don't notice the switch (performed by two other machine language subroutines at 03BE and 03C4) because the border is the same for both. The only reason for the routine is to allow fast restart of the trail after hitting a target. The trail appears to disappear, the score comes on, the entire trail returns and the game restarts. Actually, the trail is always there. You're simply looking at a display that always has a border (or for a wrap-around game, a non-border). It's a nice effect that could be used for many other games! (I haven't detailed the machine language routines, but they're simple enough, and should be easy to figure out.)

After initializing all variables with lines 022C and 022E, the game begins. I always include this initialization procedure, keeping the starting values for each variable in a data array - at 03F0, in this case. It then becomes a simple matter not only to change initial values, but more importantly, to add a new variable which wasn't thought of when I started.

Locations 0230-023C cycle V0 one time through values 2, 4, 6 and 8 to test if the key is pressed. If one has been pressed, its value is in V0; then V3 and V4 are set to zero. These two variables will later be added to the coordinates of the trail, to create the new position, and to give the appearance of movement. If no keys are pressed, then V3 and V4 do not change and the trail will continue in the direction last selected.

Locations 0242-0254 select new values for V3 and V4. The values are then added to the trail's X and Y coordinates: V8 and V9. This section is cycled through, even if no key is pressed, for the timing reasons mentioned earlier; however, the jump at 023A could be to 0252 with only a small change in effect.

Control now passes to a routine at 02A6. We'll discuss this later (it's rather complex and we'd get side-tracked from here), although its function is to record the XY coordinates of each