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

an 8080 simulator-debug package

Introduction

Why imitate one microprocessor with another? You probably purchased the Apple-80 to do one or more of the following:

* Run existing 8080 software on your Apple II.
* Test and debug your own 8080 software without having to buy a complete 8080-based system.
* Learn about the architecture and instruction set of the 8080 via hands-on experience.

After reading this manual and trying the 8080 demo program, you will soon be doing all this and more.

Note that this is only a manual - all it´s instructions are useable only after the Apple-80 is loaded into your Apple II.

When you are done reviewing this manual, refer to the section on ´Getting Started´ for instruction on loading the Apple-80 and trying the 8080 demonstration program.

Run Mode

To begin execution of an 8080 program push the ´G´ key. The simulator will begin to run from the location shown in the program counter. It will no longer directly affect the display unless your 8080 program does.

To stop operation press ´CONTROL C´ the simulator will stop and all 8080 registers will be displayed. Reaching a breakpoint or an illegal op-code will do the same, sounding the bell.

While running, all Apple I/O devices may be used by your 8080 program but use caution as control keys are used for simulating interrupts.

Single-Step Mode

Each time you press the space bar the next 8080 instruction step will be taken. This instruction is displayed to the right of the program counter under the label ´op´. It always shows the next instruction to be executed. executing an illegal op-code will cause the bell to sound.

As each step is taken the program counter and other 8080 registers will be visible changing according to the sequence of your 8080 program. Pushing ´N´ will display the next byte of 8080 program without executing it. Push ´^´ to step backward in memory.

Trace Mode

Pressing the ´T´ key will start the simulator tracing through the 8080 program. After each step the new contents of all registers will be displayed.

The speed of the trace is determined by the setting of paddle 0. Turning paddle 0 one way and another will change the trace speed while the simulator is running.

To stop tracing and return to the single-step mode press ´CONTROL C´. breakpoints will also cause a halt, as will illegal op-codes.

I/O Ports

The simulator I/O ports are arranged as a table of port addresses starting at C00 hex. Note that the addresses are entered in normal order, i.e., most signifigant byte first. The first entry in the table corresponds to 8080 port 0, the second to 8080 port 1, and so on. Ports 0 - 63 are simple memory-mapped I/O while ports 64 - 127 assume a data-direction register at the port address+1, as in a 6530 PIO.

You may assign any 8080 port to any I/O address in your Apple by using the Apple II monitor to change entries in the I/O table and then re-saving the 8080 simulator on tape.

The Apple-80 I/O ports are currently configured to match the list of Apple I/O devices beginning on page 137 in the Apple II Reference Manual. 8080 port 0 is the Apple keyboard input, port 1 is the keyboard strobe, port 2 is the cassette output, etc.

Output to ports 64 - 127 result in hex FF being written to the port address+1 for proper setting of the associated data direction register. Input from these ports causes hex 00 to be written to the data direction register.

See also the ´Embedded 8080 Code´ section of this manual.

Breakpoints

The indicator at the right end of the display, labled ´BK´, will show ´Y´ or ´N´ depending on whether or not any breakpoints are set.

To set or change breakpoints type ´B´. Type in up to 8 addresses and finish with a carriage return. When the simulator´s program counter matches any breakpoint address while in the trace or run mode, a return to the single-step mode will be forced, halting the simulator and displaying all registers. By positioning the cursor appropriately and entering a carriage return you may delete one or more breakpoints.

Interrupts

The listed control keys will generate the corresponding interrupts:

CONTROL P = RST 0 CONTROL Q = RST 1
CONTROL R = RST 2 CONTROL S = RST 3
CONTROL T = RST 4 CONTROL U = RST 5
CONTROL V = RST 6 CONTROL W = RST 7

Note that ´CONTROL U´ is identical to the ´back arrow´ key. If you plan to use the Apple II keyboard from your 8080 program be sure to put in at least a return (C9) at 1028 hex.

All RST instructions are vectored to addresses in page 10 hex rather than addresses in page 00 hex.

Restart Addresses

RST 0 = 1000 hex RST 1 = 1008 hex
RST 2 = 1010 hex RST 3 = 1018 hex
RST 4 = 1020 hex RST 5 = 1028 hex
RST 6 = 1030 hex RST 7 = 1038 hex

To avoid conflicts with the keyboard strobe, interrupts are debounced by requiring a different key (interrupt or otherwise) to be pressed between repetitions of the same interrupt.

Calling 6502 Subroutines

The 8080 simulator ´borrows´ one of the 8080´s unimplemented op-codes for use as a call to 6502 subroutines. The op-code is cb hex and the format is ´CB XXXX´, where ´XXXX´ is the address of the 6502 subroutine with it´s least signifigant byte first. The only register passed between the 8080 and the 6502 subroutine is the accumulator.

If other registers must be passed to or from the 6502, write appropriate machine language headers and trailers to transfer the desired registers. See the demo program listing for examples.

Register Modification

Pressing ´R´ will cause a cursor to appear in the row of 8080 registers. Use the keyboard arrows to tab to the register you wish to modify, type in your new hex data and enter a carriage return. the internal 8080 registers will be updated per your input.

In order to modify your 8080 program it is necessary to return to the Apple-II monitor. For details, refer to the ´Using the Apple Monitor´ section of this manual.

The ´INTE´ register can be changed only by your 8080 program.

Memory Usage

The 8080 simulator is a machine language program which interprets 8080 object code. In order to do this it must be co-resident in memory with an 8080 program. The simulator and associated tables occupies hex addresses 800 to FFF and it uses zero page addresses 0 to 1F. This precludes the use of the ´Sweet 16´ interpreter and the secondary lo-res display page.

8080 program workspace begins at 1000 hex. Care must be taken to avoid accidentally modifying with your 8080 program any memory used by the simulator.

Using the Apple Monitor

The Apple II monitor works hand in hand with the 8080 simulator. It must be used to enter 8080 object code into memory and to save it on tape for later use.

When you wish to enter an 8080 program into memory or modify an existing one simply press ´M´ to return to the Apple monitor. Use standard monitor commands (refer to your Apple II manual for details) to load, save or modify your 8080 program. To return to the simulator with registers as they were type ´CONTROL Y´. For a cold start (i.e., PC and INTE RESET, breaKpoints cleared) type ´800G´.

Command Summary

´SPACE´ executes the next instruction.

´N´ advances the program counter.

´^´ decrements the program counter.

´B´ allows breakpoints to be entered.

´R´ allows register modification.

´T´ enters the trace mode.
´CONTROL C´ halts the trace.

´G´ for ´GO´ starts the run mode.
´CONTROL C´ halts the simulator run.

´CONTROLS P-W´ result in interrupts.

Getting Started

The next program on this tape is the object code of the Apple-80. It is a machine-language program. Load it by pressing ´RESET´, typing the command ´800.1100R´ and starting the tape. This also loads the 8080 demo program.

When the tape is loaded, type ´800G´. All 8080 registers will appear on the screen. The 8080 program counter will be set to 1000 hex, which is the first address of 8080 program space. At this time various single keystroke commands will have effect as described in the other sections of this manual.

Embedded 8080 Code

If you forgo use of 8080 I/O ports 100 to 127, you may wish to use a routine there which enables embedding 8080 code in 6502 programs. The routine, ´AP80´, begins at hex address 0CDE. use it as follows:

```
20 DE0C JSR AP80
.. .... (your 8080
.. .... code here)
D9 RAP (special ´RETURN´)
.. .... (6502 code continued)
```

Be sure to take the normal precautions to avoid interference between 8080 and 6502 registers and memory.