TOOF# H LXI, M DCR, 0<>, IF, FF A MVI, TOOF STA,
12|   ELSE, CMFOF CALL, EI, 1 A MVI, TOOFF STA, THEN, RET,
13|DECIMAL ;S
14|
15|
```

```
 +--------Block    174--------
 0|( BB tractor ball logic      TBALLPRC )
 1|FORWARD NZPDX FORWARD LDOFPA BASE@ HEX
 2|CODE TBLVEL  ( tractorball vel HL-# add. D-max vel )
 3|  M A MOV, 0 H LXI, A ANA, RZ,
 4|  7 A BIT, PSW PUSH, 0< >, IF, NEG, THEN, ( abs. )
 5|  3 CPI, <, IF, 80 H LXI, ELSE, 1B0 H LXI, THEN
 6|  PSW POP,  ' COMPHL CNZ, RET,
 7|CODE TBALLVEL  ( calculates tball vel A-actual val HL-old val)
 8|   PSW PUSH, M SUB, ( new - old )
 9|     H INX, A M MOV,      H DCX, PSW POP, A M MOV, RET,
10|CODE TBALLPRC ( pitch or control outfield ) .ASSEMBLE
11|(   12 IN, NEG, A D MOV, 13 IN, NEG, ) ( x , y )
12|  11 IN, A D MOV, 10 IN, NEG,
13|  OTBALLY H LXI, ' TBALLVEL CALL,
14|  D A MOV, OTBALLX H LXI, ' TBALLVEL CALL, -->
15|
 +--------Block    175--------
 0|( BB outfield tractor ball running algorithm )
 1|  CMPT LDA, A ANA, RNZ, OFTBLGO LDA, A ANA, RZ,
 2|  TOOFF LDA, A ANA, RNZ,
 3|  TBALLDY H LXI, 1 D MVI, ' TBLVEL CALL, OFDY SHLD, H PUSH,
 4|   TBALLDX H LXI, 1 D MVI,  ' TBLVEL CALL,
 5|    OFDX SHLD, D POP, ( ofdy )
 6|-->
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    176--------
 0|( BB out fielder animation logic )
 1|    7 H BIT, 0=, IF, A XRA, H CMP, NZPDX JNZ,
 2|                     D CMP, NZPDX JNZ, E CMP, NZPDX JNZ,
 3|                     L CMP, NZPDX JNZ, STND H LXI, ( zero delt)
 4|                     LDOFPA JMP,
 5|            LABEL NZPDX RRUN H LXI,   ( pos dx )
 6|                 ELSE, LRUN H LXI,     ( neg dx )
 7|                 THEN, H PUSH, ' OUTFLDACT CALL, H POP,
 8|            LABEL LDOFPA  OFPA SHLD, RET, .END
 9|-->
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    177--------
 0|( BB pitching algorithm      TBALLPITCH )
 1|CODE TBALLPITCH  ( loads pitch velocities )
 2|    TBALLYSR H LXI, TBALLDY LDA, 7 A BIT, PSW PUSH, ( sign )
 3|    M SUB, EXAF, ( save sub )  PSW POP, ( sign )
 4|     0>=, IF, EXAF, ( subtraction )
 5|       0>=, IF, M ADD, A M MOV, 3 CPI, >=, IF, 3 A MVI, THEN,
 6|        ( max = 3 ) A M MOV, THEN, ( if pos and > dy = sr )
 7|     ELSE, ( neg dy ) M A MOV, 2 CPI,
 8|         >=, IF, M DCR, THEN, ( if neg and sr >= 2 dcr sr )
 9|      THEN,  M A MOV, A H MOV, 0 L MVI, DI, TBLDY SHLD,
10|   TBALLDX H LXI, 0 D LXI, WLKCNT LDA, 2 CPI, <, IF,
11|   M A MOV, A ANA,
12|   0<>, IF, 7 A BIT, 0=, IF, C0 D LXI, ELSE, FF40 D LXI,
13|   THEN, THEN, THEN, TBLDX SDED, EI, RET,
14|BASE! ;S
15|
 +--------Block    178--------
 0|( BB short subroutines string routines )
 1|BASE@ HEX
 2|: 2ROT ROT ROT ;
 3|: SSAFE 2400 2ROT A" SAFE " SPOST SB0 ;
 4|: SFOUL 2400 2ROT A" FOUL " SPOST ;
 5|: SOUT1 2300 2ROT A" 1 OUT" SPOST ;
 6|: SOUT2 2200 2ROT A" 2 OUTS" SPOST ;
 7|: SOUT3 2200 2ROT A" 3 OUTS" SPOST ;
 8|: SSTRIKE1 2000 2ROT A" STRIKE 1" SPOST ;
 9|: SSTRIKE2 2100 2ROT A" STRIKE 2" SPOST ;
10|-->
11|
12|
13|
14|
15|
 +--------Block    179--------
 0|( BB short subroutines string routines )
 1|: SBALL1 2200 2ROT A" BALL 1" SPOST ;
 2|: SBALL2 2200 2ROT A" BALL 2" SPOST ;
 3|: SBALL3 2200 2ROT A" BALL 3" SPOST ;
 4|: SHOMER 2300 2ROT A" HOMER" SPOST ;
 5|: SWALK 2400 2ROT A" WALK" SPOST ;
 6|  6 C= CSAFE 1 C= CFOUL 2 C= COUT 3 C= CSTRIKE 4 C= CBALL
 7| 5 C= CHOMER 7 C= CWALK
 8|: DOSOUT OUTS B@ DUP 31 = IF DROP SOUT1 ELSE 32 = IF SOUT2
 9|  ELSE SOUT3 THEN THEN ;
10|: DOSSTRIKE STRIKES B@ 31 = IF SSTRIKE1 ELSE SSTRIKE2 THEN ;
11|: DOSBALL BALLS B@ DUP 31 = IF DROP SBALL1 ELSE 32 = IF SBALL2
12|  ELSE SBALL3 THEN THEN ;
13|CODE CLRBLST 30 A MVI, BALLS STA, STRIKES STA, NEXT
14|-->
15|
```

```
 +--------Block    180--------
 0|( BB    BALLERASE , DOHOMER , DOEHOMER )
 1|CODE BALLERASE B PUSH, Y PUSHX, BLERASE CALL,
 2|  Y POPX, B POP, NEXT
 3|: DOHOMER BALLERASE SHOMER CLRBLST SBO STRINGOFFTIMER BZERO ;
 4|: DOEHOMER SHOMER ;
 5|-->
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    181--------
 0|( BB string routines    OUTTIME )
 1|: SGO 2000 2D00 828 A" GAME OVER" SPOST ;
 2|: OUTTIME   ( out prc )
 3|  OUTS DUP 1+B! B@ 33 =  IF MYAH
 4|  VMAGIC PT 08 0 DO DUP DUP VPLAYACTPCL + OFFFPA S!
 5|  VSTATUS + DUP B@ 80 OR F7 AND SB! VLENGTH + LOOP
 6|  DROP  TAKEFIELD BONE
 7|  STRIKES B@ 33 = IF BALLERASE THEN
 8|  VMAGIC R1 4 0 DO DUP DUP DUP VPLAYACTPCL + OFFFPA S!
 9|  VSTATUS + DUP B@ F7 AND ( res vgo ) SB!
10|   VWBASE + B@  IF DUP ( set active if onbase )
11|  VSTATUS + DUP B@ 80 OR SB! THEN VLENGTH + LOOP DROP
12|  PLYR1UP B@ IF INN# B@ LINN B@ = IF
13|  SGO GAMEOVER BONE FLSHOFF THEN THEN
14|  ELSE MOUT   THEN CLRBLST SBO DOSOUT ;
15|-->
 +--------Block    182--------
 0|( BB short subroutines   STRIKETIME , BALLTIME , FOULTIME )
 1|: STRIKETIME   ( strike process )
 2|  STRIKES DUP 1+B! B@ 33 = IF OUTTIME COUT OLDSTRING B!
 3|  ELSE MSTRIKE DOSSTRIKE THEN SBO ;
 4|
 5|: BALLTIME   ( balls process )
 6|  BALLS DUP 1+B! B@ 34 = IF SWALK MRAH WLKCNT 1+B!
 7|  CWALK OLDSTRING B! LBASE BONE 40 WALK B!
 8|   1 HLSIDE B! WHBODDS HITRUN ( start runners ) WUPGO
 9|  CLRBLST ELSE MBALL DOSBALL THEN SBO ;
10|
11|: FOULTIME   ( foulball process )
12|  SFOUL STRIKES DUP B@ 32 = IF DROP
13|  ELSE 1+B! SBO THEN ;
14|-->
15|
```

```
 +--------Block    183--------
 0|( BB short subroutines SCOREME , STTBL , STETBL )
 1|: DOSAFE MSAFE SSAFE ;
 2|TABLE STETBL   ( string erase table )
 3|   ' SSAFE , ' SFOUL , ' DOSOUT , ' DOSSTRIKE , ' DOSBALL ,
 4|   ' DOEHOMER , ' SSAFE , ' SWALK ,
 5|TABLE STTBL  ( string prc table )
 6|  ' SSAFE , ' FOULTIME , ' OUTTIME ,
 7|    ' STRIKETIME ,    ' BALLTIME , ' DOHOMER , ' DOSAFE ,
 8|: SCOREME   ( plop scores up )
 9|    SCORESHOW DUP B@ 2 = IF 900 8C00 4308 SCORE2
10|   ELSE 4300 8C00 4308 SCORE1 THEN 1 NPOST BZERO
11|   HOMERUN B@ IF MBOMB ELSE MRAH THEN ;
12|-->
13|
14|
15|
 +--------Block    184--------
 0|( BB string routines  STRINGGO , STRINGPRC )
 1|: STRINGGO ( given string , stringerase )
 2|   7A00 ( y ) 828 ( ex/mg ) 4 PICK 4 PICK
 3|   IF STETBL ELSE
 4|   STTBL 50 STRINGOFFTIMER B! THEN @ EX
 5|   DROP DROP ;
 6|: STRINGPRC  ( string display )
 7|   STRING B@ DUP STRING BZERO
 8|   STRINGOFFTIMER B@ IF OLDSTRING B@ 1 STRINGGO OLDSTRING B!
 9|     0 ( erase or write ) STRINGGO
10|   ELSE OLDSTRING B!  ( one string ) STRINGERASE B@ STRINGGO
11|   THEN STRINGERASE BZERO ;
12|BASE! ;S
13|
14|
15|
 +--------Block    186--------
 0|( BB pattern tables )
 1|LABLE RUNUP0    RUP0B ,  RUP0M ,
 2|LABLE RUNUP1    RUP1B ,  RUP1M ,
 3|LABLE RUNUP2    RUP2B ,  RUP2M ,
 4|LABLE RUNUP3    RUP3B ,  RUP3M ,
 5|LABLE RUNUP4    RUP4B ,  RUP4M ,
 6|LABLE FLDUP1    FUP1B ,  FUP1M ,
 7|LABLE FLDUP2    FUP2B ,  FUP2M ,
 8|LABLE NOBOD0    NOBOD ,  NOBOD ,
 9|LABLE BALL0     BALLPAT , BALLPAT ,
10|LABLE ONBS1     ONBASE1 , ONBASE1 ,
11|LABLE ONBS2     ONBASE2 , ONBASE2 ,
12|-->
13|
14|
15|
```

```
 +--------Block    187--------
 0|( BB pattern tables )
 1|LABLE THRWUP1  TUP1M ,  TUP1F ,
 2|LABLE THRWUP2  TUP2M ,  TUP2F ,
 3|LABLE THRWUP3  TUP3M ,  TUP3F ,
 4|LABLE THRWUP4  TUP4M ,  TUP4F ,
 5|LABLE PTTHW1   PTMID ,  TUP1F ,
 6|LABLE STCB1    TUP1M , TUP1M ,
 7|LABLE STCB2    TUP2M , TUP2M ,
 8|-->
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    188--------
 0|( BB pattern tables )
 1|LABLE STND1   STN1 ,  STN1 ,
 2|LABLE STND2   STN2 ,  STN2 ,
 3|LABLE STND3   STN3 ,  STN3 ,
 4|LABLE STND4   STN4 ,  STN4 ,
 5|LABLE STPT1   STNPT ,  STNPT ,
 6|LABLE CVRBUP1   CBUP1 ,  CBUP1 ,
 7|LABLE CVRBUP2   CBUP2 ,  CBUP2 ,
 8|
 9|LABLE BALLM BALL0 , BALL0 , BALL0 , BALL0 , BALL0 ,
10|LABLE STCB STCB1 , STCB1 , STCB2 ,
11|LABLE STNDPT STPT1 , STPT1 ,
12|LABLE PTPT   PTTHW1 , PTTHW1 ,
13|LABLE ONBASE ONBS1 , ONBS1 , ONBS2 ,
14|-->
15|
 +--------Block    189--------
 0|( BB pattern table matrix   PATTERNS )
 1|LABLE CVRBUP  CVRBUP1 ,  CVRBUP1 ,  CVRBUP2 ,
 2|LABLE THRWUP  THRWUP1 ,  THRWUP1 ,  THRWUP2 ,
 3|              THRWUP3 ,  THRWUP4 ,
 4|LABLE NOBODY  NOBOD0 ,  NOBOD0 ,  NOBOD0  ,
 5|              NOBOD0 ,  NOBOD0 ,
 6|LABLE RUNUP  RUNUP0 ,  RUNUP1 ,  RUNUP2 ,
 7|              RUNUP3 ,  RUNUP4 ,
 8|LABLE FLDUP  FLDUP1 ,  FLDUP1 ,  FLDUP2 ,
 9|LABLE STNDS  STND1 ,  STND1 ,  STND2 ,
10|              STND3 ,  STND4 ,
11|LABLE PATTERNS      ( MAIN MATRIX )
12| RUNUP ,  FLDUP ,  THRWUP ,  CVRBUP ,  STNDS ,
13| NOBODY , BALLM , STCB , STNDPT ,
14| PTPT , ONBASE ,
15|;S
```

