

Chapter 49 - The Rotozoomer In IE64 And IE32

The IE64 and IE32 rotozoomers are the native versions of the BASIC effect. They still use VideoChip Mode 7. They still use a texture, a back buffer, a front buffer, VBlank, and audio playback. The difference is that the frame loop is now integer machine code.

This chapter does not print full listings. It shows the pieces to look for when reading them.

49.1 Shared Layout

Both versions use the same practical layout:

Area	Purpose
\$001000	Code and tables.
\$00100000	VideoChip front framebuffer.
\$00600000	Texture.
\$00900000 and above	Back buffer or back buffers.
\$000F0000 upward	VideoChip and audio MMIO.

The layout is chosen for separation. The blitter can read the texture while writing the back buffer, and the display can scan from the front buffer.

49.2 Setup Excerpt

The setup work is the same work BASIC performed:

```
; choose VideoChip mode 0 and enable display
store.l r1, (VIDEO_MODE)
store.l r2, (VIDEO_FB_BASE)
store.l r3, (VIDEO_CTRL)

; copy or build texture
; start the selected audio player
; initialise angle and scale accumulators
```

The exact instruction spelling differs between IE64 and IE32, but the register targets do not.

49.3 Table Lookup

The BASIC version uses SIN and COS. The native versions use a table:

```
; angle_hi = angle >> 8
; sin = sine_table[angle_hi]
; cos = sine_table[(angle_hi + 64) & 255]
; recip = recip_table[scale_hi]
```

IE64 has a wide register file and fixed 8-byte instructions. IE32 has a smaller, simpler instruction set. Both reduce the per-frame work to a few loads, multiplies, shifts, and stores.

49.4 Affine Parameter Calculation

The native loop computes the same Mode 7 values as Chapter 46:

```
duCol = CA
dvCol = SA
duRow = -SA
dvRow = CA
U0     = centreU - halfW*CA + halfH*SA
V0     = centreV - halfW*SA - halfH*CA
```

In IE64 and IE32 the intermediate values are integer fixed point. That is why Chapter 48 matters: the native code is not a new effect, only a table-driven version of the BASIC effect.

49.5 Blitter Programming Excerpt

The Mode 7 write sequence is recognisable in every CPU version:

```
; BLT_OP = MODE7
; BLT_SRC = texture
; BLT_DST = draw buffer
; BLT_WIDTH = 640
; BLT_HEIGHT = 480
; BLT_SRC_STRIDE = 1024
; BLT_DST_STRIDE = 2560
; BLT_MODE7_U0/V0 = computed origin
; BLT_MODE7_DU/DV = computed deltas
; BLT_MODE7_TEX_W/H = 255
; BLT_CTRL = START
```

When reading the listing, ignore the CPU syntax at first. Follow the MMIO names. They tell you what the machine is doing.

49.6 Wait, Present, Advance

The loop ends with three jobs:

1. Wait for VBlank or a safe presentation point.
2. Present the completed buffer by copy or framebuffer-base update.
3. Advance the angle and scale accumulators.

That is the same loop from Chapter 45, only with table arithmetic and a hardware Mode 7 draw in the middle.

49.7 IE64 Compared With IE32

Topic	IE64	IE32
Instruction shape	Fixed 8-byte native instructions.	Compact 32-bit RISC-style instructions.

Topic	IE64	IE32
Registers	More wide registers for address and fixed-point temporaries.	Smaller register set, but enough for the six parameters.
Addressing	Direct low-window MMIO and full IE64 physical path.	Direct low 32-bit bus window.
Best role	Native main programme and wide-address control.	Compact integer worker or portable native target.

Chapter 50 compares the same effect across all six CPUs.