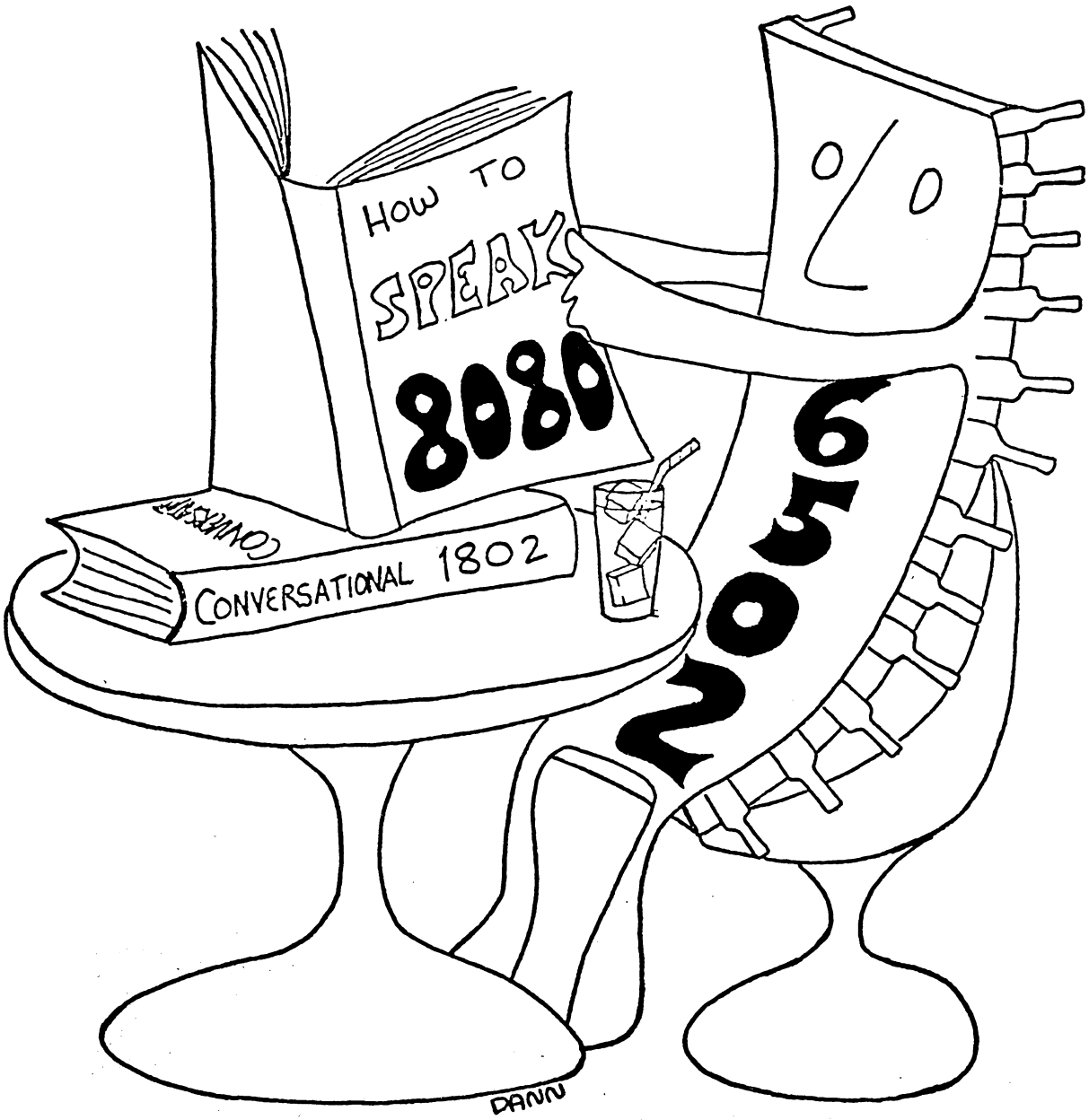
Dann McCreary

AN 8080 SIMULATOR for the 6502

KIM-1 VERSION



AN 8080 SIMULATOR FOR THE 6502
USER'S MANUAL
KIM - 1 VERSION

THE 8080 SIMULATOR ENABLES A 6502 MICROPROCESSOR TO EXECUTE THE 8080 INSTRUCTION SET. THE SIMULATOR DOES THIS BY INTERPRETING 8080 INSTRUCTIONS IN A NORMAL PROGRAM SEQUENCE. ALL INTERNAL 8080 REGISTERS ARE AVAILABLE FOR EXAMINATION AT ANY TIME. THEY MAY BE VIEWED STATICLY IN A SINGLE-STEP MODE OR DYNAMICLY IN A TRACE MODE. ALL 8080 SOFTWARE FEATURES ARE PRESENTLY SUPPORTED WITH THE EXCEPTION OF DIRECT MEMORY ACCESS (DMA).

THIS SIMULATOR VERSION RUNS ON A BASIC KIM - 1. IN ITS MINIMUM CONFIGURATION, THE 8080 SIMULATOR PROVIDES SINGLE STEP AND FULL SPEED MODES AND LEAVES 227 BYTES OF MEMORY OPEN FOR 8080 PROGRAMS. ALTERNATIVE RUN MODE FEATURES TAKE UP ADDITIONAL (OPTIONAL) MEMORY SPACE.

THE CASSETTE TAPE INCLUDED IN THIS PACKAGE IS RECORDED IN STANDARD KIM - 1 FORMAT. THE PURCHASER OF THIS PACKAGE IS HEREWITH GRANTED PERMISSION TO MAKE ONE COPY FOR HIS/HER OWN PERSONAL USE, RETAINING THE ORIGINAL AS A BACKUP.

THE AUTHOR WELCOMES ANY FEEDBACK FROM SIMULATOR USERS AND WILL BE PLEASED TO RESPOND TO SPECIFIC QUESTIONS WHICH ARE ACCOMPANIED BY A STAMPED, SELF-ADDRESSED ENVELOPE.

ADDITIONAL COPIES ARE AVAILABLE, CONSISTING OF A KIM - 1 FORMAT CASSETTE TAPE, A USER MANUAL AND A COMPLETE, WELL COMMENTED ASSEMBLY LEVEL SOURCE/OBJECT LISTING, PRICED AT $20.00 + $1.50 POSTAGE & HANDLING (CALIFORNIA RESIDENTS PLEASE ADD 6% SALES TAX), THEY MAY BE ORDERED FROM

DANN MCCREARY
P.O.BOX 16435-S
SAN DIEGO, CA 92116

AS AN ADEQUATE TREATMENT OF THE INSTRUCTION SET AND PROGRAMING TECHNIQUES FOR THE 8080 MICROPROCESSOR IS BEYOND THE SCOPE OF THIS MANUAL, IT IS SUGGESTED THAT THE USER AQUIRE THE "8080/8085 ASSEMBLY LANGUAGE PROGRAMMING MANUAL" FROM INTEL.

ACKNOWLEDGEMENT: THANKS TO GARY DAVIS FOR GENEROUS SUPPORT IN THE FORM OF ACCESS TO HIS 8080 SYSTEM AND HIS ASSISTANCE IN RUNNING COMPARISON TESTS.

INTRODUCTION

WHY IMITATE ONE MICROPROCESSOR WITH ANOTHER? YOU PROBABLY PURCHASED THIS 8080 SIMULATOR PACKAGE TO DO ONE OR MORE OF THE FOLLOWING:

- -RUN EXISTING 8080 SOFTWARE ON YOUR 6502
- -WRITE, TEST AND DEBUG YOUR OWN 8080 SOFTWARE WITHOUT HAVING TO PURCHASE A COMPLETE 8080 BASED SYSTEM
- -LEARN SOMETHING ABOUT THE ARCHITECTURE AND INSTRUCTION SET OF THE 8080 VIA HANDS-ON EXPERIENCE

BY SPENDING A LITTLE TIME WITH THIS MANUAL AND GOING THROUGH THE STEP BY STEP "FIRST TIME AROUND" PROCEEDURE, YOU WILL SOON BE DOING ALL THIS AND MORE, LATER, WHEN YOU KNOW THE CAPABILITIES AND LIMITATIONS OF THE SIMULATOR, YOU HAVE THE OPTION OF RELOCATING IT PERMANENTLY IN ROM.

THE ESSENTIAL PART OF THE 8080 SIMULATOR IS THE SUBROUTINE CALLED "SIM80", EACH TIME THIS SUBROUTINE IS CALLED, ANOTHER 8080 INSTRUCTION IS EXECUTED. PROGRAMS OF VARYING COMPLEXITY MAY CALL AND USE "SIM80", DEPENDING ON YOUR OBJECTIVES. A TYPICAL EXAMPLE OF SUCH A PROGRAM IS INCLUDED ON PAGE THREE OF THE SIMULATOR PROGRAM LISTINGS. IN CONJUNCTION WITH A SIMPLE SET OF SWITCHES, IT PROVIDES FOR EASY SELECTION OF RUN, TRACE OR SINGLE STEP MODES AND BREAPOINT OPERATION. ALTERNATELY, THE SIMULATOR MAY BE RUN IN A MINIMUM MODE IN ORDER TO LEAVE A GRFATER AMOUNT OF MEMORY FOR 8080 PROGRAMS.

AFTER STUDYING THESE CALLING PROGRAMS YOU MAY WISH TO DESIGN YOUR OWN SPECIAL PURPOSE CALLING PROGRAM. FOR EXAMPLE, TRY A "SUPER BREAKPOINT" PROGRAM. ALL SIMULATED 8080 REGISTERS ARE MAINTAINED IN RAM. YOU CAN MONITOR THEM BETWEEN 8080 INSTRUCTIONS AND FORCE A BREAKPOINT ON THE BASIS OF A PARTICULAR PATTERN.

I/O HANDLING IS FACILITATED BY A SPECIAL INSTRUCTION DESIGNED INTO THE SIMULATOR. THE "C65" INSTRUCTION ENABLES YOU TO CALL 6502 CODED SUBROUTINES FROM YOUR 8080 PROGRAM. WITH SLIGHT PRECAUTIONS, THEN, YOU CAN USE ALL YOUR SYSTEM MONITOR'S SUBROUTINES.

INCLUDED IN THE SIMULATOR LISTINGS IS AN 8080 TIME-OF-DAY CLOCK PROGRAM. BY USING THE SIMULATOR TO RUN THIS PROGRAM, YOU WILL SEE BY EXAMPLE ALL THE VARIOUS MODES AND FEATURES PROVIDED BY THE 8080 SIMULATOR. SO, WHY NOT GET STARTED?