```
 +--------Block    190--------
 0|( BB op codes for playaction defined   BSRTBL , DEACTIVATE )
 1|BASE@ HEX
 2|LABLE BSRTBL 3E80 , 6100 , 2800 , 4700 , 11C0 , 6300 ,
 3|   2800 , A600 ,
 4|CODE DEACTIVATE  ( reset active bit in status )
 5|  VACT VSTATUS X RESX, 1 A MVI, SAMEDLT STA, RET,
 6|-->
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    191--------
 0|( BB op codes for playaction defined   THWANMSET )
 1|CODE THWANMSET  ( set up throw animation )
 2|  X PUSHX, H POP, WHOTHROWS SHLD,   ( save who throws )
 3|  BLERASE CALL, 1 A MVI, INAIR STA,
 4|    THROWANM STA, RET,  ( set for sentry call )
 5|CODE ON1ST X PUSHX, H POP, FLDON1ST SHLD, RET,
 6|CODE ON2ND X PUSHX, H POP, FLDON2ND SHLD, RET,
 7|CODE ON3RD X PUSHX, H POP, FLDON3RD SHLD, RET,
 8|-->
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    192--------
 0|( BB op codes for playaction defined     BLDST )
 1|CODE BLDST   ( ball throw or hit )
 2|  B PUSH, VVEL BL LDA,
 3| ( max delt )   ' DSTLD CALL, ' DSTCALC  CALL,
 4|  LDFLAG CPI, <>, IF, A ANA, ( rdst ? ) 0<>, IF,
 5|  A XRA, GRNDR STA, INAIR STA,
 6|  THROW LDA, A ANA, 0<>, IF, ( throw or hit)
 7|  NBD H LXI, THWPA SHLD, CAUGHT STA, ELSE, H PUSH, D PUSH,
 8|  HITOF LDA, A ANA, 0<>, IF,
 9|-->
10|
11|
12|
13|
14|
15|
```

```
  +--------Block    193--------
 0|( BB op codes for playaction defined )
 1|  CMPT LDA, A ANA, CMPOF CNZ,
 2|  HOMER LDA, A ANA, 0< >, IF, STMUSIC STA, HOMERUN STA,
 3|  CHOMER A MVI, STRING STA, 10 A MVI, HMFLSHCNT STA, 5 A MVI,
 4|  HMRCNT STA,  A XRA, HOMER STA, ' KILLOF CALL, THEN,
 5|  FENCE LDA, A ANA, 0< >, IF, STMUSIC STA, FNCX LHLD,
 6|  VX BL LDED, D DAD, VDSTX BL SHLD, VY BL LHLD, FNCY LDED,
 7|  D DAD, VDSTY BL SHLD;
 8|  A XRA, FENCE STA, VACT VSTATUS X SETX, THEN, THEN,
 9|  D POP, H POP, THEN, THEN, THWDY SHLD, THWDX SDED,
10|  THEN, THWPA H LXI, VECTPC SHLD, B POP, RET,
11|-->
12|
13|
14|
15|
  +--------Block    194--------
 0|( BB op codes for playaction defined     FLDDST )
 1|CODE FLDDST ( fielders take position logic )
 2|  4 A MVI, ' DSTLD CALL, ' DSTCALC CALL, LDFLAG CPI, < >, IF,
 3|  A ANA, 0< >, IF, ( rdst ) VPT VSTATUS X BITX, 0< >, IF,
 4|  STPT B LXI, PITCHIT STA, CMPT LDA, A ANA, 0< >, IF, 4 A MVI,
 5|  ELSE, 1C A MVI, THEN, TIMEOUT STA, OFTBLGO STA, ELSE,
 6|  STND B LXI, THEN, VOF VSTATUS X BITX,
 7|  0< >, IF, OFTFCNT LDA, A DCR, OFTFCNT STA, A ANA, 0=, IF,
 8|  TAKEFIELD STA, 3 A MVI, OFTOTBL STA,
 9|  OFTFCNT STA, THEN, THEN, THEN,
10|  ( count all of outfielders in position )
11|  FLDPA SBCD, FLDDY SHLD, FLDDX SDED,
12|  THEN, FLDPA H LXI, VECTPC SHLD, RET,
13|-->
14|
15|
  +--------Block    195--------
 0|( BB op codes for playaction defined    OFFDST )
 1|FORWARD OFFPALD
 2|CODE OFFFDST ( all player off field control ) .ASSEMBLE
 3|  VOF VSTATUS X BITX, 0< >, IF, 20 A MVI, TFTIMER STA, THEN,
 4|  PLYR1UP LDA, VRUN VSTATUS X BITX, 0< >, IF, 1 XRI, THEN,
 5|  A ANA, 0=, IF, 3600 H LXI, ELSE, 1B00 H LXI,
 6|  THEN, 9300 D LXI, 5 A MVI, ( max vel )
 7|  ' DSTCALC CALL, LDFLAG CPI, < >, IF,
 8|  A ANA, 0< >, IF, NBD B LXI, VRUN VSTATUS X BITX, 0< >, IF,
 9|  FLDCLR LDA, A DCR, FLDCLR STA, 0=, IF, HOMERUN LDA, A ANA,
10|  0< >, IF, H PUSH, X PUSHX, H POP, WHOSUP LDA, L CMP, H POP,
11|  ( last guy? ) =, IF, 1 A MVI, CAUGHT STA, THROWAROUND STA,
12|  STMUSIC STA, CHEERME STA, STRINGOFFTIMER STA,
13|  A XRA, HOMERUN STA, THEN, THEN, THEN, THEN, THEN,
14|LABEL OFFPALD OFFPA SBCD, OFFDX SDED, OFFDY SHLD,
15|  THEN, OFFPA H LXI, VECTPC SHLD, RET, .END -->
```

```
  +--------Block    196--------
 0|( BB op codes for playaction defined     RUNDST )
 1| F= NVFORW F= RBACK F= ROFW F= NVAUTO F= SOFW F= RVGO F= SVAVF
 2| F= NFORW F= RFDST F= REND F= NOFPU F= CKATBS F= RVACT
 3|CODE RUNDST ( runner base running control ) .ASSEMBLE
 4|   VWBASE X B LDX, NORUN LDA, A ANA, 0<>, IF, ( caught )
 5|   X PUSHX, H POP, VMAGIC R1 LDA, L CMP, <>, IF, ( not r1 )
 6|   VLENGTH D LXI, A ANA, ( reset cy ) D DSBC, ( runner ahead )
 7|   H PUSH, Y POPX, VWBASE Y A LDX, A DCR, B CMP,
 8|   =, IF, VATBS VSTATUS Y BITX, 0<>, IF, ( man on next base )
 9|   2 A MVI, A VRSTAT X STX, ( go back ) THEN, THEN, THEN, THEN,
10|   VRSTAT X D LDX, VOFW X C LDX, HOMERUN LDA, A ANA, SVAVF JNZ,
11|   VAUTO D BIT, NVAUTO JZ, VFORW D BIT, NVFORW JZ,
12|   C A MOV, ( ofw ) A ANA, RFDST JNZ, SOFW JMP,
13|LABEL NVFORW C A MOV, ( ofw ) A ANA, CKATBS JZ,
14|LABEL ROFW A XRA, A VOFW X STX, B DCR, RVGO JMP,
15|-->
  +--------Block    197--------
 0|( BB op codes for playaction defined )
 1|LABEL NVAUTO ( runner under player control )
 2|   FORW LDA, A ANA, NFORW JZ, C A MOV, ( ofw ) A ANA,
 3|   SOFW JZ, VHW VSTATUS X BITX, ( half way ) RFDST JZ,
 4|LABEL SVAVF  3 A MVI, LVRSTAT CALL, ( vauto vforw for all )
 5|    RFDST JMP,
 6|LABEL SOFW 1 A MVI, A VOFW X STX,
 7|LABEL RVGO VGO VSTATUS X RESX, RFDST JMP,
 8|LABEL NFORW OFPU LDA, A ANA, NOFPU JZ, ( out field pick up )
 9|   2 A MVI, A VRSTAT X STX, ( set vauto res vforw ) NVFORW JMP,
10|LABEL NOFPU C A MOV, ( ofw ) A ANA, ROFW JNZ,
11|LABEL CKATBS B DCR, VATBS VSTATUS X BITX, REND JNZ,
12|LABEL RFDST B A MOV, B PUSH, ( which base )
13|   BSRTBL H LXI, A SLAR, ' INDEXW CALL, D PUSH, H INX,
14|   H INX, M E MOV, H INX, M D MOV, 4 A MVI, ( max vel ) H POP,
15|-->
  +--------Block    198--------
 0|( BB op codes for playaction defined )
 1|    ' DSTCALC CALL, LDFLAG CPI, <>, IF,  A ANA, 0<>, IF,
 2|   ( rchd dest ) VATBS VSTATUS X SETX, VHW VSTATUS X RESX,
 3|   VAUTO VRSTAT X RESX, B POP, B PUSH, ( base ) A XRA,
 4|   A VOFW X STX, VWBASE X A LDX, B CMP, =, IF, ( rchd nxt base)
 5|   A INR, 4 CPI, ( score ? ) =, IF,   SCRTIME STA, PSW POP,
 6|   FLDCLR LDA, A INR, FLDCLR STA, VACT VSTATUS X SETX,
 7|   STND B LXI, OFFPALD JMP, THEN,   A VWBASE X STX, THEN,
 8|   HOMERUN LDA, A ANA, RVACT JNZ, OFTBLGO LDA, A ANA, REND JZ,
 9|   LBASE LDA, A ANA, REND JNZ, NORUN LDA, A ANA, REND JNZ,
10|LABEL RVACT VACT VSTATUS X SETX,
11|LABEL REND RONBS B LXI, VXH X A LDX, 19 CPI, <, IF, ( on 3rd )
12|   LONBS B LXI, THEN, 0 H LXI, 0 D LXI,
13|   ELSE, VATBS VSTATUS X RESX, ( left base )
14|   THEN, RUNPA SBCD, RUNDY SHLD, RUNDX SDED, THEN, PSW POP,
15|   RUNPA H LXI, VECTPC SHLD, RET, .END -->
```

```
 +--------Block    199--------
 0|( BB playaction op code  OFMOTION , WAITTHRW , BLMOTION , OPTBL)
 1|CODE OFMOTION   ( set pc for ram pa table -tractor ball- )
 2|    OFPA ( 1+ ) LDA, 4 CPI, ( stnd ? )
 3|   =, IF, VACT VSTATUS X RESX, THEN,
 4|     OFPA H LXI, VECTPC SHLD, RET,
 5|CODE WAITTHRW   ( sets flag for 1stbaseman to wait throw )
 6|   ' THWANMSET CALL, BASESTATUS LDA, 5 CPI, RZ, A XRA,
 7|  THROWANM STA, 20 A MVI, WAITTHROW STA, RET,
 8|CODE BLMOTION   ( set pc for ram pa table -tractor ball- )
 9|        TBLPA H LXI, VECTPC SHLD, RET,
10|LABLE OPTBL
11|   ' ON1ST , ' ON2ND , ' ON3RD ,
12|   ' OFMOTION , ' BLMOTION , ' FLDDST , ' RUNDST , ' OFFFDST ,
13|   ' BLDST , ' THWANMSET , ' DEACTIVATE , ' WAITTHRW ,
14|-->
15|
 +--------Block    200--------
 0|(  BB playaction process   OPCODECHK , LOADANM )
 1|FORWARD OPRET
 2|CODE OPCODECHK   ( called by plyactupdate ) .ASSEMBLE
 3|      M A MOV, 28 CPI, <, A DCR, A DCR,    ( check if opcode )
 4|       IF, H INX,  VECTPC SHLD,     ( save pc  )
 5|        OPTBL H LXI, ' INDEXW CALL, ( which opcode )
 6|    OPRET H LXI, H PUSH, ( save ret add. ) XCHG, ( H-opcd )
 7|    PCHL, LABEL OPRET ( ex opcode return here )
 8|    VECTPC LHLD, THEN, RET, .END
 9|CODE LOADANM   ( loads magic & anm  HL-top of playact table )
10|                ( IX-proper vector add )
11|    M A MOV, A VMAGIC X STX, ( load magic )
12|    H INX, M A MOV, H PUSH, PATTERNS H LXI, ' INDEXW CALL,
13|      D VANMH X STX, E VANML X STX, ( load anm )
14|    H POP, ( pc ) H INX, RET,  ( returns updated pc in HL )
15|-->
 +--------Block    201--------
 0|(  BB playaction process    ANMSEQLOAD , PACTLOAD )
 1|CODE ANMSEQLOAD   ( finds proper anm seq by pers# )
 2|                   ( IX-proper vector add )
 3|     VANMH X H LDX, VANML X L LDX, VPERS# X A LDX,
 4|     ' INDEXW CALL,   ( anm seq look up )
 5|      D VANMSEQH X STX, E VANMSEQL X STX, RET,
 6|CODE PACTLOAD ( loads vect from plyact )
 7|  VPLAYACTPCH X  H LDX,  VPLAYACTPCL X L LDX,  ( plyact pc )
 8|     ' OPCODECHK CALL,  ( check for opcodes )
 9|  SAMEDLT LDA, A ANA, 0<>, IF, H INX, H INX, A XRA, SAMEDLT STA,
10|   H INX, H INX, H INX, H INX, H INX, ELSE, ' LOADANM CALL,
11| M A MOV, A VUPDATE# X STX, H INX, M A MOV, A VDYL X STX,
12| H INX, M A MOV, A VDYH X STX, H INX, M A MOV,
13| A VDXL X STX, H INX, M A MOV, A VDXH X STX, H INX,
14|  THEN, H  VPLAYACTPCH X STX, L VPLAYACTPCL X STX,
15|RET, -->
```

