

Chapter 44 - A Larger Whole-Machine Example

The earlier chapters teach one card at a time. This chapter puts the habits together: frame timing, a blitter sprite, keyboard input, a SoundChip voice, file save, and a coprocessor status check.

It is still a type-in programme. It is not large enough to be a game engine, but it has the same shape as one.

44.1 What It Uses

| Part | Chapter |
|---------------------------------|---------------------------|
| VideoChip mode and blitter fill | Chapter 4 |
| SoundChip tone and envelope | Chapter 12 |
| Keyboard status | Chapter 37 |
| Frame wait | Chapter 31 and Chapter 40 |
| File save | Chapter 35 |
| Coprocessor status | Chapter 32 |

The programme draws a small moving block. Pressing a key changes its direction. The block moves once per frame. A tone plays while it moves. At the end, the programme writes a small file through the File I/O registers and prints the current coprocessor worker-state mask.

44.2 Type The Programme

```
10 REM WHOLE MACHINE BLOCK
20 POKE32 &H000F0004,4
30 POKE32 &H000F0080,0
40 POKE32 &H000F0084,&H00100000
50 POKE32 &H000F0000,1
60 POKE32 &H000F0800,1
70 SOUND 0,220,180,1,128
80 ENVELOPE 0,4,8,120,8
90 GATE 0,ON
100 X=20:Y=40:DX=4
110 FOR F=1 TO 90
120 POKE32 &H000F2580,1
130 POKE32 &H000F0028,&H00100000
140 POKE32 &H000F002C,320
150 POKE32 &H000F0030,200
160 POKE32 &H000F003C,&H00000000
170 POKE32 &H000F0020,1
180 POKE32 &H000F001C,1
190 WAIT &H000F0044,2
200 OFF=&H00100000+(Y*320+X)*4
210 POKE32 &H000F0028,OFF
220 POKE32 &H000F002C,16
230 POKE32 &H000F0030,16
240 POKE32 &H000F003C,&H0000FF40
250 POKE32 &H000F0020,1
260 POKE32 &H000F001C,1
270 WAIT &H000F0044,2
280 IF PEEK8(&H000F072C) AND 1 THEN DX=-DX
290 X=X+DX
300 IF X<0 THEN X=0:DX=4
310 IF X>304 THEN X=304:DX=-4
320 NEXT F
330 GATE 0,OFF
340 N=&H00730200:D=&H00730220
350 POKE8 N,66:POKE8 N+1,76:POKE8 N+2,79:POKE8 N+3,67
360 POKE8 N+4,75:POKE8 N+5,46:POKE8 N+6,68:POKE8 N+7,65
370 POKE8 N+8,84:POKE8 N+9,0
380 POKE8 D,88:POKE8 D+1,0
390 POKE32 &H000F2200,N:POKE32 &H000F2204,D
400 POKE32 &H000F2208,2:POKE32 &H000F220C,2
410 PRINT "FILE ";PEEK32(&H000F2210)
420 PRINT "WORKERS ";PEEK32(&H000F2374)
```

Expected result: a coloured block moves left and right on the VideoChip layer. A tone plays while it moves. If a raw key is waiting, the programme reverses direction. At the end it writes BLOCK.DAT, prints FILE 0 when the write succeeds, and prints the coprocessor worker-state mask.

44.3 How The Programme Is Built

Lines 20 to 50 enable a 320 by 200 VideoChip framebuffer at \$00100000.

Lines 60 to 90 start one SoundChip voice. The sound is not tied to the frame loop. It keeps playing while the CPU updates the picture.

Lines 110 to 320 are the frame loop. Line 120 waits for VBlank. Lines 130 to 190 clear the framebuffer with the blitter. Lines 200 to 270 draw a 16 by 16 block at the current position. Lines 280 to 310 read keyboard status and update the position.

Lines 330 to 420 stop the sound, write the NUL-terminated name "BLOCK.DAT" and two data bytes into RAM, fire the File I/O write operation, and print the coprocessor worker-state register. A file status of 0 means the write succeeded. A worker-state value of 0 simply means no worker is running.

44.4 Variations

Try these small changes one at a time:

| Change | Effect |
|---|--|
| Line 70, change 220 to 330 | Higher tone. |
| Line 240, change \$0000FF40 to \$00FF8040 | Different block colour. |
| Line 100, change DX=4 to DX=2 | Slower horizontal movement. |
| Line 110, change 90 to 180 | Longer run. |
| Line 420, start a worker before running | Worker-state mask shows the active worker bit. |

44.5 Why This Is The Shape

The programme follows the rule used by most IE projects:

1. Configure devices once.
2. Allocate or choose buffers.
3. Wait for a frame boundary.
4. Update memory and MMIO.
5. Poll completion bits that matter.
6. Save or inspect state at the end.

Nothing here needs an external tool. The same pattern scales to machine-code parts: move the hot loop into IE64, 6502, Z80, M68K, or x86 code, then keep the same bus-visible buffers and device registers.

44.6 Limits

- The input test uses raw key status, not a full keyboard scanner.
- The programme clears the whole framebuffer every frame. A larger game would use dirty rectangles, sprites, or a back buffer.
- The block is drawn with FILL; it is not a masked sprite.
- The coprocessor line only reports worker state. Chapter 42 shows a positive worker request.

This is the smallest useful whole-machine shape: picture, sound, input, timing, storage, and shared-bus inspection in one typed listing.