TAPE LOADING INSTRUCTIONS

THE 8080 SIMULATOR KIM - 1 FORMAT CASSETTE TAPE CONSISTS OF 4 BLOCKS OF PROGRAMS AND DATA,

BLOCK 1 IS THE 8080 REGISTER AREA AND INTERPRETER ROUTINES, ADDRESSES #$00E3 TO #$03FF,

BLOCK 2 IS THE MAIN CONTROL LOOP, ADDRESSES #$1780 TO #$17E6,

BLOCK 3 IS THE OPTIONAL EXECUTIVE AND INTERRUPT AREA, ADDRESSES #$0076 TO #$00E2,

BLOCK 4 IS A SAMPLE 8080 PROGRAM, ADDRESSES #$0000 TO #$0075,

THE KIM DATA BLOCK I.D. FOR ALL BLOCKS IS #$80,

LOADING PROCEEDURE

ACTION	SEE DISPLAYED
(1) *POWER ON KIM & CASSETTE INTERFACE*	
(2) *REWIND CASSETTE*	
(3) PUSH [RS], [AD] 00F1	00F1 XX
(4) PUSH [DA], 00	00F1 00
(5) PUSH [AD], 17F9	17F9 XX
(6) PUSH [DA], 80	17F9 80
(7) *START CASSETTE PLAYER*	
(8) PUSH [AD], 1873	1873 A9
(9) PUSH [GO]	BLANK DISPLAY
(10) *DISPLAY RELIGHTS*	0000 XX
(14) *REPEAT STEPS 8 - 10 FOR REMANING BLOCKS*	
(15) PUSH [AD], 00F1	00F1 00
(16) PUSH [DA], FF	00F1 FF
(17) PUSH [AD],17FA	17FA XX
(18) PUSH [DA], 80	17FA 80
(19) PUSH [+]	17FB XX
(20) PUSH 17	17FB 17

OPERATING INSTRUCTIONS

TO FAMILIARIZE YOURSELF WITH 8080 SIMULATOR OPERATION LOAD TAPE BLOCKS 1 - 4 AND WORK YOUR WAY THROUGH THE FOLLOWING INSTRUCTIONS.

ONCE THE TAPE IS LOADED, BE SURE TO PUT #$FF AT ADDRESS #$00F1. NOW LOAD THE MINIMUM CONFIGURATION INTERRUPT VECTORS: #$80 INTO LOCATION #$17FA AND #$17 INTO #$17FB, THIS IS THE NMI VECTOR WHICH IS ACTIVATED BY THE [ST] KEY ON KIM. LOAD #$22 INTO #$17FE & #$1C INTO #$17FF FOR THE IRQ VECTOR.

MINIMUM CONFIGURATION
SINGLE STEP MODE

CHANGE LOCATIONS #$178E - #$1790 TO #$4C,#$22,#$1C,THE PROGRAM COUNTER SHOULD CONTAIN #$0000 (#$00 IN LOCATION #$00EF AND #$00 IN LOCATION #$00F0), IT POINTS TO THE FIRST BYTE OF 8080 PROGRAM AREA WHICH NOW CONTAINS A SAMPLE 8080 PROGRAM.

LOOK AT LOCATION #$00EF, THE LOW ORDER BYTE OF THE PROGRAM COUNTER, PUSH THE [ST] KEY AND YOU WILL SEE IT CHANGE FROM #$00 TO #$03, EACH SUBSEQUENT OPERATION OF THE [ST] KEY WILL ADVANCE THE SIMULATOR THROUGH ANOTHER PROGRAM STEP.

WITH THE DEMO PROGRAM LISTING IN HAND YOU CAN VERIFY THAT THE PROGRAM IS FOLOWING ITS PROPER SEQUENCE, BETWEEN EACH PROGRAM STEP CONTROL IS RETURNED TO THE KIM MONITOR PROGRAM, THIS MAKES IT POSSIBLE TO VIEW OR CHANGE THE CONTENTS OF ANY SIMULATED 8080 REGISTER, FOR INSTANCE, LOOK AT LOCATION #$00E5, THIS IS THE 8080 ACCUMULATOR, NOW CONTINUE PRESSING THE [ST] KEY AND YOU CAN WATCH DATA MOVING INTO THE AC-CUMULATOR, BEING INCREMENTED AND SO ON. BY USING THE [DA] AND HEX KEYS BETWEEN PROGRAM STEPS YOU CAN MODIFY THAT DATA TO SEE WHAT EFFECT THAT MAY HAVE ON PROGRAM EXECUTION. NOTE: THIS PROGRAM INTFRACTS WITH THE KIM DISPLAY BUFFER, SO BE PREPARED FOR THE DISPLAY TO CHANGE AFTER THE PROGRAM REACHES ADDRESS #$002B.

PC SINGLE STEP

THE 8080 PROGRAM COUNTER IS LOCATED ON PAGE ZERO IN THE SAME PLACE KIM MONITOR ROUTINES STORE THE 6502 PROGRAM COUNTER. IF YOU PUSH THE [PC] KEY BETWEEN INSTRUCTION STEPS, YOU WILL SEE THE CURRENT 8080 PROGRAM ADDRESS AND OP-CODE DISPLAYED. BY PLACING #$4C,#$DC,#$1C AT ADDRESS #$178E - #$1790, THIS WILL BE DONE FOR YOU BY THE KIM MONITOR.

RUN MODE

TO RUN THE SAMPLE PROGRAM AT FULL SPEED JUST CHANGE THE DATA AT LOCATION #$178E - #$1790 TO #$58,#$D0,#$EF. NOW WHEN YOU PUSH [ST] THE SIMULATOR WILL START AND CONTINUE RUNNING UNTIL YOU PUSH [RS] OR ACTIVATE THE IRQ. USING THE IRQ IS RECOMMENDED, AS [RS] WILL INTERRUPT THE SIMULATOR, LEAVING ITS REGISTERS IN AN UNKNOWN STATE. NOTE: MINIMUM CONFIGURATION DOES NOT SUPPORT INTERRUPTS BUT IT DOES EXECUTE INTERRUPT-RELATED INSTRUCTIONS.

ILLEGAL OP-CODES & BREAKPOINTS

WHEN THE 8080 SIMULATOR ENCOUNTERS AN ILLEGAL OP-CODE, A JUMP WILL BE FORCED TO THE SYSTEM MONITOR, IN THE MINIMUM CONFIGURATION THIS MAY BE USED TO ADVANTAGE BY INSERTING AN ILLEGAL OP-CODE (TRY #$10) IN PLACE OF ANY OP-CODE IN YOUR 8080 PROGRAM, THIS WILL ACT AS A BREAKPOINT, TO CONTINUE OPERATION AFTER A BREAKPOINT, REPLACE THE ORIGINAL OP-CODE, SUBTRACT 1 FROM THE 8080 PROGRAM COUNTER, AND CONTINUE NORMALLY, (NOTE: DO NOT USE #$CB AS A BREAK.)

OPTIONAL FEATURES

USING THE SIMULATOR'S OPTIONAL FEATURES REQUIRES THE ILLUSTRATED SWITCHING ARRANGEMENT CONNECTED TO KIM PORT PB, IN PARTICULAR, DO NOT ATTEMPT TO USE THE INTERRUPT SUBROUTINE WITHOUT THE APPROPRIATE SETUP, OMMITTING THE MODE CONTROL SWITCHES HOWEVER WILL ONLY RESULT IN A DEFAULT TO THE SINGLE STEP MODE, WITH SWITCHES CONNECTED, PUT #$BD, #$00 INTO #$17FA & #$17FB, PUT #$58, #$60, #$EA BACK INTO #$178F - #$1790,

REGISTER SINGLE STEP MODE

SET MODE CONTROL SWITCHES (LEFT TO RIGHT) TO REGISTER, STEP & STEP, THIS POSITION IS IDENTICAL IN OPERATION TO THE MINIMUM CONFIGURATION SINGLE STEP MODE,

REGISTER TRACE MODE

SET CONTROL SWITCHES TO REGISTER, TRACE & STEP, WHEN YOU ACTIVATE THE [ST] KEY THE SIMULATOR WILL BEGIN EXECUTING 8080 INSTRUCTIONS AT A RATE DETERMINED BY THE TIME-DELAY VALUE STORED IN "SPEED", ADDRESS #$00E2,

PROGRAM COUNTER TRACE MODE

SET CONTROL SWITCHES TO PC, TRACE & STEP, NOW THE DISPLAY WILL TRACE THE PROGRESS OF THE 8080 PROGRAM COUNTER AND THE 8080 PROGRAM OP-CODES,

PROGRAM COUNTER SINGLE STEP MODE

SET CONTROL SWITCHES TO PC, STEP & STEP, NOW PUSHING THE [ST] KEY IS REQUIRED TO ADVANCE THE PROGRAM COUNTER,

TRACE SPEED

TRACE SPEED MAY BE SET BY INSERTING A VALUE IN "SPEED", #$00E2, A VALUE OF #$00 GIVES THE SLOWEST TRACE, AND #$01 THE FASTEST,

RUN MODE

MOVE THE RUN SWITCH TO "RUN", PUSH [ST] AND YOUR PROGRAM WILL RUN AT FULL SPEED, TRY SETTING THE DEMO PROGRAM CLOCK BY KEYING IN THE TIME IN RAPID SUCCESSION, MOVE THE RUN SWITCH BACK TO REVERT TO TRACE OR SINGLE STEP,

INTERRUPT ACTION

INSERT #$20, #$76, #$00 AT #$179F - #$17A1 TO USE INTERRUPTS. WHEN PB7 IS BROUGHT HIGH AN 8080 INTERRUPT WILL OCCUR (NOTE: THE PB7 INTERRUPT LINE MUST BE HELD HIGH THROUGH AT LEAST ONE SIMULATOR CYCLE FOR AN INTERRUPT TO TAKE PLACE). NO INTERRUPT ACKNOWLEDGE LINE IS PROVIDED - IF ONE IS REQUIRED, INSERT AN OUTPUT INSTRUCTION JUST BEFORE "NOINT" IN THE INTERRUPT LOGIC ON PAGE 2 OF THE LISTINGS.

THE "INTE" INTERRUPT FLAG (ADDRESS #$00E3) IS OPPOSITE IN SENSE FROM THE 8080 INTERRUPT FLAG - I.E., IT DISABLES INTERRUPTS WHEN SET TO ONE AND ENABLES THEM WHEN SET TO ZERO, IT ALSO GOES THROUGH A TRANSITION STATE OF #$FF AFTER AN "EI" INSTRUCTION IS EXECUTED. THE RESULTANT ACTION, HOWEVER, IS IDENTICAL TO THAT OF AN ACTUAL 8080. THE INTERRUPT VECTOR IS SET BY SWITCHES AT PB 3,4,&5.

THE 8080 "HALT" INSTRUCTION

#$76 WILL CAUSE AN 8080 HALT. IN AN ACTUAL 8080, THE PROCESSOR HALTS WITH THE PROGRAM COUNTER POINTING TO THE NEXT INSTRUCTION IN SEQUENCE. ON THE SIMULTOR, HOWEVER, IN ORDER TO MAINTAIN CONTROL AND TO KEEP MONITORING FOR SIMULATED INTERRUPTS, A #$76 WILL RESULT IN NO ADVANCEMENT OF THE PROGRAM COUNTER - RATHER, THE #$76 WILL BE REPEATEDLY EXECUTED UNTIL AN INTERRUPT IS DETECTED OR UNTIL THE SIMULATOR IS HALTED EXTERNALLY. AFTER AN INTERRUPT IS SERVICED, EXECUTION WILL CONTINUE WITH THE INSTRUCTION IMMEDIATELY FOLLOWING THE HALT.

BREAKPOINT OPERATION

THE 8080 SIMULATOR ALLOWS MULTIPLE BREAKPOINTS. STORE THESE BREAKPOINT ADDRESSES IN "BKTBL", A RAM AREA. START PROGRAM EXECUTION IN THE NORMAL MANNER. WHEN THE PROGRAM REACHES THE BREAKPOINT A JUMP WILL BE FORCED, RETURNING CONTROL TO THE KIM MONITOR. YOU MAY NOW EXAMINE OR MODIFY ANY 8080 REGISTERS. TO CONTINUE OPERATION, ALL THAT IS REQUIRED IS TO PUSH [ST]. THE SIMULATOR WILL RUN UNTIL THE NEXT BREAK IS ENCOUNTERED. "BKTBL" MAY BE EXPANDED UP TO 128 ENTRIES.