```
 +--------Block    202--------
 0|( BB playaction process      PATFETCH )
 1|CODE PATFETCH  ( loads actual pattern IX-vector )
 2|    503 H LXI, HITYET LDA, A ANA, 0=, IF, VPT VSTATUS X BITX,
 3|    0< >, IF, 1812 H LXI, THEN, THEN,
 4|    VANM# X A LDX,  ( animation # )
 5|  A INR, H CMP, >=, IF, A XRA, THEN,  ( start again)
 6|  A VANM# X STX, L CMP, <, IF, A XRA, ELSE, 1 A MVI, THEN,
 7|  ( which motion ) VANMSEQH X H LDX, VANMSEQL X L LDX,
 8|  ' INDEXW CALL,  ( look up proper pattern of sequence )
 9|  D VPATH X STX, E VPATL X STX, RET,
10|-->
11|
12|
13|
14|
15|
 +--------Block    203--------
 0|( BB playaction process      PLAYACT )
 1|CODE PLAYACT  ( called in intterupt does update prc )
 2|              ( assumes vector add in vectix )
 3|  B PUSH, X PUSHX,
 4|   VECTIX LIXD,  VUPDATE# X A LDX, A ANA, ( update # chk )
 5|     0=, IF, ' PACTLOAD CALL,    ( load new play action )
 6|     ELSE, FF CPI, =,
 7|       IF, ELSE,  A DCR,  A VUPDATE# X STX,  ( dcr update# )
 8|    THEN, THEN, ' ANMSEQLOAD  CALL, ' PATFETCH CALL, ( new pat.)
 9|  X POPX, B POP, RET,
10|BASE! ;S
11|
12|
13|
14|
15|
 +--------Block    211--------
 0|( BB inning player intialization   SETFDST )
 1|BASE@ HEX
 2|: SETFDST  ( set up fielder destination values )
 3|  1E00 VDSTX SS ! 4800 VDSTY SS !
 4|  3A00 VDSTX 1ST ! 5B00 VDSTY 1ST !
 5|  1400 VDSTX 3RD ! 5B00 VDSTY 3RD !
 6|  3200 VDSTX 2ND ! 4800 VDSTY 2ND !
 7|  2800 VDSTX PT ! 6100 VDSTY PT !
 8|  1400 VDSTX LF ! 2A00 VDSTY LF !
 9|  3C00 VDSTX RF ! 2A00 VDSTY RF !
10|  2800 VDSTX CF ! 1600 VDSTY CF ! ;
11|-->
12|
13|
14|
15|
```

```
 +--------Block    212--------
 0|( BB inning player intialization    LSETTFCOOR )
 1|: LSETTFCOOR ( set  take field coor & status -active , xpand )
 2| VMAGIC R1 4 0 DO DUP DUP DUP DUP DUP DUP 28 SB! VXH + 25 SB!
 3|    VYH + A8 SB! VXPAND + 88 SB! VSTATUS + 1 SB!
 4|    VPATL + NOBOD S! VLENGTH + ( next magic ) LOOP
 5|  ( pt ) 8 0 DO DUP DUP DUP DUP DUP 28 SB! ( flders )
 6|    VXH + PLYR1UP B@ IF 1B ELSE 36 THEN SB!
 7|    VYH + 93 SB! VXPAND + 88 SB! VPATL + NOBOD S!
 8|     VLENGTH + ( nxt magic ) LOOP DROP
 9|  28 VMAGIC BL ! NOBOD VPAT BL ! 88 VXPAND BL B! 2 VSTATUS BL B!
10|  10 VSTATUS PT  B! 20 VSTATUS RF B!
11|   20 VSTATUS LF B! 20 VSTATUS CF B! 3 OFTFCNT B!
12|  7 FLDOPCODE B! 6 TBLOPCODE B! 5 OFOPCODE B!
13|-->
14|
15|
 +--------Block    213--------
 0|( BB inning player intialization RETPA , SETTFPA )
 1|  9 OFFOPCODE B! 8 RUNOPCODE B! OPBLDST THWOPCODE B!
 2|  NBD RUNPA ! STND OFPA ! NBD TBLPA ! BAL THWPA ! ;
 3|: RETPA ( return to position pa )
 4|    VMAGIC PT 8 0 DO DUP DUP VPLAYACTPC + TFPA S! VSTATUS + DUP
 5|  B@ 80 OR F7 AND SB! VLENGTH + LOOP DROP ;
 6|: SETTFPA  ( take field pa )
 7|  TFPA2 VPLAYACTPC PT ! TFPA1 VPLAYACTPC 1ST !
 8|  TFPA1 VPLAYACTPC 2ND ! TFPA1 VPLAYACTPC 3RD !
 9|    TFPA1 VPLAYACTPC SS !
10|    RUNBPA VPLAYACTPC R1 !   RUNBPA VPLAYACTPC R2 !
11|    RUNBPA VPLAYACTPC R3 !   RUNBPA VPLAYACTPC R4 !
12|   TBLPA VPLAYACTPC BL ! ;
13|-->
14|
15|
 +--------Block    214--------
 0|( BB inning player intialization ZERORAM , SPEC0RAM , LSETTF )
 1|: SPEC0RAM 0 7C00 30 FILL ;
 2|: STFLD RETPA TAKEFIELD BONE  ;
 3|: SETTF LSETTFCOOR STFLD SETTFPA ;
 4|BASE! ;S
 5|
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    215--------
 0|( BB playaction loader )
 1|190 LOAD    ( playaction process )
 2|211 LOAD   ( inning player intialization )
 3|;S
 4|
 5|
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    216--------
 0|( BB vectors )
 1|   VLENGTH BARRAY R1
 2|   VLENGTH BARRAY R2
 3|   VLENGTH BARRAY R3
 4|   VLENGTH BARRAY R4
 5|   VLENGTH BARRAY PT
 6|   VLENGTH BARRAY 1ST
 7|   VLENGTH BARRAY 2ND
 8|   VLENGTH BARRAY SS
 9|   VLENGTH BARRAY 3RD
10|   VLENGTH BARRAY LF
11|   VLENGTH BARRAY CF
12|   VLENGTH BARRAY RF
13|   VLENGTH BARRAY BL
14|   ."     VPTR END - " VPTR @ H.
15|;S
 +--------Block    225--------
 0|( BB play action tables )
 1|HEX
 2|{ : REST, } NBD , B, 0 , 0 , { ; }  ( rest time give update# )
 3|{ : ACT, } , B, , , { ; }  ( given dx dy update#  magic )
 4|{ : ON1ST, } 2 B, { ; }
 5|{ : ON2ND, } 3 B, { ; } { : ON3RD, } 4 B, { ; }
 6|0A C= OPBLDST 6 C= OPBLMOT { : WTHW, } 0D B, { ; }
 7|{ : OFMOT, } 5 B, { ; }  { : BLMOT, } OPBLMOT B, { ; }
 8|{ : RUNDST, } 8 B, { ; } { : OFFDST, } 9 B, { ; }
 9|{ : BLDST, } OPBLDST B, { ; } { : FLDDST, } 7 B, { ; }
10|{ : THW, } 0B B, { ; } { : DEACT, } 0C B, { ; }
11|{ : STP, } 0 0 0 STND ACT, DEACT, { ; }
12|{ : RCB, } 0 0 0 RCB ACT, DEACT, { ; }
13|{ : LCB, } 0 0 0 LCB ACT, DEACT, { ; }
14|-->
15|
```

```
 +--------Block    226--------
 0|( BB play action tables -runners- )
 1|LABLE RUNBPA RUNDST,        LABLE OFFFPA OFFDST,
 2|LABLE TFPA  FLDDST,       LABLE OFTBLPA OFMOT,
 3|LABLE TFPA1 10 REST, FLDDST,
 4|LABLE TFPA2 20 REST, FLDDST,
 5|LABLE THWBLPA BLDST,
 6|LABLE PITCHBLPA BLMOT,
 7|LABLE HL1ST 200 60 0E RRUN ACT, ON1ST, LCB,
 8|LABLE HL2ND -300 0 0A LRUN ACT, ON2ND, LCB,
 9|LABLE HRSS 300 -20 0D RRUN ACT, ON2ND, RCB,
10|LABLE HR3RD -200 B0 0A  LRUN ACT, ON3RD, RCB,
11|LABLE HLPT -100 0 12 LRUN ACT, 0 0 0 STPT ACT, DEACT,
12|LABLE HRPT 100 0 12 RRUN ACT, 0 0 0 STPT ACT, DEACT,
13|-->
14|
15|
 +--------Block    227--------
 0|( BB play action tables fieldlogic -infielders- )
 1|LABLE NOBODYPA 0 0 0 NBD ACT, DEACT,
 2|LABLE PITCHPA 0 0 12 PTTHW ACT, 0 0 0 STPT ACT, DEACT,
 3|LABLE RTHWPA 0 0 5 RTRW ACT, STP,
 4|LABLE LTHWPA 0 0 5 LTRW ACT, STP,
 5|LABLE FBLR1ST 220 60 16 RRUN ACT, 100 40 2 RFLD ACT, THW, STP,
 6|LABLE FBLL3RD -1F0 80 12 LRUN ACT, -80 40 2 LFLD ACT,
 7|  THW, STP,
 8|LABLE NOACTPA STP,
 9|LABLE HLL3RD -100 0 12 LFLD ACT, 0 100 5 LRUN ACT,
10|   ON3RD, RCB,
11|LABLE F3RDR3RD -30 40 0A LFLD ACT, THW, STP,
12|LABLE F3RDL3RD 80 40 0A RFLD ACT, THW, STP,
13|LABLE F3RDLSS -100 40 12 LRUN ACT, STP,
14|LABLE HL3RD 80 -40 12 RFLD ACT, -200 C0 0E LRUN ACT, ON3RD, RCB,
15|  -->
 +--------Block    228--------
 0|( BB play action tables fieldlogic -infielders- )
 1|LABLE HLSS -80 0 10 LFLD ACT, STP,
 2|LABLE FSSRSS -80 80 0C LFLD ACT, THW, STP,
 3|LABLE FSSLSS 180 80 0C RFLD ACT, THW, STP,
 4|LABLE HMSS 100 -40 0C RRUN ACT, STP,
 5|LABLE F2NDR2ND -E0 A0 0C LFLD ACT, THW, STP,
 6|LABLE F2NDL2ND 80 0 0D RFLD ACT, THW, STP,
 7|LABLE HR2ND 80 0 12 RFLD ACT, STP,
 8|LABLE HR1ST -80 -40 12 LFLD ACT, 200 90 13 RRUN ACT, ON1ST, LCB,
 9| LABLE F1STR1ST -40 30 0A LFLD ACT, WTHW, STP,
10|LABLE F1STL1ST 140 -40 0B RFLD ACT, WTHW, STP,
11|LABLE HRL1ST 100 0 13 RFLD ACT, STP,
12|LABLE F1STPT 480 0 15 RRUN ACT, ON1ST, LCB,
13|LABLE HLOUTSSFA -20 -80 10 LRUN ACT, STP,
14|LABLE HROUT2NDFA 20 80 10 RRUN ACT, STP,
15|-->
```

```
  +--------Block    229--------
 0|( BB play action tables  single player pitches )
 1|HEX
 2|{ : BACT, } BAL ACT, { ; }
 3|{ : LK, } 0 0 0 BACT, DEACT, { ; }
 4|LABLE PFB 0 300 50 BACT,
 5|LABLE PSB 0 100 80 BACT,
 6|LABLE PSU 0 100 35 BACT, 0 300 50 BACT,
 7|LABLE PSD 0 300 12 BACT, 0 100 50 BACT,
 8|LABLE PCO 0 300 12 BACT, 100 200 9 BACT, 0 200 50 BACT,
 9|LABLE PCI 20 300 12 BACT, -60 200 9 BACT, 0 200 50 BACT,
10|LABLE PICI -10 200 1A BACT, -100 200 9 BACT, 0 200 50 BACT,
11|LABLE PICO -20 200 1A BACT, 60 200 9 BACT, 0 200 50 BACT,
12|DECIMAL ;S
13|
14|
15|
  +--------Block    230--------
 0|( BB destination delta calculation   DSTCALC )
 1|  BASE@ HEX
 2|FORWARD DSTSET FORWARD BTLP1 FORWARD RSRDST
 3|CODE DSTCALC ( calc deltas to destination )
 4| ( in HL-dstx DE-dsty A-max vel )
 5| ( out HL-dy DE-dx BC-magic & pattern # ) .ASSEMBLE
 6|   EXAF, VXH X B LDX, VXL X C LDX, B DSBC, PSW PUSH,
 7| ' COMPHL CC, L SLAR, H RALR, L SLAR,
 8| H RALR, XCHG, ( diffx-DE dsty-HL ) VYH X B LDX, VYL X C LDX,
 9| B DSBC, PSW PUSH,  ( diffy-HL )  ' COMPHL CC,
10| EXAF, PSW PUSH, EXAF, PSW POP, ( max vel for cmp value )
11| A INR, A INR, H CMP, RSRDST JC, D CMP, RSRDST JC,
12| A XRA, H CMP, DSTSET JNZ, D CMP, DSTSET JNZ, ( exact point )
13|  0 H LXI, 0 D LXI, PSW POP, PSW POP,
14|  0FF A MVI, RDST STA, ( set reached destination & 0 deltas )
15|  VACT VSTATUS X RESX, VGO VSTATUS X RESX, RET, -->
  +--------Block    231--------
 0|( BB destination delta calculation )
 1|LABEL RSRDST VGO VSTATUS X BITX, 0< >, IF, PSW POP, PSW POP,
 2|  ( only calc once )  LDFLAG A MVI, SAMEDLT STA,
 3|    ELSE, H A MOV, D CMP, PSW PUSH,
 4|  <, IF, XCHG, ELSE, =, IF, L A MOV, E CMP,  <, IF, XCHG,
 5|  PSW POP, STC, PSW PUSH, THEN, THEN,
 6|  THEN, ( HL > diff ) EXAF, ( max delt ) 0 B MVI,
 7|LABEL BTLP1 B INR, H SRLR, L RARR, ( /2 ) H CMP, BTLP1 JC,
 8| D SRLR, E RARR, ( /2 ) -6 DJNZ, ( reduce same as hl )
 9| PSW POP, ( xd>yd ) <, IF, XCHG, THEN,
10|LABEL DSTSET PSW POP, ( +- dy )
11| CY, IF, ' COMPHL CALL, THEN, PSW POP, PSW PUSH, ( +- dx )
12| CY, IF, ' COMPDE CALL, THEN,  PSW POP, ( dx sign )
13|  CY, IF, LRUN B LXI, ELSE, RRUN B LXI, THEN,
14|  A XRA, RDST STA, VGO VSTATUS X SETX, THEN, RET, .END
15|-->
```

