MameDuo Projects

NEW PROJECT

MAME Score Scanner

A discovery and export tool for finding arcade score RAM, narrowing down candidates, and turning the winning addresses into second-screen hi-score layouts you can refine further in the online MameDuo Layout Builder.

Project Overview

Guide Included

Simple + Advanced modes

Exact + Delta workflows

Probe and final .lay export

Layout Builder friendly

Best use case

Start here when you need to discover score addresses or prove which candidate updates live in-game. Once the scanner gives you a clean final or probe layout, bring that result into the online Layout Builder if you want to add bezels, tweak presentation, or switch imported hi-score digits to artwork-based styles.

From RAM discovery to a usable hi-score layout

Discover

Record board-wide dumps, analyze likely score formats, and test candidate address ranges across supported BCD, integer, implied-zero, and special-case decode methods such as Galaga-style display digits.

Verify

When more than one winner survives, create probe layouts and compare them live in MAME so the screen tells you which address updates immediately and which one is only a parked or copied score slot.

Export

Build final centered stacked score layouts from P1, P2, HI, and helper rows, then carry those exported hi-score layouts into the Layout Builder if you want bezel work, second-screen polish, or digit-art styling.

Choose the analysis path that matches the game

Exact

Direct score matching

Use this when you believe the visible score is already stored directly as BCD or integer bytes. Enter the visible score, record full-board blocks, and let the scanner replay many exact score modes.

Delta

Before and after comparison

Use this when the score changes are easier to detect than the final format. Record multiple snapshots before and after scoring, then rank hot ranges and isolate a focused cluster for deeper checking.

Simple Mode

Normal daily workflow

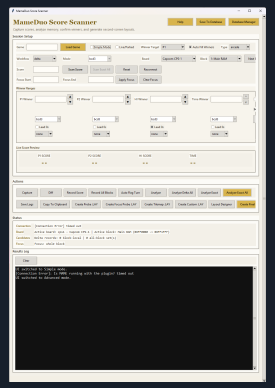
Best for quick score discovery: record all blocks, analyze across the whole board, fill winner boxes, and create probe or final layouts without needing manual block stepping.

Advanced Mode

Debugging and edge cases

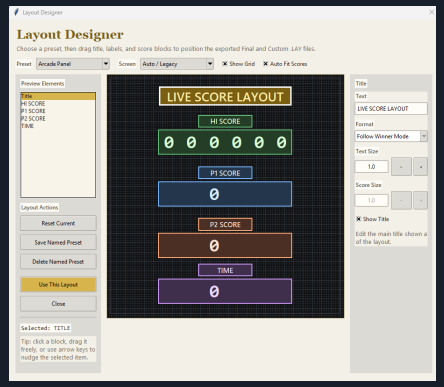
Best for discovery work: manual block stepping, manual mode forcing, focus ranges, turn-flag hunting, helper rows, and database saving once the result is solid enough to keep.

Menu workflow and built-in layout design



Main scanner workflow

The menu is built around recording, analyzing, probe generation, winner boxes, and export. It is designed to move from raw score dumps toward a final result without losing the investigative steps in between.



Layout Designer

Layout Designer handles preview and export arrangement for titles, labels, score rows, timers, and helper rows. It follows the real digit count for the selected mode and is especially useful before passing a layout into the online builder.

Use both tools together

1

Find the winning ranges

Use MAME Score Scanner to confirm the real live score ranges, create a final or probe .lay, and lock in P1, P2, HI, TIME, or helper-row behavior.

2

Import into the online builder

Bring that exported hi-score .lay into the MameDuo Layout Builder if you want to add bezels, second-screen cabinet presentation, local artwork refs, or grouped hi-score positioning controls.

3

Customize the final look

Use the builder to polish the imported score layout with bezel artwork and optional digit-art styles from imported hi-score tools, then export the presentation-ready layout you actually want to keep in MAME.

Good fit games and known special cases

- Pac-Man: confirmed bcd4rev Namco example
- Frogger: 5-digit bcd2rev_implied0 workflow
- Galaga: display-digit decode path
- Galaxian: reversed-byte 3-byte BCD
- Donkey Kong: strong winners even with mode overlap
- Popeye: live and parked slot behavior
- Genesis / Sonic: mixed helper data workflow
- SNES Mario World: timer helper decoding

What the app is for, the core modes, and supported score methods

What the app is for

The scanner helps you discover score RAM for arcade games so you can build a second-screen .lay file. In the best case it finds a winner automatically. In harder cases it narrows the field and creates probe layouts so you can verify the right address visually.