SIMULATOR I/O HANDLING

NORMAL SIMULATOR I/O INSTRUCTIONS ARE HANDLED VIA "IOTBL". THIS IS A TABLE OF ADDRESSES OF INPUT/OUTPUT PORTS. NOTE THAT THESE I/O ADDRESSES ARE STORED IN NORMAL ORDER, I.E., MOST SIGNIFIGANT ADDRESS FIRST. THE SIMULATOR ASSUMES THAT THERE IS A DATA DIRECTION REGISTER AT THE PORT ADDRESS +1. IF YOU ASSIGN RAM LOCATIONS AS SIMULATED I/O PORT ADDRESSES, 2 LOCATIONS MUST BE ALLOWED FOR EACH PORT.

THE FIRST ENTRY IN "IOTBL" WILL BE 8080 PORT 0, THE SECOND ENTRY WILL BE 8080 PORT 1, AND SO ON UP TO A MAXIMUM OF PORT 127.

THIS VERSION OF THE SIMULATOR ASSIGNS ALL I/O INSTRUCTIONS TO KIM PORTS A & SA. PORT SA IS ALSO SHARED BY THE KIM KEYPAD, SO TO AVOID INTERFERENCE DO NOT USE THE KEYBOARD (EXCEPT FOR [ST]) WHILE INPUT INSTRUCTIONS ARE BEING USED.

THE 8080 "RST" INSTRUCTION

THE SIMULATOR SENDS RST CALLS TO PAGE ZERO, AS DOES THE 8080, IF HOWEVER YOUR PAGE ZERO IS TAKEN UP WITH MONITOR ROUTINES, ETC., THE RESET PAGE MAY BE CHANGED BY CHANGING "RSTHI" FROM #$00 TO WHATEVER PAGE YOU WISH, IT IS LOCATED AT ADDRESS #$0189 ON PAGE 9 OF THE SIMULATOR LISTINGS.

THE "CALL6502", OR "C65" OP-CODE

THE SIMULATOR APPROPRIATES ONE OF THE 8080'S UNIMPLEMENTED OP-CODES FOR SPECIAL USE, IT IS #$CB, AND TAKES THE FORM "CB XXXX", WHERE XXXX IS THE ADDRESS -1 OF ANY 6502 SUBROUTINE. SUBROUTINES WHICH DON'T REQUIRE ANY PARAMETERS MAY BE CALLED DIRECTLY, SUBROUTINES WHICH REQUIRE SOME DATA PASSED TO OR FROM THEM MUST BE PREFACED OR ENDED WITH CODE TO MOVE THE REQUIRED DATA INTO OR OUT OF APPROPRIATE 8080 REGISTERS, SEE AS AN EXAMPLE THE "DSKBD" SUBROUTINE ON PAGE 20 OF THE PROGRAM LISTINGS. IT DOESN'T NEED ANY DATA PASSED TO IT, BUT MUST RETURN A KEY VALUE IN A. BE SURE TO SPECIFY THE ADDRESS MINUS ONE (ADDR,-1) WHEN USING "C65".

RELOCATION INFORMATION

THE 8080 SIMULATOR IN ITS MINIMUM CONFIGURATION DOES NOT RELY ON ANY KIM MONITOR SUBROUTINES PER-SE, RATHER, CONTROL IS PASSED BACK AND FORTH BETWEEN THE SIMULATOR AND THE KIM MONITOR. IT IS EASILY RELOCATED WITHIN A KIM SYSTEM AND EASILY ADAPTED TO OTHER NON - KIM 6502 SYSTEMS, THE SIMULATOR CAN BE RELOCATED IN ROM WITH THE EXCEPTION OF THE REGISTER AREA AND OTHER ZERO-PAGE VARIABLES.
IF YOU DON'T HAVE ACCESS TO AN ASSEMBLER BUT WOULD LIKE TO PUT THE SIMULATOR IN HIGH MEMORY, LEAVE THE RELATIVE POSITIONS OF THE MAJOR ROUTINES (#$0100 - #$03FF) THE SAME, AND JUST BE SURE TO CHANGE ALL THE HIGH ORDER ADDRESSES (I.E., THE THIRD BYTE OF ANY THREE BYTE INSTRUCTION.)

TO USE THE 8080 SIMULATOR IN A NON - KIM 6502 SYSTEM ADDITIONAL MODIFICATIONS ARE NECESSARY, ALL PROGRAM LINES WHICH REFERENCE KIM MONITOR-RESIDENT DATA HAVE THE WORD "KIM" SOMEWHERE IN THE COMMENT AREA, THESE INCLUDE I/O ADDRESSES AND SUBROUTINE CALLS. I/O ADDRESSES SHOULD BE CHANGED TO AVAILABLE PORTS IN YOUR SYSTEM.

ONE SUBROUTINE CALL IS TO SCAND, SCAND IS A MONITOR SUBROUTINE WHICH DISPLAYS THE ADDRESS SPECIFIED BY "POINTL" AND "POINTH" ALONG WITH THE DATA BYTE AT THAT ADDRESS, SUBSTITUTE A SIMILAR DISPLAY ROUTINE FROM YOUR SYSTEM MONITOR, GETKEY READS A ONE BYTE VALUE FROM KIM'S HEXADECIMAL KEYPAD, ANY COMPARABLE INPUT ROUTINE IN YOUR MONITOR WILL DO.

GENERAL OPERATING CONSIDERATIONS

6502 ADDRESSING MODES MAKE HEAVY USE OF ZERO-PAGE RESOURCES AND SINCE PAGE ONE IS USED FOR A STACK, 8080 PROGRAMS MUST BYPASS THESE AREAS. IF THE 8080 APPLICATION PROGRAM YOU WISH TO RUN USES #$00E0 - #$00FF OR #$01F0 TO #$01FF, RELOCATE THAT PORTION OF THE PROGRAM IN ANOTHER AREA AND PROVIDE JUMPS TO AND FROM THE NEW AREA.

SIMULATOR PROGRAMS RUN CONSIDERABLY SLOWER THAN THE SAME PROGRAM WOULD RUN ON ACTUAL 8080 HARDWARE. THIS IS, OF COURSE, BECAUSE THE SIMULATOR MUST EXECUTE MANY 6502 INSTRUCTIONS IN THE COURSE OF EXECUTING ONE 8080 INSTRUCTION. DON'T EXPECT BLINDING SPEED! ON THE OTHER HAND, TIME DEPENDENT EVENTS OF MODERATE FREQUENCY (SUCH AS THE INCLUDED CLOCK PROGRAM) CAN BE HANDLED REASONABLY WELL. IF YOU INTEND TO TRANSFER PROGRAMS TO AN ACTUAL 8080 BASED SYSTEM, BE SURE TO MAKE ALLOWANCES IN YOUR TIME DELAY ROUTINES. FOR HIGHLY TIME DEPENDENT EVENTS, USE THE "C65" INSTRUCTION TO HANDLE THEM IN 6502 CODE.

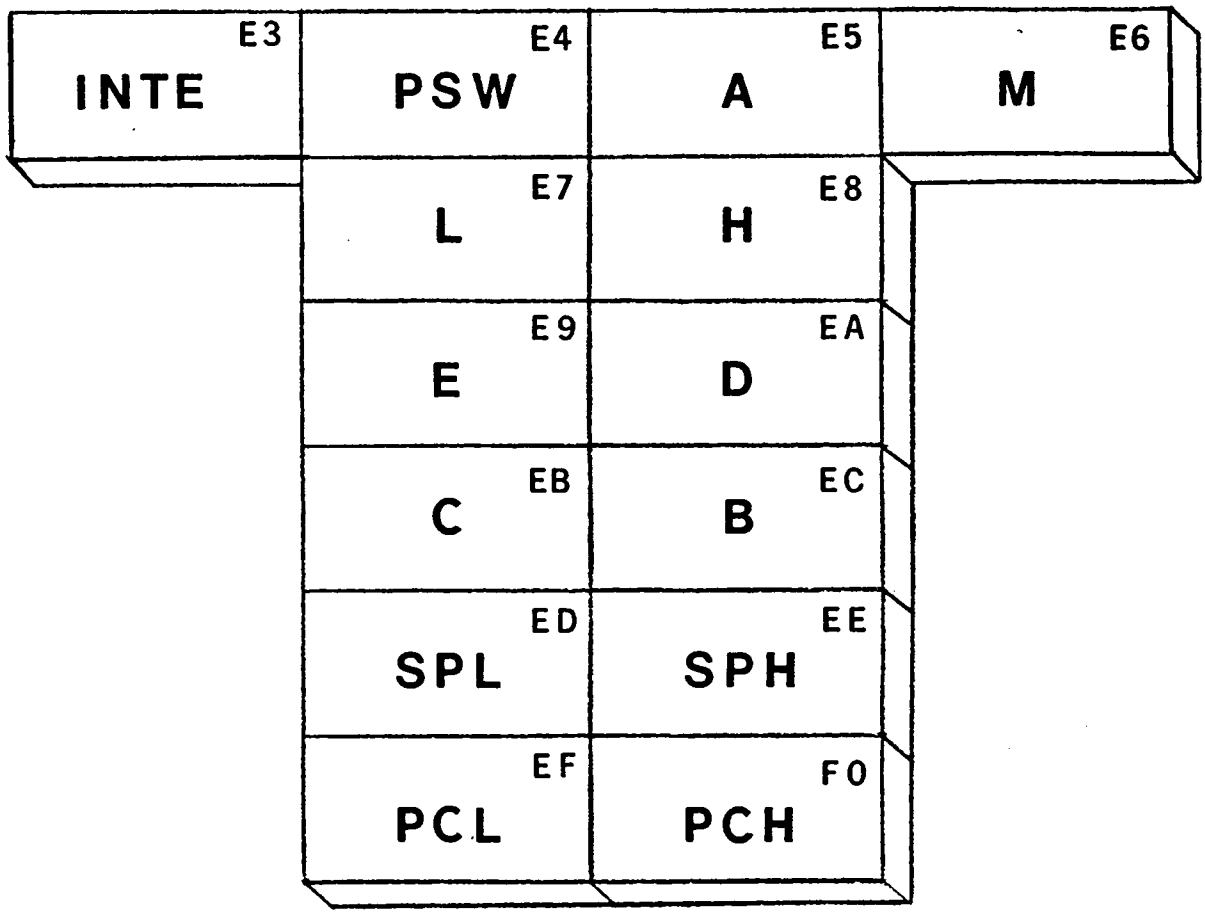
UNLIKE AN ACTUAL 8080 PROCESSOR, SIMULATOR REGISTERS ARE MAINTAINED IN MEMORY. THIS REQUIRES CARE THAT YOUR 8080 PROGRAM NOT ACCESS OR MODIFY THESE REGISTERS. THE SAME ALSO HOLDS TRUE FOR THE SIMULATOR PROGRAM ITSELF. AVOID REFERENCING ANY SIMULATOR PROGRAM AREA FROM YOUR 8080 PROGRAM. IF YOU SUSPECT THIS MAY HAVE HAPPENED, RELOAD THE SIMULATOR FROM TAPE.

AN EASY TRAP FOR 8080 PROGRAMMERS TO FALL INTO IS "ACCIDENTAL INITIALIZATION" OF REGISTERS. THESE ARE THE SYMPTOMS: YOU HAVE COMPLETED WORK ON AN 8080 PROGRAM. IT RUNS PERFECTLY ON THE SIMULATOR. YOU LOAD IT INTO THE TARGET SYSTEM, HIT "GO" AND NOTHING HAPPENS. WHY?