```
 +--------Block    232--------
 0|( BB set destination registers from vector )
 1|CODE DSTLD
 2|  VDSTXH X H LDX, VDSTXL X L LDX, VDSTYH X D LDX,
 3|  VDSTYL X E LDX,
 4|RET,
 5|BASE! ;S
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    233--------
 0|( BB short subroutines COMPHL COMPDE TIMEDCR BONE BZERO WUPGO )
 1|BASE@ HEX
 2|CODE COMPHL ( compliments HL )
 3|  H A MOV, CMA, A H MOV, L A MOV, CMA, A L MOV, H INX, RET,
 4|CODE COMPDE  H PUSH, XCHG, ' COMPHL CALL, XCHG, H POP, RET,
 5|
 6|SUBR TIMEDCR ( HL-timer add if goes to zero return 1 else 0 )
 7|  M A MOV, A ANA, RZ, ( allready zero returns false change )
 8|  A DCR, A M MOV, A ANA, 0=, IF, A INR, ( true change )
 9|  ELSE, A XRA, THEN, RET,
10|
11|: BONE   ( cange byte memory cell to 1 ) 1 SB! ;
12|: BZERO ( change byte memory cell to 0 ) 0 SB! ;
13|
14|: WUPGO WHOSUP @ DUP VSTATUS + 81 SB! VRSTAT + 3 SB! ;
15|-->
 +--------Block    234--------
 0|( BB short subroutines DIVHLBY4 INDEXW EX  )
 1|CODE EX C9 B, NEXT
 2|CODE DIVHLBY4 ( divide HL by 4 )
 3|            7 H BIT, PSW PUSH, ( dir )
 4|      ' COMPHL CNZ,   ( abs dy )
 5|         H SRLR, L RARR, H SRLR, L RARR, ( /4 )
 6|       PSW POP, ' COMPHL CNZ, RET,
 7|CODE INDEXW ( call routine A- disp HL- table add )
 8|              ( for indexing into table only )
 9|  PSW PUSH, A SLAR,  ( 2* A )
10|   L ADD, A L MOV, 0 A MVI, H ADC, A H MOV,
11|   M E MOV, H INX, M D MOV, H DCX, PSW POP, RET,
12|-->
13|     '
14|
15|
```

```
 +--------Block    235--------
 0|( BB short subroutines AUTOR1 ect. , LVRSTAT , WALKOVER )
 1|SUBR AUTOR1 3 A MVI, VRSTAT R1 STA, 81 A MVI, VSTATUS R1 STA,
 2|  RET,  ( sets vauto , vforw , vact , vrun )
 3|SUBR AUTOR2 3 A MVI, VRSTAT R2 STA, 81 A MVI, VSTATUS R2 STA,
 4|  RET,
 5|SUBR AUTOR3 3 A MVI, VRSTAT R3 STA, 81 A MVI, VSTATUS R3 STA,
 6|  RET,
 7|SUBR AUTOR4 3 A MVI, VRSTAT R4 STA, 81 A MVI, VSTATUS R4 STA,
 8|  RET,
 9|SUBR LVRSTAT  ( load vstat for all runners  in- A value )
10|  VRSTAT R1 STA, VRSTAT R2 STA, VRSTAT R3 STA, VRSTAT R4 STA,
11|  RET,
12|SUBR WALKOVER   ( walked runner at base )
13|  1 A MVI, CAUGHT STA, THROWAROUND STA, RET,
14|-->
15|
 +--------Block    236--------
 0|( BB short subroutines  MULTHLBY4 , KILLOF , WAIT )
 1|CODE MULTHLBY4
 2|  7 H BIT, PSW PUSH, ' COMPHL CNZ, L SLAR, H RALR,
 3|  L SLAR, H RALR, PSW POP, ' COMPHL CNZ, RET,
 4|CODE KILLOF   ( stop outfield tractor ball control )
 5|  0 H LXI, OFDX SHLD, OFDY SHLD, STND H LXI, OFPA SHLD,
 6|  A XRA, OFTBLGO STA, RET,
 7|CODE WAIT    ( given A wait period stop every thing )
 8|  BEGIN, PSW PUSH, 0FF A MVI, BEGIN, A DCR, 0=, END,
 9|    PSW POP, A DCR, 0=, END, RET,
10|: TWAIT ( terse wait period )
11|  EI BEGIN FF 0 DO LOOP CREDITS B@ DUP IF 1 = IF CNSW12 B@ IF
12|  ELSE DROP 1 THEN ELSE DROP 1 THEN ELSE DROP THEN
13|  50 0 DO LOOP 1 - DUP 0= END
14|  DROP DI ;
15|-->
 +--------Block    237--------
 0|( BB short subroutines INFLDACT OUTFLDACT ALLFLDACT CMTALL )
 1|CODE SETACT  ( sets avtive bit in status )
 2|  ( in HL-status 1st guy A- # of guys )
 3|  VLENGTH D LXI,
 4|  BEGIN,  VACT M SET, D DAD, A DCR, 0=, END, RET,
 5|CODE INFLDACT  ( set infielders active )
 6|  VSTATUS PT H LXI, 5 A MVI, ' SETACT CALL, RET,
 7|CODE OUTFLDACT  ( sets outfielders active )
 8|  VSTATUS LF H LXI, 3 A MVI, ' SETACT CALL, RET,
 9|CODE ALLFLDACT  ( sets all fielders active )
10|    ' INFLDACT CALL, ' OUTFLDACT CALL, NEXT
11|: HOM 4200 7A00 828 48 CPOST 4400 7F00 8828 A" OME" SPOST ;
12|: VIS 700 7A00 828 58 CPOST 900 7F00 8828 A" ISITOR" SPOST ;
13|-->
14|
15|
```

```
 +--------Block    238--------
 0|( BB short subroutines      BLERASE , WUPWRT , CHGS , DWAIT )
 1|SUBR BLERASE  ( erases ball )
 2|   X PUSHX, VMAGIC BL X LXIX, ' VERASE CALL, VACT VSTATUS X RESX,
 3|   X POPX, ( stop ball ) NOBOD H LXI, VPAT BL SHLD, RET,
 4|CODE PTAUTOTM   ( pitch auto timer )
 5|   TIMEOUT H LXI, FLDCLR LDA, A ANA, TIMEDCR CZ, NEXT
 6|: WUPWRT   ( write whos up for flash )
 7|   PTAUTOTM PLYR1UP B@ IF HOM ELSE VIS THEN ;
 8|: CHGS  1C00 2000 828 A" CHANGE SIDES" SPOST ;
 9|: DWAIT EI 0 DO FF 0 DO LOOP LOOP DI ;
10|-->
11|
12|
13|
14|
15|
 +--------Block    239--------
 0|( BB short subroutines FLSHTON FLSHTOFF FLSHWUP DOCHGS )
 1|: FLSHTON FLSHWHO ! FLSHON BONE FLSHTIME BONE ;
 2|: FLSHME FLSHWHO @ EX ;
 3|SUBR FLSHWUP  ( handle whos up flasher )
 4|   X PUSHX, Y PUSHX,
 5|   FLSHCNT H LXI, FLSHON LDA, A ANA, M A MOV, 0<>, IF,
 6|   A ANA, 0=, IF, 1 M MVI, 28 A MVI, ELSE, 0 M MVI, 8 A MVI,
 7|   THEN, FLSHTIME STA, DOVERB FLSHME
 8|   ELSE, A ANA, 0<>, IF, 0 M MVI, DOVERB FLSHME THEN,
 9|   THEN, Y POPX, X POPX, RET,
10|CODE FLSHOFF B PUSH, A XRA, FLSHON STA, DI, FLSHWUP CALL,
11|   B POP, NEXT
12|: DOCHGS ' CHGS FLSHTON 28 DWAIT FLSHOFF GAMEOVER BZERO ;
13|-->
14|
15|
 +--------Block    240--------
 0|( BB short subroutines      CHKFLSHSTAY )
 1|: WUPTON ' WUPWRT FLSHTON ;
 2|: CHKFLSHSTAY   ( when isccp turn off check for wupwrt )
 3|   FLSHSTAY B@ IF FLSHSTAY BZERO FLSHCNT B@ FLSHOFF  IF
 4|   DOWUP B@ IF WUPWRT WUPTON ELSE WUPFLSH B@ IF WUPWRT WUPTON
 5|   THEN THEN ELSE ( not flshcnt ) DOWUP B@ IF WUPTON ELSE
 6|   WUPFLSH B@ IF WUPTON THEN THEN THEN WUPFLSH BZERO
 7|   DOWUP BZERO THEN ;
 8|BASE! ;S
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    241--------
 0|( VGS write routines    relabs , magic equates )
 1|( MAGIC REGISTER BITS )
 2|2 C= MRROT 3 C= MREXP 4 C= MROR 5 C= MRXOR
 3|6 C= MRFLOP 7 C= MRFLIP
 4|SUBR relabs ( relative X Y to magic address conversion )
 5|  ( in- BC=exp/mag DE=x HL=y )
 6|  ( out- BC=exp/mag+shift HL=scradr )
 7|  H A MOV, 0 H MVI, A L MOV,
 8|  H DAD, H DAD, H DAD,
 9|  H DAD, D PUSH, L E MOV, H D MOV, H DAD, H DAD, ( *64 )
10|  D DAD, ( *80 ) XCHG, H POP, ( x )
11|  L A MOV,  ( SAVE BIT CNT ) H L MOV, 0 H MVI, D DAD, ( x+y )
12|  RLC, RLC, HEX 3 ANI,
13|  MRFLOP C BIT, 0<>, IF, NEG, 0=, IF, H DCX, THEN, THEN,
14|  3 ANI, A E MOV, C A MOV, FC ANI, E ORA, A C MOV, RET,
15|-->
 +--------Block    242--------
 0|( VGS write routines    reloff )
 1|SUBR reloff ( compute relative offset of x , y of pattern )
 2|  ( in- BC=exp/mag DE=x HL=Y IY=relpatadr )
 3|  ( out- BC=exp/mag DE=x+off HL=y+off IY=patadr )
 4|  H PUSH, XCHG, 0 Y D LDX, 0 E MVI, ( X offset )
 5|  D SRAR, E RARR, D SRAR, E RARR,
 6|  ( MREXP C BIT, 0<>, IF, E SLAR, D RALR, ) ( *2 ) ( THEN, )
 7|  MRFLOP C BIT, 0<>, IF, D DAD, ELSE, A ORA, D DSBC, THEN,
 8|  XTHL, ( push X+off, HL<-Y ) 1 Y D LDX, 0 E MVI, ( Y offset )
 9|  MRFLIP C BIT, 0<>, IF, D DAD, ELSE, A ORA, D DSBC, THEN,
10|  D POP, Y INXX, Y INXX, ( offset pattern ) RET,
11|-->
12|
13|
14|
15|
 +--------Block    243--------
 0|( VGS write routines      write )
 1|SUBR write  ( software write with x y size )
 2|  ( IY= patadr BC= ex/magic+shf DE= Y/X size HL= screenadr )
 3|  ( does not do shifter flush patterns must flush themselves )
 4|   B PUSH, B A MOV, XPAND OUT, C A MOV, MAGIC OUT,
 5|   Y PUSHX, B POP, ( patadr ) MRFLOP A BIT, 0<>, IF,
 6|     BEGIN, ( y ) D PUSH, H PUSH,
 7|       BEGIN, ( x ) B LDAX, A M MOV, H DCX, A M MOV, H DCX,
 8|       B INX, E DCR, 0=, END,
 9|     H POP, 50 D LXI, D DAD, D POP, D DCR, 0=, END,
10|   ELSE, ( no flop )
11|     BEGIN, ( y ) D PUSH, H PUSH,
12|       BEGIN, ( x ) B LDAX, A M MOV, H INX, A M MOV, H INX,
13|       B INX, E DCR, 0=, END,
14|     H POP, 50 D LXI, D DAD, D POP, D DCR, 0=, END,
15|   THEN, B POP, ( magic ) RET,  -->
```

