AN AI WRITES MACHINE CODE FOR A 4-BIT COMPUTER FROM 1981

VIBE-CODING A 1981 MICROCOMPUTER

Vintage Computer Festival
VCF West 2026
live demo & talk

Claude (Anthropic) hand-writes Microtronic machine code — games, OLED graphics and speech — for the **Busch Microtronic 2090** + **PicoRAM 2090**, and debugs every one on real hardware. Six new programs, only existing op-codes, no firmware changes.

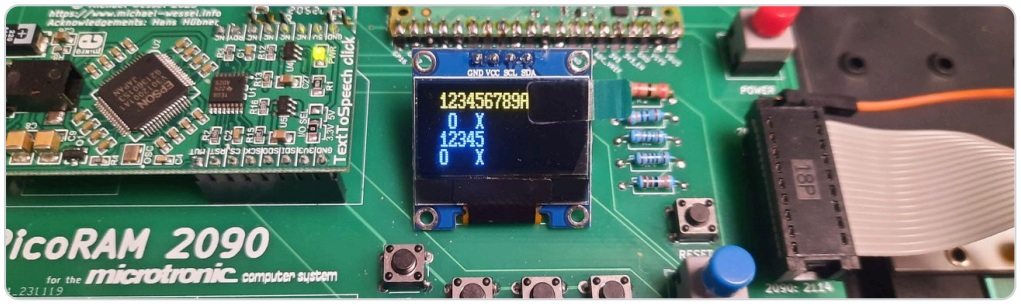
A first: the first **working Tic-Tac-Toe for the Busch Microtronic — ever.** The 1981 manual's own listing had an address typo and was never correctly run; we re-derived and verified the authentic 58-word strategy.

The showcase two-bank Tic-Tac-Toe is the first PicoRAM program to combine OLED graphics + speech + SRAM bank-switching + a real game AI **all at once**. (PicoRAM 2090 won the **RetroChallenge 2023/10 Grand Prize.**)

01 THE MACHINE

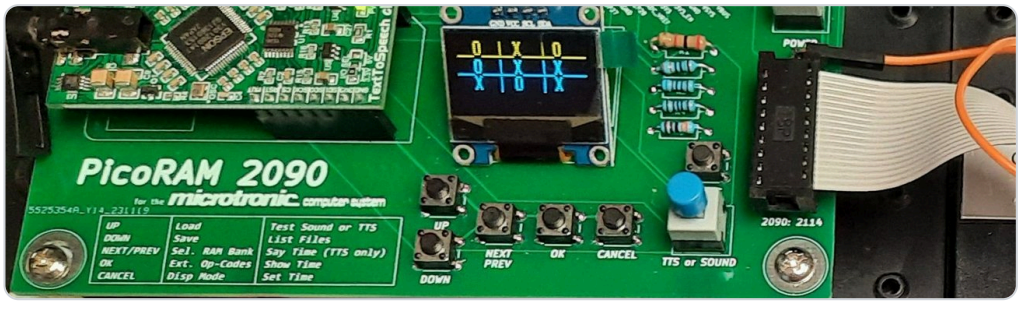
Busch Microtronic 2090 (1981, West Germany) — an educational microcomputer, a "learn how a computer works" trainer. A TI **TMS1600** 4-bit microcontroller (~500 kHz), **16 four-bit registers** (R0–RF), **256 program words** per bank, a hex keypad and a 6-digit red LED. You key in programs **one 3-hex-digit word at a time, by hand**; the monitor lives in a masked ROM — physically unchangeable.

PicoRAM 2090 (Michael Wessel, 2023) — a Raspberry Pi Pico in the Microtronic's 2114 SRAM sockets: it **emulates the RAM** and adds **SD-card load/save of memory dumps** — plus I/O the 1981 machine never had: a **128×32 OLED**, a **speech synthesizer**, **sound**, and a **battery-buffered real-time clock**.



PicoRAM 2090 — Pi Pico + speech chip + OLED on the Microtronic bus (© M. Wessel). OLED running **BLOCKADE**.

02 SEEN IN ACTION



PicoRAM's OLED running a live **Tic-Tac-Toe** game — the 3×3 grid drawn with line/char op-codes on the green PicoRAM 2090 (Pi Pico + speech chip).

Why it's hard: no compiler, no on-machine assembler, no screen editor — just 256 words of 4-bit memory and a keypad. Every byte of game logic, board graphics, AI and speech is placed by hand within that budget.