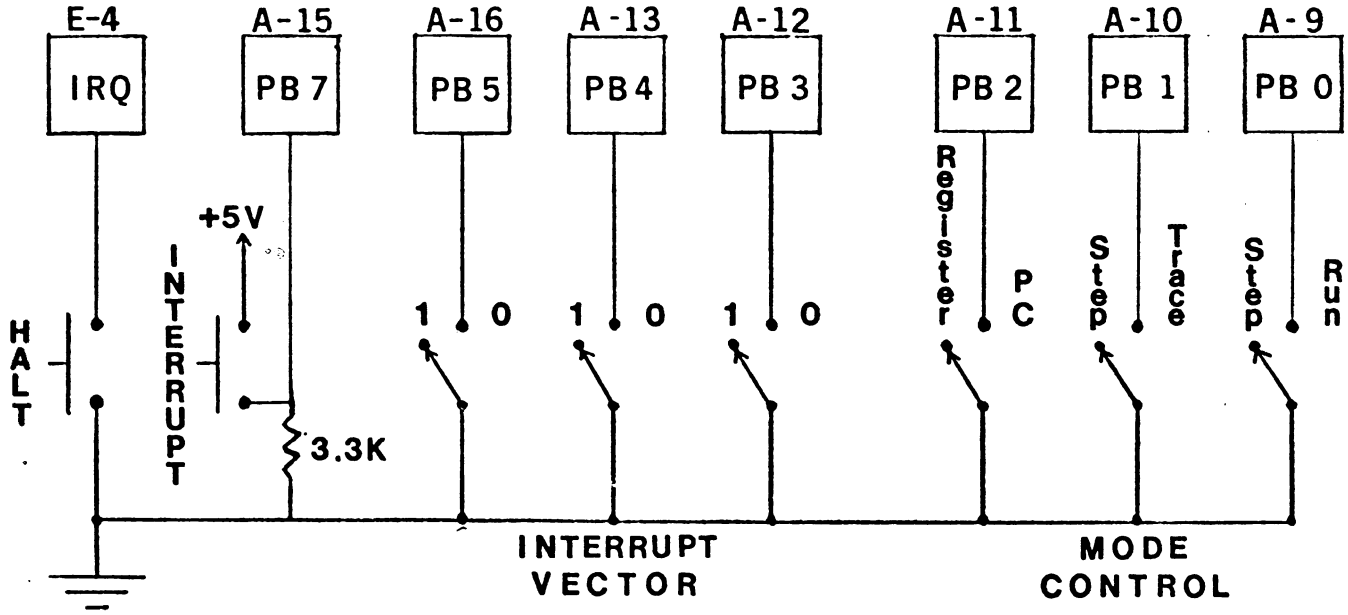
TO OPERATE PROPERLY YOUR 8080 PROGRAM MAY REQUIRE THAT ONE OR MORE REGISTERS BE INITIALIZED TO CERTAIN VALUES OR RANGES OF VALUES. THE SIMULATOR REGISTERS MAY BE PRE-SET TO THOSE CONDITIONS FROM PRIOR OPERATION. THUS YOUR PROGRAM MAY RUN ON THE SIMULATOR EVEN THOUGH IT HAS NO INSTRUCTIONS TO PROPERLY INITIALIZE THOSE REGISTERS. THIS MAY EVEN HAPPEN ON YOUR TARGET SYSTEM. DUE TO RANDOM VARIATIONS FROM ONE PROCESSOR CHIP TO ANOTHER, EACH PROCESSOR MAY "COME UP" WITH DIFFERENT RANDOM VALUES IN UNINITIALIZED REGISTERS. THE RESULT? YOUR PROGRAM WORKS FINE WITH SOME PROCESSORS BUT NOT AT ALL WITH OTHERS.

TO AVOID THIS PITFALL, TAKE THESE STEPS: BEFORE EACH SERIOUS EVALUATION OF AN 8080 PROGRAM RELOAD THE 8080 REGISTER AREA TO A POWER-ON CONFIGURATION WITH RANDOM DATA IN THE REGISTERS. TRY SEVERAL PROGRAM RUNS WITH DIFFERENT RANDOM DATA IN THE REGISTERS AT STARTUP. THIS WILL ASSURE YOU THAT YOUR INITIALIZATION SEQUENCES ARE WORKING AS YOU EXPECT.