```
 +--------Block    244--------
 0|( VGS write routines   writep , WRITE )
 1|SUBR writep  ( software write with pattern size on pattern )
 2|  ( IY=patadr BC=ex/mag+shf HL=scradr --- )
 3|  0 Y E LDX, ( X size ) Y INXX,
 4|  0 Y D LDX, ( Y size ) Y INXX, write JMP,
 5|CODE WRITE   ( write with X Y sizes ; pattern with no header )
 6|  ( in- x , y , patadr , y/x size ex/mag )
 7|        ( WRTSYS set for pattern board res for software write )
 8|  ( out- pattern on screen )
 9|  Y PUSHX, H POP, EXX, B POP, ( ex/mag ) H POP, ( sizes )
10|  Y POPX, ( patadr ) D POP, ( Y ) XTHL, ( H<-X S<-sizes )
11|  XCHG, ( X<->Y ) relabs CALL, D POP, ( sizes )
12|  write CALL, EXX, H PUSH, Y POPX, NEXT
13|-->
14|
15|
 +--------Block    245--------
 0|( VGS write routines    WRITER )
 1|CODE WRITR ( write with a relative pattern )
 2|  ( in- x , y , relpatadr , ex/mag )
 3|        ( WRTSYS set for pattern board res for software write )
 4|  ( out- pattern on screen )
 5|  Y PUSHX, H POP, EXX, B POP, Y POPX, H POP, D POP,
 6|  reloff CALL, relabs CALL,
 7|  writep CALL, EXX, H PUSH, Y POPX, NEXT
 8|-->
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    246--------
 0|( VGS character routines     cpost  )
 1|  ( option bits , top 4 bits of exp )
 2|7 C= SMFONT   ( small font if set large font if res )
 3|6 C= ZEROSP   ( zero suppress if set )
 4|CODE cpost ( post an ascii-character on the screen ; see opt. )
 5|  ( in= x , y , opt+exp/mag , ascii-char )
 6|  ( out- newx , y , opt+ex/mag ; character on screen )
 7|  Y PUSHX, H POP, EXX, H POP, ( L<-char ) L A MOV,
 8|  B POP, ( ex/mag ) SMFONT B BIT, PSW PUSH, 0=, IF,
 9|  ( large font )
10|  41 CPI, >=, IF, 36 SUI, ELSE, 30 CPI, >=, IF, 2F SUI,
11|  ELSE, 20 SUI, THEN, THEN,
12|  ELSE, ( small font ) 20 SUI,
13|  THEN,
14|-->
15|
```

```
 +--------Block    247--------
 0|( VGS character routines      cpost con't. )
 1|  A L MOV, 0 H MVI,
 2|  L SLAR, H RALR, ( *2 ) H D MOV, L E MOV,
 3|  L SLAR, H RALR, ( *4 )
 4|  L SLAR, H RALR, ( *8 ) D DAD, ( *10 ) 0 characters D LXI,
 5|  PSW POP, ( font ) 0<>, IF, ( small font )
 6|  H SRLR, L RARR, ( /2=*5 ) 0 smallfont D LXI, THEN,
 7|  D DAD, H PUSH, Y POPX, ( patadr )
 8|  H POP, ( Y ) D POP, ( X )
 9|  D PUSH, H PUSH, B PUSH,
10|  relabs CALL, 0A01 D LXI, SMFONT B BIT, 0<>, IF, 5 D MVI,
11|  1 A MVI, ELSE, 2 A MVI, THEN, PSW PUSH, ( disp value )
12|  write CALL, PSW POP, B POP, H POP, XTHL, ( H<-X )
13|  A D MOV, 0 L MVI, D DAD, ( new x )
14|  XTHL, H PUSH, B PUSH, EXX, H PUSH, Y POPX, NEXT
15|-->
 +--------Block    248--------
 0|( VGS character routines      CPOST , SPOST , 3DROP )
 1|: 3DROP DROP DROP DROP ;
 2|: CPOST ( post an ascii-character on the screen ; see options )
 3|  ( in= x , y , opt+ex/mag , ascii-char )
 4|  ( out- character on screen )
 5|  cpost 3DROP ;
 6|
 7|: PPOST ( post an ascii-string on the screen leaving x y and
 8|           options for next string ; see options )
 9|  ( in= x , y , opt+ex/mag , string )
10|        ( i.e. 0 0 8828 28 A" STRING" PPOST )
11|  ( out- character on screen   x , y , opt+ex/mag )
12|  COUNT OVER + SWAP DO I B@ cpost LOOP ;
13|
14|-->
15|
 +--------Block    249--------
 0|( VGS character routines      NPOST )
 1|: SPOST ( post an ascii-string on the screen ; see options )
 2|  ( in= x , y , opt+ex/mag , string )
 3|        ( i.e. 0 0 8828 A" STRING" SPOST )
 4|  ( out- character on screen )
 5|  PPOST 3DROP ;
 6|
 7|: dpost 0F AND OVER 4000 AND 4000 XOR OVER OR IF 30 +
 8|  SWAP BFFF AND SWAP ELSE DROP 20 THEN cpost ;
 9|
10|: NPOST ( post a bcd number on the screen ; see options )
11|  ( in= x , y , opt+ex/mag , variable adr , # of bytes )
12|  ( out- character on screen )
13|  OVER + SWAP DO I B@ SWAN dpost I B@ dpost LOOP
14|  3DROP ;
15|-->
```

```
 +--------Block    250--------
 0|( VGS character routines     BCD+ , BCD+! )
 1|CODE BCD+  ( binary to decimal arithmatic )
 2|   ( in- integer 1 , integer 2 )
 3|   ( out- bcd sum of integer 1 and 2 )
 4|   H POP, D POP, E A MOV, L ADD, DAA, A L MOV,
 5|   D A MOV, H ADC, DAA, A H MOV, H PUSH, NEXT
 6|: BCD+! ( add a bcd number to a variable )
 7|   ( in- inc-amount , msb-addr , #bytes )
 8|   ( out- value in variable incermented in bcd )
 9|   1- OVER + DO I B@ BCD+ DUP I B! SWAB [ HEX ] FF AND
10|   -1 +LOOP DROP ;
11|-->
12|
13|
14|
15|
 +--------Block    251--------
 0|( BB vector write     VWRITE )
 1|CODE VWRITE ( does reloff relabs patbrd from vector IX )
 2|   VXPAND X B LDX, VMAGIC X C LDX, VXH X D LDX, VXL X E LDX,
 3|   VPATH X H LDX, VPATL X L LDX, H PUSH, Y POPX,
 4|   VYH X H LDX, VYL X L LDX,
 5|  reloff CALL,  ( calculates relative offset )
 6|  relabs CALL,  ( calculates magic add. )
 7|   H VSCRADRH X STX, L VSCRADRL X STX,  ( set scradr for erase )
 8|  writep CALL,  ( write it )
 9|   C VMAGIC X STX, ( save shift for erase )
10|   RET,
11|-->
12|
13|
14|
15|
 +--------Block    252--------
 0|( BB interupt vector erase     VERASE )
 1|CODE VERASE  ( does pattern board erase from vector IX )
 2|   VXPAND X B LDX, VMAGIC X C LDX, VPATH X H LDX, VPATL X L LDX,
 3|   H INX, H INX, ( abs ) H PUSH, Y POPX,
 4|   VSCRADRH X H LDX, VSCRADRL X L LDX,
 5|   writep CALL, RET,
 6|DECIMAL ;S
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    253--------
 0|( BB interrupt   CHKGROUNDER )
 1|BASE@ HEX
 2|LABLE GVTBL ( grounder varb table )
 3|   0 , 40 , 40 , 40 , 0 , 0 , 0 , 0 , FFC0 , FFC0 , FFC0 ,
 4|   0 , 0 ,
 5|CODE CHKGROUNDER  ( if grounder adjust ball direction )
 6|   GRNDR LDA, A ANA, RZ,
 7|   GRNDRVALUE LDA, A INR, 0D CPI, >=, IF, A XRA, THEN,
 8|   GRNDRVALUE STA, H PUSH, GVTBL H LXI, ' INDEXW CALL,
 9|   VXH X A LDX, 0A0 CPI, ' COMPDE CC, H POP, D DAD, RET,
10|-->
11|
12|
13|
14|
15|
 +--------Block    254--------
 0|( BB interrupt     TIMER )
 1|CODE TIMER ( interrupt timer routine )
 2|    THROWTIMER H LXI, TIMEDCR CALL, A ANA, 0<>, IF,
 3|       THROWTIME STA, THEN,
 4|    STRINGOFFTIMER H LXI, TIMEDCR CALL,
 5|      A ANA, 0<>, IF, STRINGERASE STA, OLDSTRING LDA,
 6|                  STRING STA, THEN,
 7|    TFTIMER H LXI, TIMEDCR CALL, A ANA, 0<>, IF,
 8|      TFTIME STA, THEN,
 9|    WAITTHROW H LXI, TIMEDCR CALL, A ANA, 0<>, IF,
10|      THROWANM STA, THEN,
11|    CMOFTIMER H LXI, TIMEDCR CALL, A ANA, CMPOF CNZ, EI,
12|    FLSHTIME H LXI, TIMEDCR CALL, A ANA, FLSHWUP CNZ,
13|    TOTIMER H LXI, TIMEDCR CALL, A ANA, TAKEOFF CNZ,
14|    WALK H LXI, TIMEDCR CALL, A ANA, WALKOVER CNZ,
15|-->
 +--------Block    255--------
 0|( BB interrupt     PERSPECTIVE )
 1|   TOOF H LXI, TIMEDCR CALL, A ANA, TOOFCK CNZ, RET,
 2|
 3|CODE PERSPECTIVE ( pattern size determined by y )
 4|  0 E MVI, VYH X H LDX, 54 A MVI, H CMP, <, IF, 84 A  MVI, H CMP,
 5|   <, IF, 0 D MVI, ELSE, 1 D MVI, THEN,  ELSE, 3A A MVI, H CMP,
 6|   <, IF, 2 D MVI, ELSE, 1F A MVI, H CMP, <, IF, 3 D MVI,
 7|   ELSE, 4 D MVI, THEN, THEN, THEN,
 8|    VPERS# X A LDX, D CMP, ( cmp pers new & old )
 9|    D VPERS# X STX, <>, IF, VHW VSTATUS X SETX, THEN,
10|   ( for runners set commited if change )
11|   RET,
12|-->
13|
14|
15|
```

```
 +--------Block    257--------
 0|( BB interrupt    VECTOR )
 1|CODE VECTOR   ( vectors and limit checks )
 2|   C001 B LXI, ( y limits ) TAKEFIELD LDA, A ANA, 0=, IF,
 3|   VOF VSTATUS X BITX, 0<>, IF, 4009 B LXI, THEN, THEN,
 4|   ( outfielder limits ) VXH X D LDX, VXL X
 5|   E LDX, VDXH X H LDX, VDXL X L LDX,
 6|  ' DIVHLBY4 CALL, ( comm. resolution ) D DAD,
 7|  VBL VSTATUS X BITX, ' CHKGROUNDER CNZ, 1 A MVI, H CMP,
 8|  >=, IF, A H MOV, ELSE, 4E A MVI, H CMP, <, IF, A H MOV,
 9|  THEN, THEN, H VXH X STX, L VXL X STX, ( limit chk )
10|   VYH X H LDX, VYL X L LDX, VDYH X D LDX, VDYL X E LDX,
11|   D DAD, C A MOV, H CMP, >=, IF, ( y low ) A H MOV, ELSE,
12|   B A MOV,
13|  H CMP, <, IF, A H MOV, THEN, THEN, H VYH X STX, L VYL X STX,
14|   RET,
15|-->
 +--------Block    258--------
 0|( BB interrupt      BATHITCHK )
 1|LABLE BWTBL ( bat window table y span Y, xspan X, )
 2|   0EAF , 698 , 0DAF , 0D98 , 09AF , 0F98 , 08AC , 1098 ,
 3|   9A7 , 0F98 , 0DA3 , 0D98 , 0EA2 , 0696 ,
 4|CODE BATHITCHK ( check window for hit )
 5|   M A MOV, ( swing) A SLAR, BWTBL H LXI, ' INDEXW CALL,
 6|   VYH BL LDA, E SUB, D CMP, RNC, H INX, H INX, H PUSH,
 7|   VX BL LHLD, ' MULTHLBY4 CALL, H A MOV, H POP, M SUB, H INX,
 8|   M CMP, RNC, HITYET LDA, A ANA, RNZ, A INR,
 9|   HITTIME STA, HITGOING STA, BLERASE CALL, RET,
10|-->
11|
12|
13|
14|
15|
 +--------Block    259--------
 0|( BB interrupt    BATWRITE , BATSWING )
 1|LABLE BSWINGTABLE BATD90 , BATD45 , BATD30 ,
 2|       BATMID , BATU30 , BATU45 , BATU90 ,
 3|CODE BATWRITE   ( xor bat from swingtype )
 4|    H PUSH, M A MOV, BSWINGTABLE H LXI, ' INDEXW CALL,
 5|    D PUSH, ( pattern add. ) Y POPX, 2600 D LXI, AF00 H LXI,
 6|    828 B LXI, ( exp/mag )
 7|   reloff CALL, relabs CALL, writep CALL,
 8|    H POP, RET,
 9|CODE BATSWING   ( swings bat if appropriate )
10|    SWINGTYPE  H LXI, SWING LDA, A ANA,
11|     0<>, IF, STRIKE STA, M A MOV, 6 CPI,    ( end of swing ? )
12|         <, IF, ' BATWRITE CALL, M INR, ' BATWRITE CALL,
13|        HITGOING LDA, A ANA, ' BATHITCHK CZ, ( check for hit )
14|         ELSE, A XRA, SWING STA, 18 A MVI, SWINGTIME STA,
15|         THEN, -->
```

