at any time, and the target can only go in empty space. So if it hits something (i.e., the trail), it is immediately erased at line 02A0, the target flag is reset, and the game continues without a target on display until the VIP decides to create a new one. This will occasionally result in a flash and a momentary jerk of the trail, but this is hardly objectionable. A routine test to see if the location is vacant could be written, which would be invisible to the game player. Execution of such a routine would take time, however, and would be best done in machine language. The results don't seem to me to be worth the trouble.

And so ends the main program loop, with return instructions at 02A4 and 0276. The following section describes the XY coordinate control, target hit, and scoring routines (as I promised we would get to in due time).

Lines 02A6-02C8 contain one of the more interesting routines of the program. V7 is a memory index which is added to the value of I (initialized to 0400), and then incremented by 2 for each new pass. This will form an upward growing stack (or array) for storing successive XY trail coordinates in memory with the F155 instruction at 02B0. The trail length will occupy two bytes for each of its "bips". A trail of 80 bips will need memory space from 0400-04BF, and the largest possible trail (128 bips) would fill the memory page. V7, then, determines the length of the trail, and when V7 is at its maximum value, the section at 02BE is enabled (by setting flag V6=1). This section erases the bips from behind. The memory index V7 is reset to 0, and will therefore limit the size of the storage array to only the length needed. Since the index starts over when the trail reaches maximum length, the array appears circular, with the last position erased just two bytes ahead of the first bip on the trail.

The size of the trail may be altered by changing 02B2 to skip when V7 reaches the desired trail length times two. (This,