Displaying Data on A15-A8 on the Altair 8800

The technique used in the Kill the Bit program to display a value on the upper 8 address LEDs is to repeatedly access an address in a loop for which the MSB of the address is the value to display on the address LEDs. The program itself must remain in the lower 256 bytes of RAM.

The use of the DAD instruction to count cycles through the loop instead of using DCX or INX is intentional. Though not documented, the 8080's address lines are driven during intermediate instruction states for the INX, DCX, INR, and DCR instructions. These instructions, in turn, cause unwanted light patterns to appear on A15-A8.

It appears the 8080 may use the PC increment logic to implement the INX, DCX, INR, and DCR instructions. The side effect of this is that the address lines are driven for a brief period during the execution cycle by the value of the register or register pair being incremented. B, D, and H appear on A15-A8. C, E, and L appear on A7-A0. This is true for both 16 bit and 8 bit increments.

This effect is not noticeable as a program is single-stepped from the front panel since the address lines are valid (an actual memory address) during the wait state used to stop execution between each single-step.

UPDATE:

The '79 die revision of the Intel 8080 still drives the address lines for 16 bit INX, DCX, but the 8 bit INR, DCR no longer drive the address lines.

UPDATE:

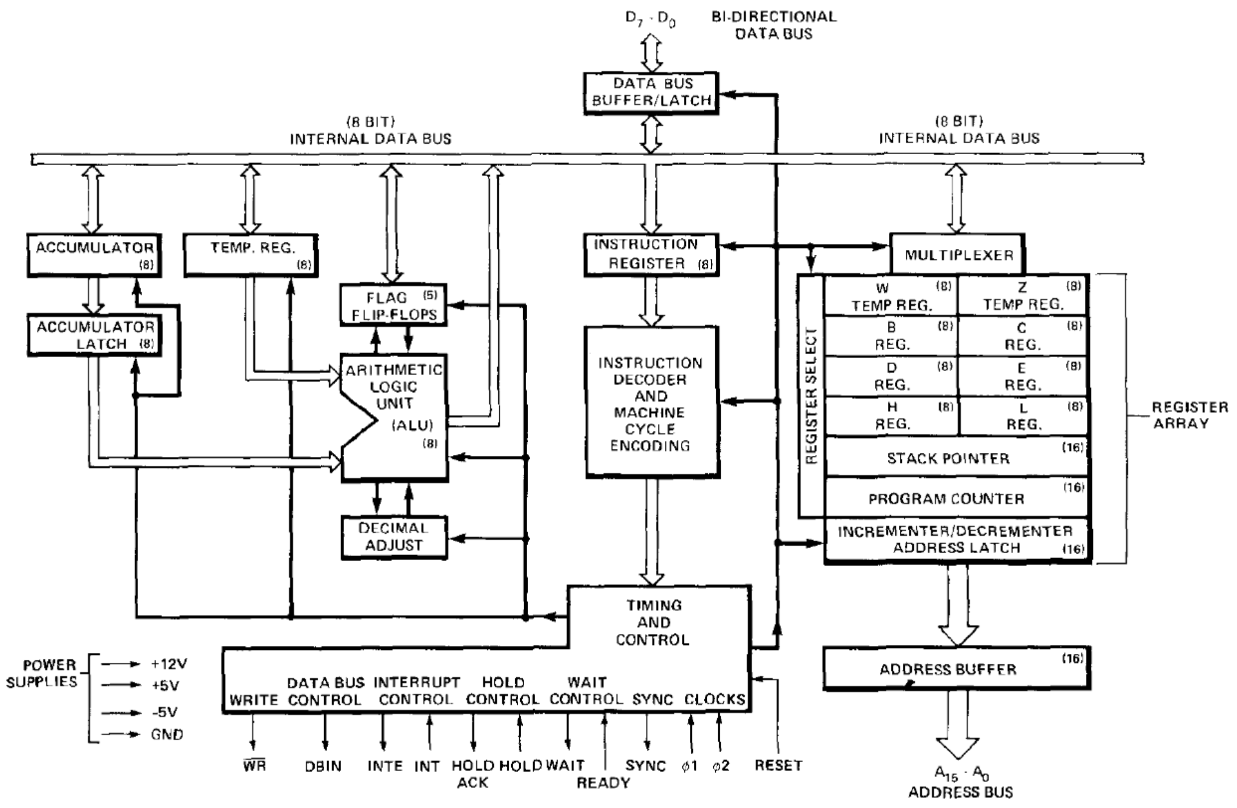
I originally considered the INX, DCX address line behavior an anomaly, but as I dug into why the behavior occurs, it is quite clear why it happens. Because of that, I was able to predict a few other instructions that produce similar results. For example, you can write Kill the Bit with SPHL and XTHL instructions as well.