```
 +--------Block    260--------
 0|( BB interrupt )
 1|          (  swing completion reset bat )
 2|     ELSE, ( swing not set ) M A MOV, 6 CPI,
 3|          =, IF, A XRA, HITGOING STA, SWINGTIME LDA, A ANA,
 4|            0=, IF, ' BATWRITE CALL, 0 M MVI, ' BATWRITE CALL,
 5|            ELSE, A DCR, SWINGTIME STA,
 6|     THEN, THEN, THEN, RET,
 7|-->
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    261--------
 0|( BB interrupt        OFBLCHK )
 1|SUBR OFBLCHK    ( chk for catch or close when cmsw )
 2|  ( in- D max distance to check for )
 3|  VLENGTH B LXI, VX BL LHLD,
 4|  VMAGIC LF Y LXIX, VYH BL LDA, A INR, A E MOV, 3 A MVI,
 5|  BEGIN, H PUSH, EXAF, VYH Y A LDX, E SUB, CY, IF, NEG, THEN,
 6|  D CMP, ( max dist ) <, IF,
 7|  D PUSH, VXH Y D LDX, VXL Y E LDX, A ANA, ( res cy )
 8|  D DSBC, ' COMPHL CC, D POP, 10 A MVI, H CMP, >=, IF,
 9|  ' MULTHLBY4 CALL, H A MOV, D CMP, <, IF, 1 A MVI, H POP, RET,
10|  THEN, THEN, THEN,
11|  B DADY, EXAF, A DCR, H POP, 0=, END, RET,
12|-->
13|
14|
15|
 +--------Block    262--------
 0|( BB interrupt        BLPOSCHK )
 1|CODE BLPOSCHK
 2|  HITYET LDA, A ANA, 0=, IF, ( pitch check )
 3|  VYH X A LDX, BC CPI, >=, IF, HITTIME LDA, A ANA, RNZ,
 4|  HITYET LDA, A ANA, RNZ, VX PT LHLD, VDSTX BL SHLD,
 5|  VY PT LHLD, VDSTY BL SHLD, THWBLPA H LXI, VPLAYACTPC BL SHLD,
 6|  A XRA, VUPDATE# BL STA, 4 A MVI, VVEL BL STA,
 7|  81 A MVI, VSTATUS BL STA, BAL H LXI, THWPA SHLD,
 8|  STRIKE LDA, A ANA, 0<>, IF, CSTRIKE A MVI,
 9|  ELSE, CBALL A MVI, THEN, STRING STA,
10|  A XRA, PITCHTIME STA, STRIKE STA, SWINGGO STA,
11|  1 A MVI, HITYET STA, THROW STA, THROWAROUND STA,
12|  ELSE, ( check for cross the plate ) 0A9 SBI, 5 CPI, RNC,
13|  VXH X A LDX, 26 SBI, 2 CPI, RNC, 1 A MVI, STRIKE STA,
14|  THEN,
15|-->
```

```
 +--------Block    263--------
 0|( BB interrupt )
 1|  ELSE, ( intercept check for catch in outfield )
 2|  VYH X D LDX, 3C A MVI, D CMP, RC, NOCATCH LDA, A ANA, RNZ,
 3|  5 D MVI, OFBLCHK CALL, A ANA, RZ, INAIR LDA, A ANA, 0<>, IF,
 4|  2 A MVI, LVRSTAT CALL,
 5|  OFCATCH STA, CAUGHT STA, WHOSUP LHLD,
 6|  PLAYON SHLD, A XRA, ELSE, A INR, OFPU STA, THEN, THROWANM STA,
 7|  A INR, NOCATCH STA, Y PUSHX, H POP, WHOTHROWS SHLD,
 8|  BLERASE CALL, ' KILLOF CALL,
 9|  A XRA, OFTBLGO STA, TOOF STA, ( ofl t.o. timer ) THEN, RET,
10|-->
11|
12|
13|
14|
15|
 +--------Block    264--------
 0|( BB interupt main    WINTBL , INTERRUPT )
 1|LABLE WINTBL  ( which interrupt table )
 2|  VMAGIC R3 , VMAGIC 1ST , VMAGIC 3RD , VMAGIC RF ,
 3|
 4|FORWARD  INTTOP
 5|ICODE INTERRUPT       .ASSEMBLE
 6|  INTFLAG LDA, A INR, INTFLAG STA, 2 CPI, <, IF,
 7|LABEL INTTOP B PUSH, Y PUSHX, X PUSHX,
 8|-->
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    265--------
 0|( BB interupt main )
 1|  WHICHINT LDA, A INR, 4 CPI, =, IF, A XRA, THEN,
 2|  WHICHINT STA, WINTBL H LXI, ' INDEXW CALL, D PUSH, X POPX,
 3|  ( 3 guys every 4 interrupts )
 4|-->
 5|
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    266--------
 0|( main vectoring, does 3 vectors given IX-starting vect )
 1|  3 A MVI, VLENGTH D LXI,
 2|BEGIN  PSW PUSH, D PUSH, VACT VSTATUS X BITX,
 3|  0< >, IF, A XRA, BLWAIT STA,
 4|         ' VERASE CALL,     ( erase )
 5|        VECTIX SIXD, ' PLAYACT CALL,   ( player action control )
 6|         ' VECTOR CALL,      ( vector limits & perspective chk )
 7|         ' PERSPECTIVE CALL, ( change of size )
 8|         ' VWRITE CALL,    ( write )
 9|  THEN,    D POP, A XRA, ( res carry ) X PUSHX, H POP,
10|  D DSBC, ( nxt vector ) H PUSH, X POPX, PSW POP, A DCR,
11|0=, END,
12|-->
13|
14|
15|
 +--------Block    267--------
 0|( BB interupt ball and bat process )
 1|    ' BATSWING CALL,   ( if swing every int. )
 2|  VMAGIC BL X LXIX, VACT VSTATUS X BITX,
 3|    0< >, IF, BLWAIT LDA, A ANA, 0< >, IF,
 4|   ' WAIT CALL, ELSE,  2 A MVI, BLWAIT STA, THEN,
 5|       ' VERASE CALL,    ( erase )
 6|       VECTIX SIXD, ' PLAYACT CALL,  ( ball action logic )
 7|       ' VECTOR CALL,  ( vector & limit chk )
 8|       ' VWRITE CALL,   ( write )
 9|   THEN,
10|       ' BLPOSCHK CALL, ( check pitch or outfiled catch )
11|-->
12|
13|
14|
15|
 +--------Block    268--------
 0|( BB interrupt )
 1|   X POPX, Y POPX, Y PUSHX, X PUSHX, ' MUSCPU CALL,
 2|  ' TIMER CALL,    ( system timers decrimented & flaged )
 3|  TAKEFIELD LDA, A ANA, ' TBALLPRC CZ,  ( tractor ball )
 4|   X POPX, Y POPX, CHKCOIN1 CALL, CHKCOIN2 CALL,
 5|  STRING LDA, A ANA, 0< >, IF, DOVERB STRINGPRC THEN,
 6|  SCORESHOW LDA, A ANA, 0< >, IF, DOVERB SCOREME THEN,
 7|  B POP,
 8|  INTFLAG LDA, A DCR, INTFLAG STA, INTTOP JNZ, THEN,
 9|  INEXT .END
10|-->
11|
12|
13|
14|
15|
```

```
 +--------Block    269--------
 0|( BB interrupt call      INTERRUPTS0 , HMRINT , SI0 , SI1 )
 1|ICODE HMRINT   ( homer shill interrupt )
 2|   HMFLSHCNT H LXI, M A MOV, A ANA, 0=, IF,
 3|   HMRFLSH LDA, 1 XRI, HMRFLSH STA, 20 M MVI,
 4|   A ANA, 0<>, IF, 7 A MVI, 0 D MVI, ELSE, 7 D MVI, THEN,
 5|   0 OUT, 4 OUT, D A MOV, 1 OUT, 2 OUT, 3 OUT, THEN, M DCR,
 6|   INEXT
 7|<INTERRUPTS INTERRUPTS0
 8|73 I' INTERRUPT INTERRUPTS>
 9|<INTERRUPTS INTERRUPTS1
10|73 I' HMRINT C0 I' INTERRUPT INTERRUPTS>
11|: SI0 INTERRUPTS0 ISTART ; : SI1 INTERRUPTS1 ISTART ;
12|BASE! ;S
13|;S
14|
15|
 +--------Block    270--------
 0|( BB field table  dugout pattern    cnline )
 1|HEX
 2|LABLE DUGOUT 0 0 0 3 0 0 80 0F 0 0 E0 1F 0 0 F0 7F 0 0 FC FE
 3|   0 0 7E 78 0 80 1F 10 0 C0  0F 0 0 F0 3 0 0 F8 1 0
 4|   0 7E 0 0 0 3F 0 0 80 1F 0 0 80 1F 0 0 0 7F 0 0 0 FE 0 0
 5|   0 78 0 0 0 10 0 0 12 4 0 0 24STF 24STF 24STF B, B, B, B,
 6|BTABLE FIELD
 7|   9D B, AC B, 0 B, 3E B, 1 B, 2D B, 4 B, 24 B, 8 B, 1D B,
 8|   0D B, 16 B, 14  B, 0D B, 1F B, 06 B, 28 B, 03 B, A0 B, 03 B,
 9|
10|CODE cnline   ( calls vector line write )
11|   B PUSH, Y PUSHX, X PUSHX, VECTOR CALL,
12|   X POPX, Y POPX, B POP, NEXT
13|-->
14|
15|
 +--------Block    271--------
 0|( BB line vector routines   CNLINE , DLINE )
 1|CODE strtline   ( given X Y sets up start of line )
 2|   H POP, ( y ) D POP, ( x ) E SLAR, D RALR, E SLAR, D RALR,
 3|   E SLAR, D RALR, E SLAR, D RALR, E SLAR, D RALR,
 4|   E SLAR, D RALR, ( adjust x for crelabs ) B PUSH,
 5|   0 C MVI,
 6|   relabs CALL, 3 ANI, BIT-POS STA, 4000 D LXI, D DAD,
 7|   SADR SHLD, B POP, NEXT
 8|
 9|: CNLINE    ( continue line to this point , given X Y )
10|   YIN ! XIN ! cnline ;
11|
12|: DLINE   ( draw a line , given X Y start coor X Y ending coor )
13|   YIN ! XIN ! 2DUP Y-AXIS ! X-AXIS !
14|   SWAB ( set y for crelabs ) strtline cnline ;
15|-->
```

```
 +--------Block    272--------
 0|( BB line vector routines   RECTAN , OUTLINE )
 1|: RECLINE 4 PICK 4 PICK 2DUP SWAP 6 PICK + SWAP DLINE ;
 2|: RECTAN  ( in X, Y, X length, Y length, pxtype set )
 3|   BEGIN RECLINE ROT 1+ ROT ROT 1- DUP 0= END DROP DROP DROP
 4|   DROP ;
 5|
 6|: OUTLINE  ( creates rectanguler outline )
 7|   ( in- X , Y , X length , Y length ; pxtype set )
 8|   4 NDUP DROP 3 PICK + OVER 4 NDUP DLINE 5 PICK + CNLINE
 9|   3 PICK + CNLINE DROP DROP CNLINE ;
10|
11|: SB0 3700 B000 308 BALLS B@ CPOST 3F00 B000 308 STRIKES B@
12|   CPOST 4700 B000 308 OUTS B@ CPOST ;
13|-->
14|
15|
 +--------Block    273--------
 0|( BB field write main      FIELDWRT , SUP )
 1|BV= FINDEX
 2|: GETF FINDEX B@ FIELD B@ FINDEX 1+! ;
 3|: REVX 13F GETF  - ;
 4|: FIELDWRT   ( write field outline )
 5|   GETF GETF GETF GETF DLINE
 6|   8 0 DO GETF GETF CNLINE LOOP  ( left side )
 7|   FINDEX BZERO REVX GETF REVX GETF DLINE
 8|   8 0 DO REVX GETF CNLINE LOOP ( right side ) ;
 9|
10|: SUP 8C00 8828 A" UP" SPOST ;
11|: SCRS 1F 8A 18 0E RECTAN 107 8A 18 0E RECTAN
12|  B00 8C00 308 30 CPOST 4500 8C00 308 30 CPOST ;
13|: FLFILL EI 0 4000 3C00 FILL DI ;
14|: ZERORAM EI 0 7C30 270 FILL 3 OFTFCNT B! DI ;
15|-->
 +--------Block    274--------
 0|( BB field write main      FL )
 1|HEX
 2|: FL   ( set up all of background )
 3|    ZERORAM FLFILL
 4|EI      PX1 FIELDWRT
 5|   10C0 6600 BASEPAT 4428 WRITR 3FC0 6600 BASEPAT 4428 WRITR
 6|   2880 4C00 BASE2PAT 4428 WRITR
 7|   2800 AC00 HOMEPLATE 8808 WRITR  2600 AF00 BATD90 8828 WRITR
 8|   0C AA 6E 15 OUTLINE
 9|   400 B000 828 A" INN" SPOST
10|   B00 AF00 8848 A" 1 2 3 4 5 6 7 8 9" SPOST
11|   C4 AA 6C 15 OUTLINE 3300 B000 828 42 CPOST
12|   D7 AE 10 0E RECTAN 3B00 B000 828 53 CPOST
13|   F7 AE 10 0E RECTAN 4300 B000 828 4F CPOST
14|EIDI    117  AE 10 0E RECTAN
15|-->
```