Main modes of use

- Simple Mode: normal day-to-day use with full-board recording, analyze, winner boxes, and final or probe layout creation.
- Advanced Mode: debugging and discovery work with manual block stepping, manual mode forcing, focus ranges, probe creation, and database work.

The two core workflows

- Exact: use this when you think the score is stored directly as a number.
- Delta: use this when you want to compare snapshots before and after score changes.

bcd2 / bcd2rev

bcd2_implied0 / bcd2rev_implied0

bcd3 / bcd4

bcd3rev / bcd4rev

bcd3_implied0 / bcd4_implied0

bcd3rev_implied0 / bcd4rev_implied0

u16 / u24 / u32 integer storage

u16 / u24 / u32 implied0

galaga_digits display decode

Families and everyday workflows

The board dropdown is meant to match clues users can see in MAME, such as Nintendo or Z80, not just internal scanner preset names. Start with the family that best matches the game's hardware or source information if you know it.

Capcom CPS-1 | Generic 64K | Namco 8-bit / 64K | Nintendo Z80 / 64K | Konami 8-bit / 64K | Taito 8-bit / 64K | Sega Z80 / 64K | Midway 8080 / 64K | Irem M62 / 64K | Capcom Pre-CPS / 64K

Typical simple mode workflow

- Load the game in MAME, then open the scanner so it can connect to the bridge.
- Let the scanner auto-detect the game if possible, or type/load the game profile manually.
- Choose the likely board family if it is not already set correctly.
- Enter a known score and click `Record All Blocks`.
- Score points, pause again, enter the new score, and repeat `Record All Blocks` several times.
- Run `Analyze Exact All` to test many supported score methods across the whole board.
- If a clean result is found, move it into the winner box and create the final layout.
- If several candidates remain, create a probe layout and verify them visually.

Typical advanced mode workflow

- Switch to Advanced Mode.
- Use Exact workflow when you believe the score is stored directly as BCD or integer bytes.
- Use Delta workflow when you need to compare changes or work through a noisy game.
- Step through blocks manually if the board-wide approach is too broad.
- Use `Apply Focus` to isolate a hot cluster after `Analyze` or `Analyze Delta All`.
- Use `Create Focus Probe .LAY` or `Create Probe .LAY` to watch uncertain survivors live.
- Use `Save To Database` when you are confident enough in the final ranges and mode.

What the main buttons and winner boxes do

Important buttons

- Load Game: loads a saved game profile if it exists, and also helps after bridge auto-detection fills the game field.
- Record All Blocks: captures a full set of board blocks for the score currently entered.
- Analyze Exact All: replays many supported score modes against all-block dumps across the selected board.
- Analyze Delta All: ranks hot blocks and suggests a focus cluster from all-block delta data.
- Create Probe .LAY: builds a visual compare layout from remaining exact candidates.
- Create Focus Probe .LAY: builds a raw hex/live-byte probe from the current focus range.
- Create Custom .LAY: builds a centered stacked score layout from addresses you enter manually.
- Create Final .LAY: builds a centered stacked score layout from the P1, P2, and HI winner boxes.
- Save To Database: stores the current game, board, mode, ranges, and notes in the local profile database.

Winner boxes

Winner boxes are the main handoff point between analysis and export. They should hold the full range, not just the first byte, for example `0x60B5->0x60B7`. If the scanner finds a result automatically it can place it in the currently selected winner target box.

You can also type or edit these boxes manually after confirming a probe. Each field can keep its own mode, leading-zero preference, and optional display-adjust helper of `none`, `+1`, or `-1` for preview/export-only corrections.

Turn flags, layout designer, and probe layouts

Turn flag workflow

- Use this only after Live, parked score fields, and HI are already confirmed for a live/parked game.
- Turn on Live/Parked mode and, if you are only testing, turn `Auto Fill Winners` off first.
- Pause on P1, click `Capture`, switch to P2 and pause again, then click `Diff`.
- Click `Capture` again on P2, switch back to P1 and pause, then click `Diff` again.
- Run `Auto Flag Turn` to rank likely turn-flag bytes and suggested P1/P2 flag values.
- Only fill `Active Flag`, `P1 Flag`, and `P2 Flag` after the parked ranges themselves are already confirmed.