8080 SIMULATOR REGISTER MAP



CONTROL CONFIGURATION



```
**************************************************************
************ AN 8080 SIMULATOR FOR THE 6502 ************
********************  KIM-1 VERSION  ********************
**************************************************************
* THIS PROGRAM SIMULATES THE OPERATION OF 8080 MICRO-
* PROCESSOR SOFTWARE ON A KIM - 1 6502 MICROCOMPUTER.
* IN ORDER TO SIMPLIFY RELOCATION TO ANOTHER NON -
* KIM 6502 SYSTEM, LINES DEPENDENT ON KIM MONITOR
* RESIDENT DATA ARE FLAGGED WITH THE WORD *KIM* IN
* THE COMMENT FIELD.
**************************************************************

**************************************************************
* SYMBOL DEFINITIONS
*
GETKEY  EQU     #$1F6A  ;*KIM* KEYPAD SUBROUTINE
SCAND   EQU     #$1F19  ;*KIM* DISPLAY SUBROUTINE
SCANDS  EQU     #$1F1F  ;*KIM* DISPLAY SUBROUTINE
PADD    EQU     #$1701  ;*KIM* PORT A DATA DIR. REG.
PA      EQU     #$1700  ;*KIM* PORT A DATA
SADD    EQU     #$1741  ;*KIM* PORT SA DATA DIR. REG.
SA      EQU     #$1740  ;*KIM* PORT SA DATA
PBDD    EQU     #$1703  ;*KIM* PORT B DATA DIR. REG.
PB      EQU     #$1702  ;*KIM* PORT B DATA
MNITOR  EQU     #$1C22  ;*KIM* RESET ENTRY TO MONITOR
PCCMD   EQU     #$1CDC  ;*KIM* PC SINGLE STEP ENTRY
REGS    EQU     A       ;8080 REG. BASE ADDR.
HL      EQU     L       ;HL REGISTER PAIR
PC      EQU     PCL     ;8080 PROGRAM COUNTER
INST    EQU     SCR     ;CURRENT 8080 INSTRUCTION
(HL)    EQU     HL-REGS ;INDEXES TO REGISTERS, ETC.
(DE)    EQU     DE-REGS
(SP)    EQU     SP-REGS
(PC)    EQU     PC-REGS
(DECIT) EQU     DECIT-REGS
(INCIT) EQU     INCIT-REGS
(PNT)   EQU     PNT-REGS
(SCR)   EQU     SCR-REGS
SMASK   EQU     #$80    ;ISOLATES SIGN BIT FROM PSW
PMASK   EQU     #$04    ;ISOLATES PARITY BIT
CMASK   EQU     #$01    ;ISOLATES CARRY BIT
ZMASK   EQU     #$40    ;ISOLATES ZERO BIT
SINC    EQU     #$01    ;FOR INR/DCR
MINSPD  EQU     #$00    ;SETS MINIMUM TRACE SPEED
HALT    EQU     #$76    ;8080 HALT OP-CODE
BKTBLN  EQU     ENDBK-BKTBL ;LENGTH OF BRKPNT TABLE
NOBRK   EQU     #$FFFF  ;DUMMY BREAKPOINT ENTRY
RSTHI   EQU     #$00    ;HIGH ORDER 8080 RST VECTOR
INPG2   EQU     #$02    ;SECOND PG OF INTERP. ROUTINES
INTDIS  EQU     #$01    ;DISABLES INTERRUPTS
```

```
                ********KIM-1 INTERRUPT VECTORS********

17FA                    ORG     #$17FA

17FA    8D00    NMI     DC 2,   START   ;*KIM* NON-MASKABLE INTERRUPT

17FC    FFFF    RESET   DC 2,   #$FFFF  ;*KIM* RESET VECTOR

                ********MINIMUM CONFIGURATION VECTORS********

*17FA   8017    NMI     DC 2,   MNSTRT  ;*KIM* NON-MASKABLE INTERRUPT

*17FC   FFFF    RESET   DC 2,   #$FFFF  ;*KIM* RESET

*17FE   221C    IRQ     DC 2,   MNITOR  ;*KIM* IRQ VECTOR

                ************************************************

0076                    ORG     #$0076

                ********8080 INTERRUPT LOGIC********

0076    AA      INT     TAX             ;SAVE HALT DATA
0077    A9 01           LDAIM   #$01    ;READ INTERRUPT PORT
0079    8D 0317         STA     PBDD    ;*KIM*
007C    A8              TAY             ;USE TO SET INTE
007D    AD 0217         LDA     PB      ;*KIM*
0080    25 E3           ANDZ    INTE    ;INTE=FF IF ENABLED
0082    10 07           BPL     NOINT   ;INTERRUPT?
0084    84 E3           STYZ    INTE    ;YES, DISABLE FURTHER INTERRUPTS
0086    09 C7           ORAIM   #$C7    ;FORM RST INSTRUCTION
0088    85 FE           STAZ    INST    ;SAVE NEW INSTRUCTION
008A    88              DEY             ;INDICATE INTERRUPT TAKEN
008B    A5 E3   NOINT   LDAZ    INTE
008D    D0 02           BNE     NINT
008F    C6 E3           DECZ    INTE    ;ENABLE INTERRUPT
0091    8A      NINT    TXA
0092    D0 01           BNE     NOHLT   ;WAS THIS A HALT?
0094    88              DEY             ;YES
0095    98      NOHLT   TYA             ;NORMAL RETURN?
0096    D0 05           BNE     NRMRET  ;YES, GO RETURN
0098    68              PLA             ;NO, CHANGE RETURN TO
0099    18              CLC
009A    69 03           ADCIM   #$03    ;POINT PAST JSR INCPC
009C    48              PHA
009D    60      NRMRET  RTS
```

```
                ********OPTIONAL EXECUTIVE LOOP********
                **********ENTER AT "START"***********

009E    A5 E2   TRACE   LDAZ    SPEED    ;GET SPEED VALUE
00A0    85 FE           STAZ    SCR

00A2    20 191F TRA     JSR     SCAND    ;*KIM* DISPLAY
00A5    C6 FE           DECZ    SCR      ;TIMEOUT?
00A7    D0 F9           BNE     TRA      ;NO, KEEP DISPLAYING

00A9    A2 02   BRK     LDXIM   BKTBLN   ;LOAD TABLE LENGTH
00AB    A5 EF   BKLP    LDAZ    PCL      ;GET LO PC BYTE
00AD    DD DE00         CMPX    BKTBL-2  ;DOES IT MATCH?
00B0    D0 07           BNE     NXBK     ;NO
00B2    A5 F0           LDAZ    PCH      ;GET HI PC BYTE
00B4    DD DF00         CMPX    BKTBL-1  ;DOES IT MATCH?
00B7    F0 24           BEQ     MONIT    ;YES! GO TO MONITOR
00B9    CA      NXBK    DEX              ;NO, TRY NEXT BREAK
00BA    CA              DEX
00BB    D0 EE           BNE     BKLP     ;MORE

00BD    A2 FF   START   LDXIM   #$FF     ;SET STACK POINTER
00BF    9A              TXS
00C0    20 8317         JSR     SIM80    ;DO ONE 8080 INSTR,

00C3    A9 00           LDAIM   INSET    ;SET FOR INPUT
00C5    8D 0317         STA     PBDD     ;*KIM*
00C8    AD 0217         LDA     PB       ;*KIM* READ CONTROL PORT
00CB    29 07           ANDIM   #$07     ;ONLY TEST CONTROL SWITCHES

00CD    4A              LSRA             ;FULL SPEED?
00CE    90 D9           BCC     BRK      ;YES, GO RUN
00D0    4A              LSRA             ;NO- REGISTER MODE?
00D1    D0 08           BNE     REG      ;YES, SKIP PC UPDATE

00D3    A5 EF           LDAZ    PCL      ;NO, MOVE PC TO POINT
00D5    85 FA           STAZ    POINTL   ;*KIM*
00D7    A5 F0           LDAZ    PCH
00D9    85 FB           STAZ    POINTH   ;*KIM*

00DB    90 C1   REG     BCC     TRACE    ;TRACE?
00DD    4C 221C MONIT   JMP     MNITOR   ;*KIM* NO, STEP
```

```
                *******BREAKPOINT ADDRESS TABLE*******

00E0    FFFF    BKTBL   DC 2,   NOBRK   ;BREAKPOINT TABLE
00E2            ENDBRK                  ;NEXT ADDRESS AFTER TABLE

00E2    00      SPEED   DC      MINSPD  ;SETS TRACE SPEED

                ****************************************

 *THE FOLLOWING ITEMS ON THIS PAGE MUST BE LOCATED IN PAGE 0 RAM MEMORY*
       *THEY MUST REMAIN IN ORDER AND BEGIN ON AN ODD BOUNDRY*

                *******8080 REGISTERS*******

00E3    01      INTE    DC      INTDIS  ;INTERRUPT ENABLE FLAG
00E4    02      PSW     DC      #$02    ;8080 PROCESSOR STATUS
00E5    00      A       DC      #$00    ;8080 ACCUMULATOR
00E6    00      M       DC      #$00    ;DUMMY MEMORY REGISTER
00E7    00      L       DC      #$00    ;HL REGISTER PAIR
00E8    00      H       DC      #$00
00E9    00      E       DC      #$00    ;DE REGISTER PAIR
00EA    00      D       DC      #$00
00EB    00      C       DC      #$00    ;BC REGISTER PAIR
00EC    00      B       DC      #$00
00ED    74      SPL     DC      #$74    ;8080 STACK POINTER
00EE    00      SPH     DC      #$00
00EF    00      PCL     DC      #$00    ;8080 PROGRAM COUNTER
00F0    00      PCH     DC      #$00

                *******CONSTANTS AND DATA AREA*******
        *NOTE:  #$00F1 MUST BE SET TO #$00 WHEN READING OR WRITING*
        *TAPES AND MANUALLY SET TO #$FF BEFORE SIMULATOR OPERATION*

00F1    FFFF    DECIT   DC 2,   #$FFFF  ;DOUBLE PRECISION -1
00F3    0100    INCIT   DC 2,   #$0001  ;DOUBLE PRECISION 1
00F5    00      FLAG    DC      #$00    ;USED BY SIMULATOR
00F6    00      DESTDA  DC      #$00    ;DESTINATION DATA
00F7    00      DEST    DC      #$00    ;DESTINATION INDEX
00F8    00      SRC     DC      #$00    ;SOURCE INDEX
00F9    00              DC      #$00    ;*KIM* DISPLAY BUFFER
00FA    FA      POINTL  DC      #$FA    ;*KIM* DISPLAY POINTER
00FB    00      POINTH  DC      #$00    ;*KIM*
00FC    0100    PNT     DC      #$0001  ;DATA POINTER
00FE    0001    SCR     DC      #$0100  ;SCRATCHPAD REGISTER
```

```
1780                        ORG     #$1780

                  *******MAIN CONTROL LOOP*******

1780    A2 FF    MNSTRT   LDXIM   #$FF      ;SET STACK POINTER
1782    9A                TXS               ;FOR MINUMUM CONFIGURATION

1783    20 9117  SIM80    JSR     MAIN      ;EXECUTE ONE 8080 OP-CODE

1786    A5 E4             LDAZ    PSW       ;GET 8080 STATUS
1788    29 D7             ANDIM   #$D7      ;CLEAR PRECLEAR BITS
178A    09 02             ORAIM   #$02      ;SET PRESET BITS
178C    85 E4             STAZ    PSW       ;SAVE IT
178E    58        EXIT    CLI               ;ALLOW INTERRUPTS
178F    60                RTS               ;RETURN TO CALLER
1790    EA                NOP

        *******VARIATIONS FOR MINIMUM CONFIGURATION*******

*178E   4C 221C           JMP     MNITOR    ;*KIM* REGISTER SINGLE STEP
*178E   4C DC1C           JMP     PCCMD     ;*KIM* PC SINGLE STEP

*178E   58                CLI               ;FULL SPEED RUN
*178F   D0 EF             BNE     MNSTRT

        ************************************************

1791    78        MAIN    SEI               ;DON'T ALLOW INTERRUPTS
1792    D8                CLD               ;CLEAR DECIMAL MODE

1793    A2 00             LDXIM   #$00      ;LOAD INDEX

1795    A1 E7             LDAIX   HL        ;FETCH MEMORY
1797    85 E6             STAZ    M

1799    A1 EF             LDAIX   PCL       ;FETCH INSTRUCTION

179B    85 FE             STAZ    INST      ;SAVE IT
179D    49 76             EORIM   HALT      ;IS THIS A HALT?
179F    F0 04             BEQ     ITR       ;YES, DON'T UPDATE PC
17A1    EA                NOP

        *******VARIATION FOR INTERRUPT ACTION*******

*179F   20 7600           JSR     INT       ;GO CHECK FOR INTERRUPT

        ************************************************

17A2    20 8303           JSR     INCPC     ;NO, INCREMENT PC

17A5    A2 02    ITR      LDXIM   #$02      ;SET COUNTER/INDEX
17A7    A5 FE             LDAZ    INST      ;RECOVER INSTRUCTION
```

```
17A9    48        DIV     PHA             ;TEMP SAVE A
17AA    29 07             ANDIM   #$07    ;MASK FOR LS3 BITS
17AC    49 07             EORIM   #$07    ;REVERSE 'EM
17AE    95 F6             STAZX   DEST-1  ;SAVE 'EM
17B0    68                PLA             ;RECOVER TEMP
17B1    4A                LSRA            ;SHIFT TO NEXT FIELD
17B2    4A                LSRA
17B3    4A                LSRA
17B4    CA                DEX             ;DEC COUNTER/INDEX
17B5    D0 F2             BNE     DIV     ;LOOP TILL DONE

17B7    86 F5             STXZ    FLAG    ;CLEAR FLAG

17B9    A0 02             LDYIM   INPG2   ;SET FOR 2ND INTERP PAGE
17BB    4A                LSRA            ;TEST FOR QUADRANT
17BC    B0 08             BCS     IMI
17BE    F0 0E             BEQ     1ST

17C0    A9 11     IMARL   LDAIM   #$11    ;NO, ARILOG
17C2    65 F7             ADCZ    DEST
17C4    AA                TAX
17C5    A9 00             LDAIM   (A)     ;SET DEST = A
17C7    85 F7             STAZ    DEST
17C9    F0 07     IMI     BEQ     MOVE

17CB    88        2ND     DEY             ;CHANGE PAGES
17CC    A9 07             LDAIM   #$07    ;OFFSET FOR 2ND

17CE    65 F8     1ST     ADCZ    SRC
17D0    AA                TAX             ;XFER TO INDEX
17D1    E8                INX

17D2    98        MOVE    TYA             ;SET HIGH ADDRESS
17D3    48                PHA
17D4    BD E303           LDAX    XFRTBL  ;GET LO ADDR
17D7    48                PHA             ;SET LO ADDRESS

17D8    A6 F8             LDXZ    SRC     ;GET SOURCE INDEX
17DA    B4 E5             LDYZX   REGS    ;GET SOURCE DATA IN Y

17DC    A6 F7             LDXZ    DEST    ;GET DEST INDEX
17DE    B5 E5             LDAZX   REGS
17E0    85 F6             STAZ    DSTDA   ;SAVE DESTINATION DATA

17E2    A9 44             LDAIM   #$44    ;SET 6502 STATUS
17E4    48                PHA
17E5    8A                TXA             ;DEST IND, IN A & X
17E6    40                RTI             ;GO INTERPRET
```

```
0100                    ORG     #$0100

0100    4A      IN      LSRA            ;DETERMINE TYPE
0101    F0 06           BEQ     XCHG
0103    90 33           BCC     INPUT
0105    A8              TAY             ;SET Y=01
0106    4C 7A03         JMP     CALL65  ;CALL 6502 SUBROUTINE

0109    20 6303 XCHG    JSR     RP/SCR  ;MOVE HL TO SCRATCH
010C    A2 04           LDXIM   (DE)    ;POINT TO DE
010E    A0 02           LDYIM   (HL)    ;AND HL
0110    20 6503         JSR     RP/RP   ;MOVE DE TO HL
0113    A2 04           LDXIM   (DE)    ;POINT TO DE
0115    4C 7003         JMP     SCR/RP  ;MOVE SCRATCH TO DE

0118    20 6303 XTHL    JSR     RP/SCR  ;MOVE HL TO SCRATCH
011B    20 7603         JSR     SP/PNT  ;MAKE SP POINTER
011E    A2 02           LDXIM   (HL)    ;POINT TO HL
0120    20 A003         JSR     MEM/RP  ;MOVE STACK TOP TO HL
0123    A2 19           LDXIM   (SCR)   ;POINT TO SCRATCH
0125    D0 56           BNE     RP/MEM  ;MOVE SCRATCH TO STACK

                *******I/O, ETC, ENTRY*******

0127    4A      I/O     LSRA            ;WHICH INST?
0128    D0 03           BNE     LBL
012A    86 E3           STXZ    INTE    ;SET/CLEAR INTE
012C    60              RTS

012D    A2 02   LBL     LDXIM   (HL)    ;POINT TO HL
012F    90 CF           BCC     IN      ;IS IT INPUT?
0131    4A              LSRA            ;NO,IS IT XTHL?
0132    F0 E4           BEQ     XTHL    ;YES
0134    B0 58           BCS     JUMPUN  ;IS IT JUMP?
0136    C6 F5   OUTPUT  DECZ    FLAG    ;NO, FLAG AN OUTPUT
0138    20 AB03 INPUT   JSR     IMM     ;GET PORT #
013B    A5 FE           LDAZ    SCR