```
 +--------Block    275--------
 0|( BB field write )
 1|  3300 8D00 DUGOUT 828 WRITR 1D00 8D00 DUGOUT 868 WRITR
 2| HOM VIS SCRS EIDI  30 DUP DUP BALLS B! STRIKES B! OUTS B! SBO
 3|  F00 SUP
 4|  300 A300 8828 A" CREDITS" SPOST EIDI  ;
 5|DECIMAL ;S
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    280--------
 0|( PIXEL TABLES ) BASE@ HEX
 1|{ : FOWR } FORWARD { ; }
 2|V= PXTYPE V= XIN V= YIN V= Y-AXIS V= X-AXIS
 3|V= BIT-POS V= SADR  V=  V-PXTYPE V= V-MODE ( 0= write 1= xor )
 4|BTABLE PXT0 0 , 0 ,
 5|BTABLE PXT1 40 B, 10 B, 4 B, 1 B,
 6|: PX1 0 PXT1 PXTYPE ! ;
 7|: PX0 0 PXT0 PXTYPE ! ;
 8|BTABLE MASK 3F B, 0CF B, 0F3 B, 0FC B,
 9|BASE!  -->
10|
11|
12|
13|
14|
15|
 +--------Block    281--------
 0|( VECTOR GENERATOR )
 1|FOWR VECTOR FOWR XYLOOP FOWR X-RET FOWR Y-RET
 2|FOWR INC-X FOWR DEC-X FOWR INC-Y FOWR DEC-Y
 3|FOWR YSTEP  FOWR XYSHIFT  FOWR SINCR
 4|FOWR XYTEST  FOWR PWRT
 5|ASM .ASSEMBLE ( START 2 PASS ASSEMBLER HERE )
 6|LABEL INC-X E RRCR, E RRCR, ( SHIFT BIT VAL )
 7|   D RRCR, D RRCR, ( SHIFT MASK ) CY~,
 8|   EXX, IF, SADR LHLD, H INX, SADR SHLD, THEN,
 9|   X-AXIS LHLD, H INX, X-AXIS SHLD, EXX, X-RET JMP,
10|LABEL DEC-X E RLCR, E RLCR, ( SHIFT BIT VAL )
11|   D RLCR, D RLCR, ( SHIFT MASK ) CY~,
12|   EXX, IF, SADR LHLD, H DCX, SADR SHLD, THEN,
13|   X-AXIS LHLD, H DCX, X-AXIS SHLD, EXX, X-RET JMP, -->
14|
15|
```

```
 +--------Block    282--------
 0|( VECTOR GENERATOR ) BASE@ DECIMAL
 1|LABEL INC-Y Y-AXIS H LXI, M INR,
 2| D PUSH, 80 D LXI, SADR LHLD, D DAD, SADR SHLD, D POP,
 3| RET,
 4|LABEL DEC-Y Y-AXIS H LXI, M DCR,
 5| D PUSH, -80 D LXI, SADR LHLD, D DAD, SADR SHLD, D POP,
 6| RET,
 7|LABEL YSTEP PCIY,
 8|BASE!  -->
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    283--------
 0|( VECTOR GENERATOR )
 1|( INPUT IS XIN, YIN AS PLACE TO MOVE TO. NOTE SADR & BIT-POS
 2|  MUST ALREADY BE SET, AS WELL AS X-AXIS AND Y-AXIS )
 3|LABEL VECTOR X-AXIS LDED, XIN LHLD, A XRA, D DSBC,
 4|  CY~, IF, INC-X X LXIX,
 5|     ELSE, DEC-X X LXIX, XCHG, A XRA, 0 H LXI, D DSBC, THEN,
 6|  Y-AXIS LDED, YIN LDA, E SUB, ( DIF OF Y )
 7|  CY~, IF, INC-Y Y LXIX,
 8|     ELSE, DEC-Y Y LXIX, CMA, A INR, ( MAKE A + ) THEN,
 9|EXX, A L MOV, 0 H MVI, EXX,  L E MOV, H D MOV, ( copy X to DE )
10|  L ORA, A L MOV, ( OR X & Y ) H ORA, RZ, ( ret if 0 dif )
11|( normalize - shift up until carry occurs )
12|LABEL XYSHIFT H DAD, ( shift up combined X & Y ) SINCR JC,
13|   XCHG, H DAD, ( shift up X ) XCHG,
14|   EXX, H DAD, EXX, ( shift up Y )
15|   XYSHIFT JMP, LABEL SINCR -->
 +--------Block    284--------
 0|( VECTOR GEN )
 1|  D PUSH, EXX, H PUSH, EXX, B POP, C E MOV, B D MOV, ( Y parms )
 2| EXX,
 3| ( PUT MASK IN D, PIXEL VALUE IN E -NOTE THESE GO IN ALT REGS )
 4|  BIT-POS LBCD, PXTYPE LHLD, B DAD, M E MOV,
 5|  0 MASK H LXI, B DAD, M D MOV,
 6|   H POP, ( X dif ) H B MOV, L C MOV,  EXX,
 7|LABEL XYLOOP EXX, B DAD, CY,
 8|   IF, PCIX, LABEL X-RET
 9|       EXX, XCHG, B DAD, XCHG, CY,
10|          IF, YSTEP CALL, THEN, PWRT  JMP,
11|   ELSE, EXX, XCHG, B DAD, XCHG, CY,
12|          IF, YSTEP CALL, PWRT  JMP, THEN,
13|   THEN,
14| XYLOOP JMP,
15|-->
```

```
  +--------Block    285--------
 0|( VECTOR GEN - WRITE POINT AND TEST )
 1|LABEL PWRT
 2|( WRITE THE POINT )
 3|    EXX, D PUSH, EXX, H POP, XCHG, H PUSH, ( save Y sum )
 4|    SADR LHLD, V-MODE LDA, A ORA, 0<>, IF, E M MOV, ELSE,
 5|    M A MOV, D ANA, ( MASK OFF )
 6|    E ORA, ( OR IN BIT VAL ) A M MOV,       THEN,
 7|    D POP, ( return Y sum )
 8|( END CHECKS )
 9|    YIN LDA, A H MOV, Y-AXIS LDA, H CMP, XYLOOP JNZ,
10|    D PUSH, A XRA, XIN LDED, X-AXIS LHLD, D DSBC, D POP,
11|      XYLOOP JNZ,
12|  X-AXIS LDA, 3 ANI, BIT-POS STA,
13| RET,
14|.END  ( END 2 PASS ASSEMBLER )
15|;S
  +--------Block    286--------
 0|( COIN READING ROUTINE ) HEX
 1|( A= mask of bits to look for )
 2|( D= # of times to try to accept value )
 3|( E= # of consecutive values to find )
 4|( H= port from which to read )
 5|( L= value to look for , ** PUSH DE, THEN HL, THEN A ** )
 6|
 7|FORWARD READPORT SUBR debounce .ASSEMBLE
 8|B PUSH, H C MOV, E B MOV, A H MOV, LABEL READPORT
 9|A INP, L XRA, H ANA, 0=, IF, E DCR, 0=, IF, ( good ) A INR,
10|B POP, RET, THEN, READPORT JMP,
11|THEN, D DCR, 0=, IF, A XRA, B POP, RET, THEN,
12|B E MOV, ( try again ) READPORT JMP, .END
13|
14|-->
15|
  +--------Block    287--------
 0|( BB coin routine         CHKCOIN1 )
 1|SUBR CHKCOIN1  ( chk door 1 for coin drop )
 2|  CNTM1 LDA, A ANA, 0=, IF, ( waiting for coin )
 3|  20 A MVI, 1020 D LXI, 14DF H LXI, debounce CALL,
 4|  A ANA, RZ, CNTM1 STA,   CREDITS H LXI, M INR,
 5|  UPCRED STA, A XRA, SPBON STA, DOVERB MCOIN 16 IN, RET,
 6|  ELSE, ( waiting for coin to drop ) 20 A MVI, 520 D LXI,
 7|    14DF H LXI, debounce CALL, A ANA, RNZ, CNTM1 STA,
 8|  17 IN, THEN, RET,
 9|-->
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    288--------
 0|( BB coin routine          CHKCOIN1 )
 1|SUBR CHKCOIN2  ( chk door 2 for coin drop )
 2|   CNTM2 LDA, A ANA, 0=, IF, ( waiting for coin )
 3|   10 A MVI, 1020 D LXI, 14EF H LXI, debounce CALL,
 4|   A ANA, RZ, CNTM2 STA, UPCRED STA,
 5|   CREDITS H LXI, 15 IN, 1 A BIT, 0<>, IF, ( susan b dollar )
 6|   M INR, M INR, M INR, M INR, M INR, ELSE,
 7|   M INR, THEN,
 8|   A XRA, SPBON STA, DOVERB MCOIN 16 IN, RET,
 9|   ELSE, ( waiting for coin to drop ) 10 A MVI, 520 D LXI,
10|     14EF H LXI, debounce CALL, A ANA, RNZ, CNTM2 STA,
11|   17 IN, THEN, RET,
12|DECIMAL ;S
13|
14|
15|
 +--------Block    290--------
 0|( I/O PORT DEFINES ) BASE@ HEX
 1|  9 C= HORCB   0A C= VERBL
 2|( MUSIC PORTS )
 3|10 C= TONMO   11 C= TONEA   12 C= TONEB   13 C= TONEC
 4|14 C= VIBRA   16 C= VOLAB   15 C= VOLC   17 C= VOLN   18 C= SNDBX
 5|0D C= INFBK   0E C= INMOD   0F C= INLIN   8 C= CONCM
 6|0C C= MAGIC   19 C= XPAND   8 C= INTST   0E C= VERAF
 7|0F C= HORAF
 8|-->
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    291--------
 0|( INTERRUPT ROUTINES ) HEX
 1|VPTR { @ 1 AND } VPTR { +! } ( ALIGN TO EVEN BOUNDARY )
 2|0 VARIABLE IPNT
 3|{ : <INTERRUPTS } DATA HERE { ; }
 4|{ : INTERRUPTS> } { [ } ASM { ] } JMP, { ; }
 5|: ]'[ [COMPILE] ' ;
 6|{ : I' } 0CD B, ]'[ { 2+ } , B, { ; }
 7|{ : ICODE } CODE { [ } ASM { ] } NEXT XTHL, D PUSH, B PUSH,
 8|   PSW PUSH, M A MOV, INLIN OUT, H INX, IPNT SHLD,
 9|   EXX, EXAF, H PUSH, D PUSH, B PUSH, PSW PUSH,
10|   ASM { ; }
11|CODE IEX PSW POP, B POP, D POP, H POP, EXX, EXAF,
12|    PSW POP, B POP, D POP, H POP, EI, RET,
13|{ : INEXT } { LIT [ } ' IEX { , } ASM { ] } JMP, { ; }
14|-->
15|
```

```
 +--------Block    292--------
 0|( Interrupt routines )
 1|CODE ISTART DI, H POP, IPNT SHLD, 8 A MVI, INMOD OUT,
 2|   IPNT ( SWAB ) A MVI, STAI, IPNT A MVI,
 3|      INFBK OUT, IM2, EI, NEXT
 4|CODE DI DI, NEXT CODE EI EI, NEXT
 5|CODE XDI DI, A XRA, INMOD OUT, NEXT
 6|CODE SWAN H POP, L A MOV, RLC, RLC, RLC, RLC, A L MOV,
 7|   H PUSH, NEXT ( SWAP NIBBLES IN LOW BYTE )
 8|-->
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    293--------
 0|( VGS screen handeling verbs   INTCOMMERCIAL , FILL , SCRERASE )
 1|: INTHIGHRES   ( intialize screen for commercial mode )
 2|   1 8 OUTP ( con,com port ) C0 0A OUTP ( vertbl )
 3|   0 9 OUTP ( horzcb ) ;
 4|: FILL   ( fill screen whith constant data )
 5|   ( in- constant , starting address , # of bytes to fill )
 6|   ( out- does sequential fill whith constant specified )
 7|   ROT ROT 2DUP ! SWAP DROP DUP 1+ ROT 1- BMOVE ;
 8|: SCRERASE   ( erase entire screen )
 9|   0 4000 3F00 FILL ;
10|-->
11|
12|
13|
14|
15|
 +--------Block    294--------
 0|( VGS        NDUP )
 1|CODE NDUP EXX, B POP, C DCR, 1 H LXI, SP DAD, B DAD, B DAD,
 2|   BEGIN, M D MOV, H DCX, M E MOV, D PUSH, H DCX, C DCR, 0<,
 3|   END, EXX, NEXT ( DUPLICATE TOP N ELEMENTS OF THE STACK )
 4|-->
 5|
 6|
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    295--------
 0|( HIGH SPEED RANDOM NUMBER ROUTINE )
 1|2 ARRAY RND# ( 4 BYTE RANDOM # BUFFER, SEED APPROPRIATELY ! )
 2|CODE RND EXX, 0 RND# LBCD, 1321 H LXI, B DAD, H PUSH,
 3| 2776 H LXI, B DADC, 1 RND# LDED, D DAD, XTHL,
 4| B DAD, XTHL, D DADC, XTHL, B DAD, XTHL, D DADC, XTHL,
 5| E D MOV, B E MOV, C B MOV, 0 C MVI, B DAD, 0 RND# SHLD,
 6| H POP, D DADC, 1 RND#  SHLD, EXX,
 7| 0 H LXI,,H D MOV, L E MOV, EXX, XCHG, B POP,
 8| 0 H LXI, BEGIN, B SRLR, C RARR, CY, IF, D DAD, EXX, D DADC,
 9| EXX, THEN, B A MOV, C ORA, <>, IF, E SLAR, D RALR, EXX, E RALR,
10| D RALR, EXX, SWAP JMP, THEN, EXX, H PUSH, NEXT
11|-->
12|
13|
14|
15|
 +--------Block    296--------
 0|( NUMBER TABLE FOR STRING DISPLAY ROUTINES )
 1|BTABLE characters ( FILL IN CHARACTER TABLES )
 2|( SPACE 0 1 2 3 4 5 6 7 8 9 )
 3|00 B, 00 B, 00 B, 00 B, 00 B, 00 B, 00 B, 00 B, 00 B, 00 B,
 4|3C B, 7E B, 66 B, 66 B, 66 B, 66 B, 66 B, 66 B, 7E B, 3C B,
 5|18 B, 38 B, 18 B, 18 B, 18 B, 18 B, 18 B, 18 B, 3C B, 3C B,
 6|3C B, 7E B, 66 B, 06 B, 3E B, 7C B, 60 B, 60 B, 7E B, 7E B,
 7|3C B, 7E B, 66 B, 06 B, 1C B, 1E B, 06 B, 66 B, 7E B, 3C B,
 8|66 B, 66 B, 66 B, 66 B, 7E B, 7E B, 06 B, 06 B, 06 B, 06 B,
 9|7C B, 7C B, 60 B, 60 B, 7C B, 7E B, 06 B, 66 B, 7E B, 3C B,
10|3C B, 7C B, 60 B, 60 B, 7C B, 7E B, 66 B, 66 B, 7E B, 3C B,
11|7E B, 7E B, 06 B, 0E B, 0C B, 1C B, 18 B, 38 B, 30 B, 30 B,
12|3C B, 7E B, 66 B, 66 B, 3C B, 7E B, 66 B, 66 B, 7E B, 3C B,
13|3C B, 7E B, 66 B, 66 B, 7E B, 3E B, 06 B, 06 B, 3E B, 3C B,
14|-->
15|
 +--------Block    297--------
 0|( CHARACTER PATTERN TABLE FOR DISPLAY )
 1|( A B C D E F G H I J K L M )
 2|18 B, 3C B, 7E B, 66 B, 66 B, 66 B, 7E B, 7E B, 66 B, 66 B,
 3|7C B, 7E B, 66 B, 66 B, 7C B, 7E B, 66 B, 66 B, 7E B, 7C B,
 4|3C B, 7E B, 66 B, 60 B, 60 B, 60 B, 60 B, 66 B, 7E B, 3C B,
 5|7C B, 7E B, 66 B, 66 B, 66 B, 66 B, 66 B, 66 B, 7E B, 7C B,
 6|7E B, 7E B, 60 B, 60 B, 7C B, 7C B, 60 B, 60 B, 7E B, 7E B,
 7|7E B, 7E B, 60 B, 60 B, 7C B, 7C B, 60 B, 60 B, 60 B, 60 B,
 8|3C B, 7E B, 60 B, 60 B, 60 B, 6E B, 6E B, 66 B, 7E B, 3C B,
 9|66 B, 66 B, 66 B, 66 B, 7E B, 7E B, 66 B, 66 B, 66 B, 66 B,
10|3C B, 3C B, 18 B, 18 B, 18 B, 18 B, 18 B, 18 B, 3C B, 3C B,
11|06 B, 06 B, 06 B, 06 B, 06 B, 06 B, 66 B, 66 B, 7E B, 3C B,
12|66 B, 66 B, 6E B, 7C B, 78 B, 78 B, 6C B, 6E B, 66 B, 66 B,
13|60 B, 60 B, 60 B, 60 B, 60 B, 60 B, 60 B, 60 B, 7E B, 7E B,
14|C3 B, E7 B, E7 B, DB B, DB B, C3 B, C3 B, C3 B, C3 B, C3 B,
15|-->
```