The 8080 register file is organized as a collection of 16 bit registers that include BC, DE, HL, SP, PC, WZ (working registers), an address latch, and a 16 bit incrementer/decrementer. The output of the address latch drives the address pins through a buffer that is enabled at all times other than during HOLD and RESET states. The output of the address latch also feeds the incrementer/decrementer, the output of which can be routed back into any of the 16 bit registers. The incrementer/decrementer can simply pass through value of the address in addition to incrementing/decrementing the value.

This register file organization explains why INX/DCX and other instructions drive the address pins. In order to increment/decrement a register, or to move one register to another register, the value is moved to the address latch so that the increment/decrementer can then perform it's +/-/none operation and route the result to the destination register. While the value is in the address latch, it is driven onto the address pins.

At first thought, I predicted the XCHG instruction would also drive the address pins (exchange HL with DE). I assumed it would probably work like this: Move HL to WZ (HL shows on address pins), move DE to HL (DE shows on address pins), move WZ to DE (WZ shows on address pins). If HL=DE, then XCHG could be used to display HL (DE) as well. However, when I looked at the instruction state table for XCHG, it was clear the instruction does not go through this sequence of steps. In fact, it executes in just 4 cycles like a NOP and the other fastest instructions in the 8080. I figured it must have special hardware in the register file that allows exchange of HL and DE in a single machine state. Testing proved this out - XCHG is optimized to not pass through the address latch and incrementer.

However, 8-bit increments and decrements do still remain a bit of an anomaly. In 8080's prior to the pre-1979 die revision, these instructions are clearly done by the 16-bit incrementer even though the earliest datasheets show 8-bit INR/DCR going through the ALU in the instruction state table.


D7 - D0
BI-DIRECTIONAL DATA BUS
DATA BUS BUFFER/LATCH
(8 BIT) INTERNAL DATA BUS
(8 BIT) INTERNAL DATA BUS
ACCUMULATOR (8)
TEMP. REG. (8)
ACCUMULATOR LATCH (8)
FLAG (5) FLIP-FLOPS
ARITHMETIC LOGIC UNIT (ALU) (8)
DECIMAL ADJUST
INSTRUCTION REGISTER (8)
INSTRUCTION DECODER AND MACHINE CYCLE ENCODING
MULTIPLEXER
REGISTER SELECT
W (8) TEMP REG.
Z (8) TEMP REG.
B (8) REG.
C (8) REG.
D (8) REG.
E (8) REG.
H (8) REG.
L (8) REG.
STACK POINTER (16)
PROGRAM COUNTER (16)
INCREMENTER/DECREMENTER ADDRESS LATCH (16)
REGISTER ARRAY
TIMING AND CONTROL
POWER SUPPLIES
+12V
+5V
-5V
GND
WRITE
DATA BUS CONTROL
INTERRUPT CONTROL
HOLD CONTROL
WAIT CONTROL
SYNC
CLOCKS
ADDRESS BUFFER (16)
WR
DBIN
INTE
INT
HOLD ACK
HOLD
WAIT
READY
SYNC
ϕ1
ϕ2
RESET
A15 - A0 ADDRESS BUS