013D    0A              ASLA            ;DOUBLE INDEX
013E    AA              TAX
013F    BD FC03 IOPLP   LDAX    IOTBL   ;GET I/O ADDRESS
0142    99 F800         STAY    PNT-1   ;STORE IN POINTER
0145    E8              INX             ;BUMP INDEX
0146    88              DEY             ;DECREMENT COUNTER
0147    D0 F6           BNE     IOPLP
0149    C8              INY             ;POINT TO DATA DIR, REG,
014A    A5 F5           LDAZ    FLAG    ;GET INPUT OR OUTPUT DATA
014C    91 FC           STAIY   PNT     ;SET DATA DIR, REGISTER
014E    88              DEY             ;POINT TO PORT
014F    4A              LSRA            ;INPUT?
0150    90 06           BCC     IOIN    ;YES, GO DO IT
0152    A5 E8           LDAZ    A       ;NO, OUTPUT A
0154    91 FC           STAIY   PNT
0156    B0 04           BCS     IOEX
0158    B1 FC   IOIN    LDAIY   PNT     ;INPUT FROM PORT
015A    85 E5           STAZ    A       ;TO A
015C    60      IOEX    RTS
```

```
                ********ARILOG IMMEDIATE ENTRY********

015D    20 A803 ARIM    JSR     IMM     ;GET IMMEDIATE BYTE
0160    18              CLC
0161    4C C017         JMP     IMARL   ;GO DO ARITH ROUTINE

                ********PUSH ENTRY********

0164    4A      PUSH    LSRA            ;IS THIS A PUSH?
0165    B0 07           BCS     SKP     ;YES, GO PUSH
0167    C9 03           CMPIM   #$03    ;NO, IS THIS A CALL?
0169    F0 27           BEQ     CALLUN  ;YES, GO CALL
016B    4C 1102         JMP     UNDEF   ;NO, UNDEFINED

016E    0A      SKP     ASLA            ;MAKE AN INDEX
016F    AA              TAX             ;IS IT PUSH PSW?
0170    D0 01           BNE     PUSHT   ;NO, KEEP INDEX
0172    CA              DEX             ;YES, ADJUST INDEX
0173    8A      PUSHT   TXA             ;TEMP SAVE
0174    48              PHA
0175    20 9703         JSR     DECSP   ;DECREMENT STACK POINTER
0178    20 7603         JSR     SP/PNT  ;IT BECOMES POINTER
017B    68              PLA             ;RECOVER TEMP SAVE
017C    AA              TAX
017D    A0 01   RP/MEM  LDYIM   #$01    ;CLEAR INDEX
017F    B5 E6   RPLP    LDAZX   REGS+1  ;GET NEXT RP DATA
0181    91 FC           STAIY   PNT     ;STORE IN MEMORY
0183    CA              DEX
0184    88              DEY
0185    F0 F8           BEQ     RPLP
0187    60      RTS     RTS

                ********JUMP ENTRY********

0188    B8      JMP     CLV             ;INDICATE A JUMP

                ********CALL ENTRY********

0189    20 0002 CALL    JSR     CONDIT  ;TEST CONDITION
018C    F0 04           BEQ     CALLUN  ;MET
018E    4C 8003         JMP     DBLINC  ;NOT MET

0191    B8      JUMPUN  CLV             ;INDICATE A JUMP
0192    20 5D03 CALLUN  JSR     PC/PNT  ;GET NEXT 2 BYTES
0195    A2 19           LDXIM   (SCR)   ;INTO SCRATCH
0197    20 A003         JSR     MEM/RP
019A    50 08           BVC     JM      ;JUMP
019C    20 8003         JSR     DBLINC  ;BUMP PC

019F    A2 0A   SVR     LDXIM   (PC)    ;SAVE RETURN
01A1    20 7301         JSR     PUSHT   ;ON STACK

01A4    A2 0A   JM      LDXIM   (PC)    ;POINT TO PC
01A6    4C 7003         JMP     SCR/RP  ;GIVE IT NEW ADDRESS
```

```
                ********RETURN ENTRY********

01A9    20 0002 RETURN  JSR     CONDIT  ;TEST CONDITION
01AC    D0 D9           BNE     RTS     ;NOT MET
01AE    A2 0A   RETUN   LDXIM   (PC)     ;POP RETURN OFF
01B0    D0 14           BNE     POPIT   ;STACK INTO PC

                ********RESET ENTRY********

01B2    A5 FE   RST     LDAZ    INST    ;CONVERT INSTRUCTION
01B4    29 38           ANDIM   #$38    ;TO RESET VECTOR
01B6    85 FE           STAZ    SCR     ;STORE IT IN SCRATCH
01B8    A9 00           LDAIM   RSTHI   ;HIGH RESET VECTOR
01BA    85 FF           STAZ    SCR+1
01BC    90 E1           BCC     SVR

                ********POP ENTRY********

01BE    4A      POP     LSRA            ;POP?
01BF    90 17           BCC     OTH     ;NO
01C1    0A              ASLA
01C2    AA              TAX             ;POP PSW?
01C3    D0 01           BNE     POPIT   ;NO, KEEP INDEX
01C5    CA              DEX             ;YES, ADJUST INDEX
01C6    8A      POPIT   TXA             ;TEMP SAVE INDEX
01C7    48              PHA
01C8    20 7603         JSR     SP/PNT  ;SP IS POINTER
01CB    68              PLA             ;RECOVER TEMP
01CC    AA              TAX
01CD    20 A003         JSR     MEM/RP  ;PULL OUT STACK DATA

01D0    20 D301 INCSP   JSR     ISP     ;BUMP SP
01D3    A2 08   ISP     LDXIM   (SP)
01D5    4C 8503         JMP     INC

01D8    A2 02   OTH     LDXIM   (HL)    ;POINT TO HL
01DA    A0 08           LDYIM   (SP)    ;ASSUME SPHL
01DC    0A              ASLA            ;IS IT SPHL?
01DD    F0 08           BEQ     SPHL    ;YES, GO DO IT
01DF    C9 04           CMPIM   #$04    ;NO, WHAT IS IT?
01E1    F0 2E           BEQ     UNDEF   ;UNDEFINED
01E3    10 C9           BPL     RETUN   ;RETURN
01E5    A0 0A           LDYIM   (PC)    ;PCHL
01E7    4C 6503 SPHL    JMP     RP/RP

                ********TABLE OF PSW MASKS********

01EA    80      MSKTBL  DC      SMASK   ;SIGN BIT MASK
01EB    04      FOUR    DC      PMASK   ;PARITY BIT MASK
01EC    01              DC      CMASK   ;CARRY BIT MASK
01ED    40              DC      ZMASK   ;ZERO BIT MASK

                **********************************

*PAGE 1 RAM MEMORY ADDRESSES #$01F0 TO #$01FF MUST BE RESERVED FOR STACK*
```

```
0200                        ORG     #$0200

                    *******TEST SELECTED PSW BIT*******

0200    4A          CONDIT  LSRA              ;MAKE INDEX TO MASK TABLE
0201    AA                  TAX
0202    BD EA01             LDAX    MSKTBL    ;GET CONDITION FLAG MASK
0205    25 E4               ANDZ    PSW       ;IS CONDITION SET?

0207    B0 03               BCS     RTN       ;LEAVE Z FLAG SET
0209    5D EA01             EORX    MSKTBL    ;OR REVERSE IT
020C    60          RTN     RTS

                    *******NOP ENTRY*******

020D    A5 FE       NOP     LDAZ    INST      ;CHECK INSTRUCTION
020F    F0 FB               BEQ     RTN       ;IF NOP, JUST RETURN
0211    4C 221C     UNDEF   JMP     MNITOR    ;*KIM* GO TO SYSTEM MONITOR

                    *******DAD/LXI ENTRY*******

0214    4A          DADLXI  LSRA              ;DROP LSB
0215    0A                  ASLA
0216    D0 02               BNE     NSP       ;USE SP?
0218    A9 00               LDAIM   (SP)      ;YES, POINT TO IT
021A    48          NSP     PHA               ;SAVE POINTER
021B    8A                  TXA               ;GET DESTINATION INDEX
021C    4A                  LSRA              ;DAD?
021D    B0 12               BCS     LXI       ;NO, GO DO LXI

021F    68          DAD     PLA               ;YES, RECOVER POINTER
0220    A8                  TAY
0221    A2 02               LDXIM   (HL)      ;SET HL AS DESTINATION
0223    20 8703             JSR     INCDEC    ;GO ADD RP & HL

                    *******SET 8080 CARRY PER 6502 CARRY*******

0226    A8          CARRY   TAY               ;SAVE RESULT
0227    2A                  ROLA              ;BRING IN CARRY
0228    45 F5               EORZ    FLAG      ;CHG TO BORROW IF SUBTRACT
022A    46 E4               LSRZ    PSW
022C    4A                  LSRA              ;RESTORE CARRY(BORROW)
022D    26 E4               ROLZ    PSW
022F    98                  TYA               ;RECOVER RESULT
0230    60                  RTS

0231    20 5D03     LXI     JSR     PC/PNT    ;PC INTO PNT
0234    68                  PLA               ;RECOVER POINTER
0235    AA                  TAX               ;USE TO INDEX DESTINATION RP
0236    20 A003             JSR     MEM/RP    ;MOVE IMMEDIATE DATA TO RP
0239    4C 8003             JMP     DBLINC    ;BUMP PROGRAM COUNTER
```

```
                ********LOAD & STORE ENTRY********

023C    2C EB01 LDSTR   BIT     FOUR    ;INDIRECT?
023F    F0 08           BEQ     DIRECT  ;NO

0241    4A              LSRA            ;STAX?
0242    90 23           BCC     LDAX    ;NO

0244    A5 E5   STAX    LDAZ    A       ;YES,STORE A
0246    81 E4           STAIX   REGS+1
0248    60              RTS

0249    48      DIRECT  PHA             ;TEMP SAVE
024A    20 5D03         JSR     PC/PNT  ;PC TO POINTER
024D    20 8003         JSR     DBLINC
0250    A2 19           LDXIM   (SCR)
0252    20 A003         JSR     MEM/RP
0255    A2 17           LDXIM   (PNT)
0257    20 7003         JSR     SCR/RP
025A    68              PLA             ;RECOVER TEMP

025B    4A              LSRA            ;LOAD?
025C    B0 0E           BCS     STORE   ;NO, STORE

025E    0A      LOAD    ASLA            ;MAKE INDEX
025F    AA              TAX
0260    F0 03           BEQ     LDAD    ;LOAD A DIRECT?
0262    4C A003         JMP     MEM/RP  ;NO, LOAD HL DIRECT

0265    A2 17   LDAD    LDXIM   (PNT)   ;POINT TO POINTER

0267    A1 E5   LDAX    LDAIX   REGS    ;LOAD A
0269    85 E5           STAZ    A
026B    60              RTS

026C    0A      STORE   ASLA            ;MAKE INDEX
026D    AA              TAX
026E    F0 03           BEQ     STAD    ;STORE A DIRECT?
0270    4C 7D01         JMP     RP/MEM  ;NO, STORE HL DIRECT

0273    A2 18   STAD    LDXIM   (PNT)+1 ;POINT TO POINTER
0275    D0 CD           BNE     STAX
```

```
                ********INX/DCX ENTRY********

0277    A0 0C   INXDCX  LDYIM   (DECIT) ;SET FOR DECREMENT

0279    4A              LSRA
027A    90 02           BCC     DRP     ;YES

027C    A0 0F           LDYIM   (INCIT) ;NO, SET FOR INCR

027E    0A      DRP     ASLA            ;DROP LSB
027F    D0 02           BNE     NOR     ;FOR SP?

0281    A9 08           LDAIM   (SP)    ;YES, POINT TO SP

0283    AA      NOR     TAX             ;USE AS INDEX
0284    4C 8703         JMP     INCDFC

                ********DCR ENTRY********

0287    38      DCR     SEC             ;SET FOR DECREMENT
0288    C6 F5           DECZ    FLAG

                ********INR ENTRY********

028A    A0 01   INR     LDYIM   SINC    ;SET FOR INCREMENT

028C    20 BD03 BTH     JSR     ADDR    ;GO ADD
028F    4C 3803         JMP     STATUS  ;SET STATUS, NOT CARRY

                ********MOVE IMMEDIATE ENTRY********

0292    20 AB03 MVI     JSR     IMM     ;GET IMMEDIATE BYTE
0295    4C D217         JMP     MOVE    ;MOVE IT TO DESTINATION

                ********MOVE ENTRY********

0298    98      MOVIT   TYA             ;GET SOURCE DATA
0299    CA      MVIT1   DEX             ;IS DEST MEMORY?
029A    D0 02           BNE     NMEM    ;NO

029C    81 E7           STAIX   HL      ;YES, STORE IN MEMORY

029E    95 E6   NMEM    STAZX   REGS+1  ;STORE IN DEST REGISTER
02A0    60              RTS
```

```
                **ARITHMETIC ROUTINES**
                ********ADC ENTRY********

02A1    88      ADC     CLV             ;FLAG W/CARRY
02A2    50 08           BVC     ADD

                ********CMP ENTRY********

02A4    A2 19   CMP     LDXIM   (SCR)   ;CHG DEST TO SCR
02A6    D0 01           BNE     SUB     ;SO ONLY PSW IS AFFECTED

                ********SBB ENTRY********

02A8    B8      SBB     CLV             ;FLAG W/BORROW

                ********SUB ENTRY********

02A9    C6 F5   SUB     DECZ    FLAG    ;FLAG = FF FOR SUBTRACT
02AB    38              SEC

                ********ADD ENTRY********

02AC    4C 3203 ADD     JMP     AD/CRY

                ***LOGICAL OPERATIONS**
                ********ORA ENTRY********

02AF    98      ORA     TYA             ;GET SOURCE DATA
02B0    05 E5           ORAZ    A       ;LOGICAL OR
02B2    90 03           BCC     SOM     ;SET FOR A CLEAR AUX CRY

                ********XRA ENTRY********

02B4    98      XRA     TYA             ;GET SOURCE DATA
02B5    45 E5           EORZ    A       ;LOGICAL EXCLUSIVE OR
02B7    A0 00   SOM     LDYIM   #$00    ;00 WON'T SET ANY FLAGS
02B9    90 0B           BCC     ALL     ;GO SAVE RESULT

                ********ANA ENTRY********

02BB    98      ANA     TYA             ;GET SOURCE DATA
02BC    48              PHA             ;SAVE IT
02BD    05 E5           ORAZ    A       ;LOGICAL OR OF BIT
02BF    0A              ASLA            ;THREE BECOMES AUX CARRY
02C0    29 10           ANDIM   #$10
02C2    A8              TAY
02C3    68              PLA             ;GET SOURCE DATA
02C4    25 E5           ANDZ    A       ;LOGICAL AND

02C6    84 FF   ALL     STYZ    SCR+1   ;SAVE FLAG SETTING DATA
02C8    85 E5           STAZ    A       ;LOGICAL RESULT INTO A
02CA    20 3803         JSR     STATUS  ;SET STATUS
02CD    A9 EE           LDAIM   #$EE    ;SELECT STATUS BITS TO CLEAR
02CF    25 E4   CY&AC   ANDZ    PSW     ;CLEAR SELECTED BITS
02D1    05 FF           ORAZ    SCR+1   ;SET SELECTED BITS
02D3    85 E4   STPSW   STAZ    PSW     ;SAVE NEW PSW
02D5    60              RTS
```

```
                *******ROTATE, ETC. ENTRY*******

02D6    A4 E5   ROTATE  LDYZ    A         ;GET ACCUMULATOR
02D8    4A              LSRA              ;IS THIS CMC OR STC?
02D9    D0 0A           BNE     ROCOMP    ;NO, IT IS A ROTATE OR CMA

02DB    A5 E4           LDAZ    PSW       ;YES, CHANGE CARRY
02DD    49 01           FORIM   #$01      ;CMC
02DF    90 02           BCC     STPS
02E1    09 01           ORAIM   #$01      ;STC
02E3    70 EE   STPS    BVS     STPSW

02E5    B0 13   ROCOMP  BCS     LEFT      ;LEFT OR RIGHT?
02E7    4A              LSRA              ;CMA?
02E8    D0 06           BNE     ROT       ;NO, ROTATE
02EA    98              TYA               ;YES, COMMPLEMENT A
02EB    49 FF           FORIM   #$FF
02ED    85 E5           STAZ    A         ;DOESN'T SET STATUS
02EF    60              RTS

02F0    98      ROT     TYA               ;GET ACCUMULATOR
02F1    B0 02           BCS     RRC
02F3    A5 E4           LDAZ    PSW       ;GET 8080 CARRY