```
 +--------Block    298--------
 0|( CHARACTER PATTERN TABLE CONT. )
 1|( N O P Q R S T U V W X Y Z )
 2|66 B, 66 B, 76 B, 7E B, 7E B, 6E B, 66 B, 66 B, 66 B, 66 B,
 3|3C B, 7E B, 66 B, 66 B, 66 B, 66 B, 66 B, 66 B, 7E B, 3C B,
 4|7C B, 7E B, 66 B, 66 B, 7E B, 7C B, 60 B, 60 B, 60 B, 60 B,
 5|3C B, 7E B, 66 B, 66 B, 66 B, 66 B, 66 B, 6E B, 64 B, 3A B,
 6|7C B, 7E B, 66 B, 66 B, 7E B, 7C B, 6E B, 66 B, 66 B, 66 B,
 7|3C B, 7E B, 66 B, 60 B, 7C B, 3E B, 06 B, 66 B, 7E B, 3C B,
 8|7E B, 7E B, 18 B, 18 B, 18 B, 18 B, 18 B, 18 B, 18 B, 18 B,
 9|66 B, 66 B, 66 B, 66 B, 66 B, 66 B, 66 B, 66 B, 7E B, 3C B,
10|66 B, 66 B, 66 B, 66 B, 66 B, 7E B, 3C B, 3C B, 18 B, 18 B,
11|C3 B, C3 B, C3 B, DB B, DB B, DB B, FF B, E7 B, C3 B, C3 B,
12|66 B, 66 B, 7E B, 3C B, 18 B, 18 B, 3C B, 7E B, 66 B, 66 B,
13|66 B, 66 B, 7E B, 3C B, 18 B, 18 B, 18 B, 18 B, 18 B, 18 B,
14|7E B, 7E B, 06 B, 0E B, 1C B, 38 B, 70 B, 60 B, 7E B, 7E B,
15|-->
 +--------Block    299--------
 0|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
 1|BTABLE smallfont 00 B, 00 B, 00 B, 00 B, 00 B, ( SPACE )
 2|  040 B, 040 B, 040 B, 00 B, 040 B, ( ! )
 3|  0A0 B, 0A0 B, 00 B, 00 B, 00 B, ( " )
 4|  0A0 B, 0E0 B, 0A0 B, 0E0 B, 0A0 B, ( # )
 5|  040 B, 0E0 B, 080 B, 0E0 B, 040 B, ( % )
 6|  080 B, 020 B, 040 B, 080 B, 020 B, ( % )
 7|  00 B, 00 B, 40 B, 0A0 B, 0A0 B, ( & )
 8|  040 B, 040 B, 00 B, 00 B, 00 B, ( ' )
 9|  040 B, 080 B, 080 B, 080 B, 040 B, ( LEFT PAREN )
10|  040 B, 020 B, 020 B, 020 B, 040 B, ( RIGHT PAREN )
11|  00 B, 0A0 B, 040 B, 0A0 B, 00 B, ( * )
12|  00 B, 040 B, 0E0 B, 040 B, 00 B, ( + )
13|  00 B, 00 B, 00 B, 040 B, 080 B, ( , )
14|  00 B, 00 B, 0E0 B, 00 B, 00 B, ( - )
15|  00 B, 00 B, 00 B, 00 B, 40 B, ( . ) -->
 +--------Block    300--------
 0|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
 1|  00 B, 020 B, 040 B, 080 B, 00 B, ( / )
 2|  040 B, 0A0 B, 0A0 B, 0A0 B, 040 B, ( 00 )
 3|  040 B, 040 B, 040 B, 040 B, 040 B, ( 01 )
 4|  0E0 B, 020 B, 0E0 B, 080 B, 0E0 B, ( 2 )
 5|  0E0 B, 020 B, 060 B, 020 B, 0E0 B, ( 3 )
 6|  0A0 B, 0A0 B, 0E0 B, 020 B, 020 B, ( 4 )
 7|  0E0 B, 080 B, 0C0 B, 020 B, 0C0 B, ( 5 )
 8|  0E0 B, 080 B, 0E0 B, 0A0 B, 0E0 B, ( 6 )
 9|  0E0 B, 020 B, 040 B, 040 B, 040 B, ( 7 )
10|  0E0 B, 0A0 B, 0E0 B, 0A0 B, 0E0 B, ( 8 )
11|  0E0 B, 0A0 B, 0E0 B, 020 B, 0E0 B, ( 9 )
12|  00 B, 040 B, 00 B, 040 B, 00 B, ( : )
13|  00 B, 040 B, 00 B, 040 B, 080 B, ( ; )
14|  020 B, 040 B, 080 B, 040 B, 020 B, ( < )
15|  00 B, 0E0 B, 00 B, 0E0 B, 00 B, ( = )  -->
```

```
 +--------Block    301--------
 0|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
 1|  080 B, 040 B, 020 B, 040 B, 080 B, ( > )
 2|  0E0 B, 020 B, 040 B, 00 B, 040 B, ( ? )
 3|  0E0 B, 0A0 B, 0E0 B, 080 B, 0C0 B, ( @ )
 4|  0E0 B, 0A0 B, 0E0 B, 0A0 B, 0A0 B, ( A )
 5|  0E0 B, 0A0 B, 0C0 B, 0A0 B, 0E0 B, ( B )
 6|  0E0 B, 080 B, 080 B, 080 B, 0E0 B, ( C )
 7|  0C0 B, 0A0 B, 0A0 B, 0A0 B, 0C0 B, ( D )
 8|  0E0 B, 080 B, 0C0 B, 080 B, 0E0 B, ( E )
 9|  0E0 B, 080 B, 0C0 B, 080 B, 080 B, ( F )
10|  0E0 B, 080 B, 0A0 B, 0A0 B, 0E0 B, ( G )
11|  0A0 B, 0A0 B, 0E0 B, 0A0 B, 0A0 B, ( H )
12|  0E0 B, 040 B, 040 B, 040 B, 0E0 B, ( I )
13|  020 B, 020 B, 020 B, 0A0 B, 0E0 B, ( J )
14|  0A0 B, 0A0 B, 0C0 B, 0A0 B, 0A0 B, ( K )
15|  080 B, 080 B, 080 B, 080 B, 0E0 B, ( L ) -->
 +--------Block    302--------
 0|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
 1|  0A0 B, 0E0 B, 0E0 B, 0A0 B, 0A0 B, ( M )
 2|  0C0 B, 0A0 B, 0A0 B, 0A0 B, 0A0 B, ( N )
 3|  0E0 B, 0A0 B, 0A0 B, 0A0 B, 0E0 B, ( O )
 4|  0E0 B, 0A0 B, 0E0 B, 080 B, 080 B, ( P )
 5|  0E0 B, 0A0 B, 0A0 B, 0E0 B, 020 B, ( Q )
 6|  0C0 B, 0A0 B, 0C0 B, 0A0 B, 0A0 B, ( R )
 7|  0E0 B, 080 B, 0E0 B, 020 B, 0E0 B, ( S )
 8|  0E0 B, 040 B, 040 B, 040 B, 040 B, ( T )
 9|  0A0 B, 0A0 B, 0A0 B, 0A0 B, 0E0 B, ( U )
10|  0A0 B, 0A0 B, 0A0 B, 0A0 B, 040 B, ( V )
11|  0A0 B, 0A0 B, 0E0 B, 0E0 B, 0A0 B, ( W )
12|  0A0 B, 0A0 B, 040 B, 0A0 B, 0A0 B, ( X )
13|  0A0 B, 0A0 B, 040 B, 040 B, 040 B, ( Y )
14|  0E0 B, 020 B, 040 B, 080 B, 0E0 B, ( Z )
15|-->
 +--------Block    303--------
 0|( VGS-SMALL FONT CHARACTER SET 3 BY 5 )
 1|  0C0 B, 080 B, 080 B, 080 B, 0C0 B, ( [ )
 2|  00 B, 080 B, 040 B, 020 B, 00 B, ( BACK SLASH )
 3|  060 B, 020 B, 020 B, 020 B, 060 B, ( ] )
 4|  040 B, 0E0 B, 040 B, 040 B, 040 B, ( ^ )
 5|  020 B, 040 B, 0E0 B, 040 B, 020 B, ( RIGHT )
 6|BASE! ;S
 7|
 8|
 9|
10|
11|
12|
13|
14|
15|
```

```
 +--------Block    304--------
 0|( system verbs )
 1|: 1-B! DUP B@ 1- SWAP B! ; : 1+B! DUP B@ 1+ SWAP B! ;
 2|: S! SWAP ! ; : SB! SWAP B! ;
 3|( : V= ) 0 VARIABLE ( ; ) ( : BV= ) 0 BVARIABLE ( ; )
 4|( : C= ) CONSTANT (  ; ) ( : F= ) FORWARD ( ; )
 5|( : LABLE ) DATA ( ; )
 6|CODE EIDI EI, DI, NEXT
 7|( : DTC ) DECIMAL EDIT ( ; )
 8|;S
 9|
10|
11|
12|
13|
14|
15|
 +--------Block    305--------
 0|( BB sentry string routines  DPCN CHKGMCNT ) HEX
 1|: DP1CN 400 2800 828 A" DEPOSIT 1 COIN TO CONTINUE THIS GAME"
 2|  SPOST ;
 3|: DP12CN CNSW1 B@ IF 300 2800 828
 4| A" DEPOSIT 2 COINS TO CONTINUE THIS GAME" SPOST
 5|  ELSE DP1CN THEN ;
 6|: DPCN ( wait for next coin )
 7|  DP1CN CREDITS BZERO FF TWAIT FF TWAIT DP1CN CREDITS B@ IF
 8| GAMEOVER BZERO ELSE LINN BZERO UPCRED BONE THEN CREDITS 1+B! ;
 9|: CHKGMCNT DP12CN C0 TWAIT DP12CN CREDITS B@ ;
10|: INSWAIT EI BEGIN FF 0 DO LOOP
11|  14 INP 0F AND 0F <> IF DROP 1 INSOUT BONE THEN
12|  1- DUP 0= END DROP DI ;
13|: YOU1ST 1A00 1000 828 A" YOU ARE UP 1ST" SPOST 0F00 3000 828
14|  A" BEAT ME FOR EXTRA INNING" SPOST 1F00 5000 828
15|  A" GOOD LUCK" SPOST ; -->
 +--------Block    306--------
 0|( BB sentry string routines  INSTRC )
 1|: INSTRC  ( instructions )
 2|  INSOUT BZERO BEGIN
 3|  FLFILL 2300 1000 828 A" PITCH" SPOST 700 4000 828
 4|  A" PRESS PITCH BUTTON TO START PITCH" SPOST 0F00 6000 828
 5|  A" ROLLERBALL CONTROLS PITCH" SPOST 0C00 8000 828
 6|  A" ROLLERBALL MOVES OUTFIELDERS" SPOST
 7|  80 INSWAIT INSOUT B@ IF ELSE
 8|  FLFILL 2500 1000 828 A" BAT" SPOST B00 4000 828
 9|  A" PRESS BAT BUTTON TO SWING BAT" SPOST 500 6000 828
10|  A" HOLD BUTTON DOWN TO ADVANCE RUNNERS" SPOST
11|  60 INSWAIT INSOUT B@ IF ELSE
12|  FLFILL 0A00 2800 828 A" PRESS ANY BUTTON TO START GAME" SPOST
13|  30 INSWAIT THEN THEN INSOUT B@ END ;
14|DECIMAL ;S
15|
```