▶ **Try it at the table!** The Microtronic + PicoRAM is running right here — beat the Tic-Tac-Toe AI, land the Eagle, or solve MERLIN.

03 THE TRICK — NEW INSTRUCTIONS FOR A SEALED MACHINE

The Microtronic's firmware ROM is **mask-programmed into the TMS1600 chip itself** — there is no external ROM/EPROM to swap, and the chip cannot be reflashed. It is **doubly sealed**, so PicoRAM can't add real CPU instructions. Instead it **watches the data bus** for op-codes that never occur in normal programs — "vacuous" no-ops that change nothing — and repurposes them as new side effects:

```
50F = "add 0 to register F" → changes nothing → PicoRAM hears: SPEAK
0xx = "move register x to itself" → a no-op → PicoRAM reads it as an argument / GRAPHICS
```

The 4-bit CPU runs them as if nothing happened; the Pico gives them meaning — a whole graphics-and-speech API smuggled into the gaps of a 45-year-old instruction set (`502` clear · `504` refresh · `506` char · `508` cursor · `50A` line · `50F` speak) — plus `70x` to switch SRAM banks and break the 256-word ceiling.

04 THE PROGRAMS — SIX NEW GAMES

TIC-TAC-TOE ×2 — the first working tic-tac-toe for the Microtronic, ever. **Human-first (two-bank)** — you move first against a board-scanning tactical AI (win→block→center→corner→side): across all 489 lines of play it **wins 350, draws 127**, beatable only by a deliberate two-corner fork. AI in bank 0, OLED + speech in bank 1. **Computer-first** — the authentic Busch 58-word look-ahead, verbatim, with OLED + speech ("YOU?" / "I WIN" / "DRAW").

MERLIN + MERLIN 2 — graphical **"Lights Out"**: turn all nine OLED cells on; key beeps + a victory melody. MERLIN 2 adds difficulty — set a move budget and beat the move counter, or get a sad falling D-minor melody and a new puzzle.

BLOCKADE — the two-lane **Nim** game from the Busch 2094 "Computerspiele" booklet, logic (00–87) transcribed **word-for-word** from the manual, given an OLED makeover + "YOU WIN"/"I WIN" speech; flicker-free incremental redraw, exactly 256 words.

★ **LUNAR LANDER** — the odd one out: **not a port — a wholly original game designed from scratch** by the AI. Turn-based (the 1969 form). Red LED = **ALT·VEL·FUEL**; OLED = the window (lander, thrust flame, smoking crater); land at velocity ≤ 2 for **"THE EAGLE HAS LANDED"**, else **"CRASH"**. Pure-integer physics, 172 words.

```
descending    burning     landed      crashed
    A            A
                 V
 _______     _______     ___A___     __*X*__
```

▶ **Watch it built — live.**
A YouTube stream of Claude vibe-coding the Microtronic on real 1981 hardware.
youtu.be/Smc7S6Ny78w

05 HOW CLAUDE BUILT IT

- **Hand-wrote raw Microtronic machine code** — three hex digits per word, only existing op-codes, no firmware changes.
- Wrote an **assembler**, `masm.py` — readable `.asm` (labels, mnemonics, raw words) → loadable `.MIC`.
- Wrote the headline tool, `mtsim.py` — a **two-bank simulator that renders the OLED as text and transcripts the speech**, so an entire game is played and verified **off-device** before touching hardware.
- Modeled the quirks that actually bite: **no-borrow compare carry**, the `70x` bank-switch **prefetch** (handoff = `70x; NOP; GOTO`), and the `50x` argument stream (stray no-op words get swallowed as draw/speech args).
- Verified in the simulator, **then on a real Microtronic** — iterating with Michael over photos of the live hardware.
- **Learned the machine** from its 1981 German manual (which we translated to English); the extended op-codes are undocumented, so Claude read the PicoRAM firmware source.



The **Busch Microtronic 2090** — from the 1981 Busch brochure: hex keypad, 6-digit red LED, breadboard & DIN sockets. „Programmieren · experimentieren · spielend lernen" (program · experiment · learn by playing).

Programs by Claude (Anthropic) · with **Michael Wessel**, on real hardware · code & write-ups: **github.com/lambdamikel/picoram2090** → `software/vibe-coded` · Busch Microtronic

© Busch 1981, PicoRAM © M. Wessel 2023