ROAD 1.0 RC2 – more speed, and why the road stays with BOX

Short report on the measurements on the Pico Computer 3 · 378 MHz · MODE 2 · firmware 7 and 8.00.00b2 · Matthias (homa), 2026

In short: RC2 changes the speed and nothing else. In the busiest scene (the start of Green Valley, about 40 palms) the frame rate goes from **14 to 17–18 fps**; on open road it runs at 20–23 fps. RC2 still runs on firmware 7 and on firmware 8. Only ROAD.BAS is new; the track files stay as they are.

1. Where the time goes

One frame at the start of Green Valley took 59 ms (17 fps). This is how it splits up:

Part	ms	What happens
BG	7	sky and background pictures (BLIT FLASH)
RD + PO	19 + 4	road: project 120 segments, collect the rows as BOX and draw them
SP	20	sprites: palms, signs, opponents, own car
Rest	9	game logic, opponents, sound, HUD, frame to the screen

The most important rule: on the PicoMite every *BASIC statement* costs time, hardly every *pixel*. Even with OPTION COMPILE ON, where the compiler turns the program into ready-made operations beforehand, every statement stays a step of its own with its own overhead. A firmware command such as BOX, on the other hand, does the whole drawing job in one go. Measured: drawing 480 BOXes takes 0.55 ms – the BASIC arithmetic that prepares those 480 BOXes about 10 ms. So ROAD only gets faster with *fewer statements*, not with faster drawing commands.

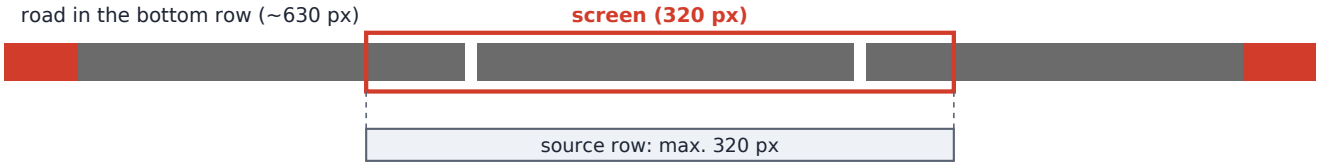
2. What was tried

Idea	Result	
BLIT RESIZE per road row (idea from the forum)	road 30 instead of 26 ms, far centre stripes partly missing	no
Shortcuts in the BOX code (skip rumble, lines, grass)	road 2–3 ms slower: the checks cost more than they save	no
FRAMEBUFFER WARPH (new in FW8)	very fast itself (0.4 ms), but does not fit ROAD – see 3.	no
Sort out sprites outside the screen first	only 3 of 42 were outside – no gain	no
Draw the scenery sprites without a SUB call	5–7 ms less per frame, same picture	RC2
Scenery distance 60 instead of 80 segments, only on busy stretches	+1–2 fps; 40 segments would visibly pop in	RC2

3. Why FRAMEBUFFER WARPH does not help here – simply explained

WARPH copies a picture row by row and may move each row sideways and take it from any source row. For the classic arcade road that is ideal: every screen row has a fixed road width there, curves are just shifts, hills a squeezing of rows. The whole road then costs less than 1 ms. ROAD is built differently:

- **The road is wider than the screen.** At the bottom it is about 630 pixels wide, the screen shows 320. WARPH can only *shift* a row, not stretch it (the command has no parameter for scaling) – and a source row is at most 320 pixels wide, so only half the road. When the car changes lanes, exactly the part that now comes into view is missing (picture below).
- **The width is not fixed.** Real hills, narrower zones and a changing number of lanes need a source row of their own for every width, colour and lane count: more than 270 rows. Only 22 are free in the buffer – the rest holds the sprites. The extra buffers L and T are always shown in MODE 2, so they cannot serve as hidden storage.
- **The time is spent elsewhere.** Even if the road cost nothing at all (measured by simply leaving it out in a test), the frame rate in the palm avenue would only rise from 14 to 18 fps. The palms cost more than the whole road.



Bottom screen row: only the middle of the road fits into a source row. When the car moves right, road comes into view that WARPH does not have.

So WARPH is not a bad command – it is made for a road that is designed around it. ROAD follows the javascript-racer with real segments, and those are needed anyway for the positions of the palms and cars.

4. What RC2 does exactly

- **Scenery without a SUB call:** width and alignment of every palm are worked out once when loading; drawing leaves only a few statements per sprite. The picture is the same.
- **Scenery distance chosen automatically:** when loading, ROAD counts where more than 30 scenery sprites lie ahead. There the distance goes down from 80 to 60 segments – starting 100 segments earlier and by one segment per frame, so without a jump. Per track it can be set with `SCENERY, 80, 60, 30` in the .trk file.

Measurement (378 MHz)	RC1	RC2
Start of Green Valley, car standing	14 fps	17–18 fps
Open road, full throttle	18–20 fps	20–23 fps
Crossing the finish line	13–14 fps	15–18 fps

5. Ideas for later

- **ROAD 1.1 – sound with firmware 8:** the new MOD functions of FW 8 (e.g. the Doppler demo) come in ROAD 1.1. From then on ROAD needs firmware 8; RC2 stays the version for firmware 7.
- **Oil slicks and the like:** possible today – as a flat sprite on the road, triggered like the skulls in Texas, just with a slide instead of a crash.
- **Pictures on the road** such as zebra crossings, arrows or cracks in correct perspective would need pictures row by row – too slow in BASIC, possible with a WARPH road or in C.
- **"ROAD Turbo" with a CSUB:** road and sprite list worked out in C – that should bring clearly more fps. But then no longer pure MMBasic – so, if at all, as an experiment of its own.
- **A classic WARPH road** as a game variant of its own: very fast, but with a different view (higher camera, narrower road).

All values measured on real hardware (Pico Computer 3, 378 MHz, MODE 2), averaged over one second each from the info line (key I). Have a good drive!