Layout Designer

Layout Designer affects preview and export only. It lets you move the title, labels, and score rows, rename exported text, and choose row digit formatting for HI, P1, P2, and TIME using trim, minimum-2-digit, fixed-width, implied-trailing-zero, follow-winner, or timer-specific display styles.

It follows the real digit count for the selected mode, uses matching label and score colors, repacks only visible rows, auto-applies presets when selected, and remembers your current working layout while the app session stays open.

Time helper rows can also follow timer-specific export styles or digit-style helper decoding, and the left Preview Elements list remains the safest way to manage the layout because hidden rows stay selectable there.

Turn flags, layout designer, and probe layouts

How to use probe layouts

- Create a probe layout when more than one candidate remains.
- Load the probe in MAME and compare each panel against the live in-game score.
- Watch which panel updates immediately while playing and which one only updates later or after death.
- Put the confirmed winning range into the matching winner box.

Saving to the database

`Save To Database` is the way the scanner gets smarter over time. It stores the game name, board family, block, current mode, P1/P2/HI ranges, optional Time helper range, notes, solved status, and per-field preview/export helpers such as leading zeros and display adjust.

Extra helper rows are also preserved when used. For console systems, make sure the Game box contains the real game title rather than a generic system name such as `genesis` or `snes` before saving.

PreMame, known cases, and troubleshooting

PreMame for console per-game layouts

Some console systems boot through one shared artwork\system\default.lay, which makes per-game console layouts awkward if you want each cart to have its own small file. `PreMame` is an optional wrapper that solves that by checking the loaded cart name before MAME starts, temporarily swapping in a matching per-game .lay if one exists, and restoring the original `default.lay` after MAME closes.

- If a matching per-game .lay exists, PreMame temporarily uses it as `default.lay` for that launch.
- If no matching per-game .lay exists, the normal `default.lay` remains in use.
- The original console `default.lay` is restored automatically after exit.
- Arcade games do not need this unless you explicitly route them through the wrapper.

Known special cases

- Galaga uses display-byte decoding instead of plain BCD.
- Galaxian uses reversed-byte 3-byte BCD.
- Pac-Man is confirmed as a working `bcd4rev` Namco example using the known RAM map ranges.
- Donkey Kong can produce a strong address winner even when more than one mode still survives.
- Popeye uses live and parked score-slot behavior rather than a simple fixed P1/fixed P2 layout.
- Frogger is a strong example of a 2-byte reversed BCD score with an implied trailing zero, using `bcd2rev_implied0` and a 5-digit exported layout.
- Genesis/Sonic is a good example of mixed helper data: score works as an implied-zero integer, while timer preview is better treated as a dedicated helper field.
- SNES Mario World is a good example of mixed helper data too: score/lives are numeric HUD values, but the timer is better treated as a 3-digit display helper and may need preview/export display adjustment.

PreMame, known cases, and troubleshooting

Troubleshooting

- No RAM patterns found: try a different block, use all-block recording, or switch workflow.
- The same score candidate keeps coming back for both players: check whether the game swaps live and parked score addresses between turns.
- Layout displays correct values only after death: keep it as a useful clue, but prefer the address that updates live while playing.
- More than one winner remains: generate a probe layout and verify the remaining candidates visually.
- Generated layout shows labels but no scores: the export may still contain an old broken Lua callback. Regenerate the layout after updating the scanner to the fixed builder.
- Console game keeps using the shared default layout: check the exported filename and use PreMame if you want separate console .lay files.

Recommended working rule

If the scanner gives you a clean address winner, treat that as the strongest signal. If the mode is still ambiguous, keep testing live in a layout until the visual result tells you which decode is correct. When in doubt, trust the game screen more than the raw candidate list.

MAME Score Scanner

Ready to try it

Start with the scanner, then finish in the builder

Download the current guide, use the scanner to discover or verify the score data, then open the online Layout Builder if you want to import that hi-score layout and dress it up with bezel artwork or custom digit styles.

Support MameDuo

Optional support page

Enjoyed the showcase?

If this project sparked ideas or helped you decide what to try next, you can support future updates here.

Buy Me a Coffee:

https://buymeacoffee.com/mameduo

Scan to open the support page

Thank you for using MameDuo projects, guides, and downloads.