02F5    4A      RRC     LSRA
02F6    98              TYA               ;GET ACCUMULATOR
02F7    6A              RORA              ;ROTATE RIGHT
02F8    70 0D           BVS     JCRY      ;GO STORE&SET CARRY

02FA    4A      LEFT    LSRA              ;DECIMAL ADJUST?
02FB    F0 0F           BEQ     DAA       ;YES, GO  DO IT
02FC    98              TYA               ;NO, GET ACCUMULATOR
02FE    B0 04           BCS     RLC       ;RLC?
0300    A5 E4           LDAZ    PSW       ;NO, GET 8080 CARRY
0302    6A              RORA              ;MOVE IT INTO MSNIB
0303    6A              RORA
0304    0A      RLC     ASLA              ;MOVE IT INTO CARRY
0305    98              TYA               ;GET ACCUMULATOR
0306    2A              ROLA              ;MOVE IT LEFT
0307    85 E5   JCRY    STAZ    A         ;SAVE IT
0309    4C 2602         JMP     CARRY     ;GO SET STATUS

030C    18      DAA     CLC
030D    08              PHP               ;PRESERVE STATUS
030E    98              TYA               ;GET SOURCE DATA
030F    EA              NOP
0310    85 F6           STAZ    DESTDA    ;PREP FOR ADD
0312    29 0F           ANDIM   #$0F      ;LOOK AT LSNIB
0314    69 06           ADCIM   #$06      ;IF >=0A WILL CAUSE AUX  CRY
0316    05 E4           ORAZ    PSW       ;OR IF AC IS ALREADY SET
0318    29 10           ANDIM   #$10      ;EITHER SET?
031A    F0 02           BEQ     NOSIX     ;NO, DON'T ADJUST LSNIB
031C    A9 06           LDAIM   #$06      ;YES, ADJUST IT
031E    A8      NOSIX   TAY
```

```
031F    65 F6           ADCZ    DESTDA  ;GET SOURCE
0321    B0 08           BCS     SXTY
0323    69 60           ADCIM   #$60     ;IS MSNIB NOW >= A0?
                                        ;IF SO, CARRY IS SET
0325    2A              ROLA            ;GET CARRY
0326    05 E4           ORAZ    PSW     ;OR WITH 8080 CARRY
0328    4A              LSRA            ;IS EITHER SET?
0329    90 04           BCC     DA      ;NO, DON'T ADJUST MSNIB
032B    98      SXTY    TYA             ;YES, ADJUST MSNIB
032C    69 5F           ADCIM   #$5F    ;(5F + CARRY = 60)
032E    A8              TAY
032F    A2 00   DA      LDXIM   #$00    ;DESTINATION IS A
0331    28              PLP             ;RESTORE STATUS
0332    20 BD03 AD/CRY  JSR     ADDR
0335    20 2602         JSR     CARRY

                *******STATUS ROUTINE*******
        *SETS ZERO, SIGN AND PARITY, LEAVES CARRY AND AC*

0338    A8      STATUS  TAY             ;SAVE RESULT
0339    26 E4           ROLZ    PSW     ;CLEAR PSW SIGN BIT
033B    0A              ASLA            ;PUT NEW SIGN IN CARRY
033C    85 F5           STAZ    FLAG    ;CLEAR LSB OF FLAG
033E    A5 E4           LDAZ    PSW     ;PUT NEW SIGN IN PSW
0340    6A              RORA
0341    09 46   SGN     ORAIM   #$46    ;PRESET Z,P & PRESET

0343    AA              TAX             ;SAVE IN X
0344    98              TYA             ;RECOVER WORD
0345    F0 13           BEQ     DONE    ;IF ZERO, ALL DONE

0347    E6 F5   FLIP    INCZ    FLAG    ;FLIP FLAG
0349    4A      PAR     LSRA            ;TEST EACH BIT
034A    F0 02           BEQ     ALL     ;NO MORE BITS
034C    90 FB           BCC     PAR
034E    B0 F7   ALL     BCS     FLIP

0350    46 F5           LSRZ    FLAG    ;TEST FLAG
0352    8A              TXA             ;RECOVER PSW
0353    29 BF           ANDIM   #$BF    ;CLEAR Z
0355    B0 02           BCS     REC     ;PARITY EVEN?
0357    29 FB           ANDIM   #$FB    ;NO, CLEAR P
0359    AA      REC     TAX             ;BACK TO X

035A    86 E4   DONE    STXZ    PSW     ;STORE AS PSW
035C    60              RTS
```

```
                ********SUBROUTINES********
                ***MOVE REGISTER PAIRS***

035D    A2 0A   PC/PNT  LDXIM   (PC)     ;SOURCE IS PC

035F    A0 17   RP/PNT  LDYIM   (PNT)    ;DESTINATION IS PNT
0361    D0 02           BNE     RP/RP    ;GO TRANSFER

0363    A0 19   RP/SCR  LDYIM   (SCR)    ;DESTINATION IS SCR

0365    B5 E5   RP/RP   LDAZX   REGS     ;GET LOW ORDER SOURCE
0367    99 E500         STAY    REGS     ;STORE IN LOW DESTINATION
036A    B5 E6           LDAZX   REGS+1   ;GET HIGH SOURCE
036C    99 E600         STAY    REGS+1   ;STORE IN HI DESTINATION
036F    60              RTS

0370    8A      SCR/RP  TXA              ;RP IS DESTINATION
0371    A8              TAY
0372    A2 19           LDXIM   (SCR)    ;SCR IS SOURCE
0374    D0 EF           BNE     RP/RP

0376    A2 08   SP/PNT  LDXIM   (SP)     ;SP IS SOURCE
0378    D0 E5           BNE     RP/PNT   ;UNCONDITIONAL BRANCH

                ********CALL 6502 SUBROUTINE********
                ***(CALL SUBROUTINE ADDRESS -1)***

037A    B1 EF   CALL65  LDAIY   PC       ;GET ADDRESS
037C    48              PHA
037D    88              DEY              ;POINT TO LO CALL
037E    10 FA           BPL     CALL65   ;DONE?

                ********INC, DEC, OR ADD REGISTER PAIRS********

0380    20 8303 DBLINC  JSR     INCPC    ;INC PC TWICE
0383    A2 0A   INCPC   LDXIM   (PC)     ;INC PC

0385    A0 0E   INC     LDYIM   (INCIT)  ;POINT TO INC DATA

0387    18      INCDEC  CLC
0388    B5 E5   INDE    LDAZX   REGS     ;GET
038A    79 E500         ADCY    REGS     ;ADD
038D    95 E5           STAZX   REGS     ;SAVE
038F    C8              INY              ;BUMP INDEXES
0390    E8              INX
0391    98              TYA              ;CHECK INDEX
0392    29 01           ANDIM   #$01     ;LAST PASS?
0394    D0 F2           BNE     INDE     ;NO
0396    60              RTS              ;YES

0397    20 9A03 DECSP   JSR     DSP      ;DEC SP TWICE

039A    A2 08   DSP     LDXIM   (SP)     ;DEC SP

039C    A0 0C   DEC     LDYIM   (DECIT)  ;POINT TO DEC DATA
039E    D0 E7           BNE     INCDEC   ;GO DO IT
```

```
                        *******SUBROUTINES*******
                        *MOVE MEMORY TO REG PAIR*

03A0    A0 01   MEM=RP  LDYIM   #$01     ;MOVE DATA IN MEMORY

03A2    B1 FC   MRLP    LDAIY   PNT      ;GET NEXT BYTE
03A4    95 E6           STAZX   REGS+1   ;STORE IN DESTINATION
03A6    CA              DEX              ;BUMP INDICES
03A7    88              DEY
03A8    F0 F8           BEQ     MRLP
03AA    60              RTS

                        *******FETCH IMMEDIATE BYTE*******

03AB    A2 00   IMM     LDXIM   #$00     ;GET BYTE POINTED AT
03AD    A1 EF           LDAIX   PCL      ;BY PROGRAM COUNTER
03AF    85 FE           STAZ    SCR      ;SAVE IT IN SCR

03B1    A9 19           LDAIM   (SCR)    ;MAKE SCR THE SOURCE INDEX
03B3    85 F8           STAZ    SRC
03B5    20 8303         JSR     INCPC    ;BUMP PC
03B8    A2 00           LDXIM   #$00
03BA    A0 02           LDYIM   INPG2    ;POINT TO INTERP PAGE 2
03BC    60              RTS

                        *******ADD CONTENTS OF Y & DESTDA*******

03BD    98      ADDR    TYA              ;GET SOURCE DATA
03BE    45 F5           EORZ    FLAG     ;CHANGE IF SUBTRACT
03C0    A8              TAY
03C1    29 0F           ANDIM   #$0F     ;SAVE LSNIB
03C3    85 FE           STAZ    SCR
03C5    2A              ROLA             ;GET CARRY IN A
03C6    70 02           BVS     W/OCRY   ;ADD/SUB W/CARRY?

03C8    45 E4           EORZ    PSW      ;YES, CHG PER 8080 CARRY

03CA    48      W/OCRY  PHA              ;SAVE FOR LATER
03CB    4A              LSRA             ;SET CARRY

03CC    A5 F6           LDAZ    DESTDA   ;GET ACCUM DATA
03CE    29 0F           ANDIM   #$0F     ;LOOK AT LSNIB
03D0    65 FE           ADCZ    SCR      ;ADD TO SOURCE LSNIB
03D2    29 10           ANDIM   #$10     ;WAS THERE AN AC?

03D4    85 FF           STAZ    SCR+1    ;SAVE AC
03D6    A9 EF           LDAIM   #$EF     ;CLEAR AC
03D8    20 CF02         JSR     CY&AC

03DB    68              PLA              ;GET CARRY BACK
03DC    4A              LSRA
03DD    98              TYA              ;GET SOURCE DATA
03DE    65 F6           ADCZ    DESTDA   ;ADD TO ACCUMULATOR
03E0    4C 9902         JMP     MVIT1    ;GO PUT RESULT IN DESTINATION
```

```
                ********ENTRY ADDRESS TABLE FOR TRANSFER********
                 ********TO INSTRUCTION INTERPRETATION********

03E3    98      XFRTBL  DC      MOVIT   ;ENTRIES ON 2ND INTRP, PAGE
03E4    D8              DC      ROTATE
03E5    92              DC      MVI
03E6    87              DC      DCR
03E7    8A              DC      INR
03E8    77              DC      INXDCX
03E9    3C              DC      LDSTR
03EA    14              DC      DADLXI
03EB    0D              DC      NOP

03EC    B2              DC      RST     ;ENTRIES ON 1ST INTRP, PAGE
03ED    5D              DC      ARIM
03EE    64              DC      PUSH
03EF    89              DC      CALL
03F0    27              DC      I/O
03F1    88              DC      JMP
03F2    BE              DC      POP
03F3    A9              DC      RETURN

03F4    A4              DC      CMP     ;ENTRIES ON 2ND INTRP, PAGE
03F5    AF              DC      ORA
03F6    B4              DC      XRA
03F7    BB              DC      ANA
03F8    A8              DC      SBB
03F9    A9              DC      SUB
03FA    A1              DC      ADC
03FB    AC              DC      ADD

                ********INPUT/OUTPUT PORT ADDRESS********
                    ********DEFINITION TABLE********
*ENTRIES MUST BE IN NORMAL ORDER, IE, MOST SIGNIFIGANT ADDRESS BYTE FIRST*

03FC    1700    IOTBL   DC 2,   #$0017  ;8080 PORT 0 - *KIM* PORT A
03FE    1740            DC 2,   #$4017  ;8080 PORT 1 - *KIM* PORT 5A
```

```
                *******8080 DEMONSTRATION PROGRAM*******
                    *******TIME-OF-DAY CLOCK*******

0000                        ORG     #$0000

0000    31 7400 DEMO        LXI SP  #$0074  ;SET STACK POINTER
0003    F3                  DI              ;DISABLE INTERRUPTS
0004    00                  NOP             ;NO OPERATION
0005    C3 0900             JMP     TIME

0008    F1          INTIME  POP     PSW     ;POP UNUSED RETURN OFF STACK

0009    B7          TIME    ORA     A       ;CLEAR CARRY
000A    79                  MOV     A,C     ;UPDATE SECONDS
000B    3C                  INR     A
000C    27                  DAA
000D    4F                  MOV     C,A
000E    FE 60               CPI     #$60    ;MINUTE UPDATE?
0010    FA 2800             JM      SHOW    ;NOT NOW
0013    0E 00               MVI     C,#$00  ;YES, CLEAR SECONDS

0015    78                  MOV     A,E     ;UPDATE MINUTES
0016    3C                  INR     A
0017    27                  DAA
0018    5F                  MOV     E,A
0019    FE 60               CPI     #$60    ;HOUR UPDATE?
001B    FA 2800             JM      SHOW    ;NOT NOW
001E    1E 00               MVI     E,#$00  ;YES, CLEAR MINUTES

0020    7A                  MOV     A,D     ;UPDATE HOURS
0021    3C                  INR     A
0022    27                  DAA
0023    57                  MOV     D,A
0024    FE 13               CPI     #$13    ;USE #$24 FOR 24 HOUR CLOCK
0026    FA 2800             JM      SHOW
0029    16 01               MVI     D,#$01  ;USE #$00 FOR 24 HOUR CLOCK

002B    21 F800 SHOW        LXI     H,POINTH ;*KIM*
002E    72                  MOV     M,D      ;MOVE TIME TO DISP, BUFFER
002F    2B                  DCX     H
0030    73                  MOV     M,E
0031    2B                  DCX     H
0032    71                  MOV     M,C

0033    06 88   NXTKY       MVI     B,#$88  ;SET TIME DELAY VALUE

                *******ALTERNATE FOR INTERRUPT TIMEBASE*******

*0033   06 37   NXTKY       MVI     B,#$37  ;DEBOUNCE INTERRUPT*

                ************************************************
```

```
0035    CB 6500 WAIT    C65     DSKBD-1 ;6502 DISPLAY&KEYBOARD SUB
0038    FE 0A           CPI     #$0A    ;NUMERIC KEY PRESSED?
003A    FA 4500         JM      SET     ;YES, GO SET TIME
003D    05              DCR     B       ;NO, COUNT DOWN DELAY
003E    C2 3500         JNZ     WAIT    ;KEEP WAITING
0041    FB              EI              ;ENABLE INTERRUPTS
0042    C3 0900         JMP     TIME    ;GO SET TIME

                *******ALTERNATE FOR INTERRUPT TIMEBASE*******

*0042   C3 3500         JMP     WAIT    ;KEEP WAITING FOR INTERRUPT*

                **************************************************

0045    47      SET     MOV     B,A     ;SAVE INPUT IN B
0046    EB              XCHG            ;SHIFT THE TIME ONE DIGIT LEFT
0047    29              DAD     H       ;BY ADDING HL TO ITSELF
0048    29              DAD     H
0049    29              DAD     H
004A    29              DAD     H
004B    EB              XCHG

004C    79              MOV     A,C     ;AND ROTATING THE SECONDS LEFT
004D    07              RLC
004E    07              RLC
004F    07              RLC
0050    07              RLC

0051    4F              MOV     C,A
0052    E6 0F           ANI     #$0F
0054    B3              ORA     E
0055    5F              MOV     E,A

0056    79              MOV     A,C
0057    E6 F0           ANI     #$F0
0059    B0              ORA     B       ;INSERT NEW DIGIT
005A    4F              MOV     C,A

005B    CB 6500 DBNC    C65     DSKBD-1 ;WAIT FOR KEY RELEASE
005E    FE 15           CPI     #$15    ;RELEASED?
0060    FA 5B00         JM      DBNC    ;NOT YET
0063    C3 2B00         JMP     SHOW    ;YES, GO SHOW TIME

                *******6502 SUBROUTINE CALLED FROM 8080 PROGRAM*******
                *******(CALL MUST BE TO SUBROUTINE ADDRESS - 1)*******

0066    20 1F1F DSKBD   JSR     SCANDS  ;*KIM* DISPLAY BUFFER
0069    20 6A1F         JSR     GETKEY  ;READ *KIM* KEYPAD
006C    85 E5           STAZ    A       ;RESULT TO 8080 ACCUMULATOR
006E    60              RTS             ;RETURN TO SIMULATOR

                *******8080 STACK #3006F-#30074*******
```

8080 SIMULATOR ADDENDA

1. SPECIAL MINIMUM CONFIGURATION PROCEDURE

This procedure does not require the control configuration hardware on KIM port PB or the use of the NMI interrupt (KIM's ST key). It is limited to running 8080 programs (i.e., no single-step or other features).

1. Load the Simulator programs from tape as specified in steps 1 - 16 of the Tape Loading Instructions of the User Manual.
2. Be absolutely certain to put #$FF in location #$00F1 !
3. Change #$173F from #$60 to #$D0.
4. Change #$1790 from #$EA to #$EF.
5. Examine location #$1780 - it contains #$A2.
6. Push the GO key.
7. The demonstration 8080 Time Of Day Clock program is now running. You should see a 1HZ count begin in the display.
8. The clock may be set by entering the time from the keypad in rapid sequence, less than 1 second between digits. The time digits will shift in from right to left. If you make a mistake, just begin again and the new time will be entered.
9. When restarting the program or another 8080 program, be sure to set the 8080 program counter to zero, or to whatever your 8080 program starting address may be.

2. TELETYPE NOTES

The following routines may be called from your 8080 program to communicate with your teletype via KIM teletype routines. They may be located anywhere in memory. Call them from your 8080 program with the special "C65" opcode in the format, CB XXXX, where XXXX is the address of GET80-1 or OUT80-1, least significant byte first.

Read a character from the teletype:	20 5A1E	GET80	JSR GETCH
	85 E5		STAZ A
	60		RTS
Write a character to the teletype:	A5 E5	OUT80	LDAZ A
	20 AC1E		JSR OUTCH
	60		RTS

3. ADDITIONAL RELOCATION REQUIREMENTS

1. Don't change the third byte of three byte references to page zero. (i.e., Addresses 0142, 0367, 036C, 0384)
2. Change references to INPG2 (at 17B9 and 03BA) from #$02 to the most significant byte of the page to which you are moving page two data.
3. Don't attempt to relocate the Simulator at an arbitrary address. To do so would require modifying the page select logic on page 6 of the Simulator listings. For simplicity sake keep the code in the same position relative to the page boundary.

If arbitrary relocation is a necessity, change XFRTBL to two byte entries, including the most significant byte of address with each entry. Change the logic on page 6 to extract both bytes from the table and push them onto the stack.