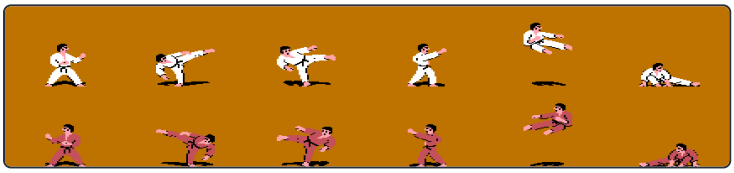
PROJECT SUMMARY · OCTOBER 2026 · ROM BUILDER V0.9

International Karate for the Atari Lynx

A one-to-one port of the 1986 Atari 8-bit classic, rebuilt from a reverse-engineered original



The finished Lynx version in demo mode: Sydney Harbour, the referee calls BEGIN, the original title line runs along the bottom.



Fighter frames decoded from the original disk by the analysis tools (Atari resolution). The Lynx build converts the same data automatically.

At a glance

Original	International Karate, System 3, 1986 (Atari 8-bit)
Design & code	Archer Maclean
Music	Rob Hubbard
Target	Atari Lynx, 65C02 at 4 MHz, 64 KB RAM, 160 × 102 pixels
Engine	Rewritten routine by routine from the original 6502 code, first in C, since v0.8 entirely in 6502 assembler
Assets	Read from the user's own disk image when the game is built
Backgrounds	All 7 scenes with animated details, a new one fading in after every bonus round
ROM	252,992 bytes (enhanced build)
Sound	Rob Hubbard's music and the original effects, in stereo on the Lynx II
Modes	50 Hz Classic like the PAL original, 60 Hz Speed (NTSC tempo) or 60 Hz Smooth: original speed with 30 instead of 10 poses per second
Players	1 against the computer, or 2 on two Lynx linked by a ComLynx cable
Extras	Move overview in the pause, picture turned for left-handed play, high scores kept in the cartridge EEPROM, soft edges as an option
ROM Builder	Windows tool and Linux command line, v0.9: builds the Lynx version from International Karate or World Karate Championship

Development	15–23 October 2026, by Claude in Claude Code, directed and play-tested by the project's author

Overview

This is a faithful port of *International Karate* from the Atari 8-bit computers to the Atari Lynx handheld. The aim was a one-to-one conversion "with everything": the original fighting engine, the computer opponent, the referee and scoring, both bonus rounds, the seven backgrounds, the high score table, the music and the sound effects. The two-player mode followed: on the Lynx it needs two handhelds connected by a ComLynx cable. Some additions go beyond the original: a Smooth mode with two new in-between frames for every step of the fighters, living backgrounds with small animated details in every scene, and since version 0.9 an overview of all moves in the pause, a turned picture for left-handed players, a high score table that is kept in the cartridge and, as an option, soft edges for the fighters.

The game logic was rewritten in C, following the original 6502 routines step by step, and later converted entirely to 6502 assembler. Graphics, palette, game tables, music data and sound samples are not redrawn or re-recorded. They are read from the user's own disk image every time the game is built. No original data is stored in the source code or in the tool.

The **ROM Builder**, a single Windows program (and a command-line version for Linux), does exactly that for everybody: it reads your own copy of the game, recognises the release, converts the data and compiles the Lynx cartridge image on your PC. It also works with *World Karate Championship* (Epyx, 1986), the US release of the game.

AI vs. human: who did what

This port was made with AI, and that is no secret. Claude, Anthropic's AI model, running in Claude Code on the author's Windows PC, wrote every line of code in this project. But a game is more than its code, and an AI on its own would not have made this one. This is how the work was shared.

The author

- **The idea and the goal:** International Karate on the Lynx, one to one "with everything". Claude's first plan was to redraw the graphics; the author insisted on the original data, and so the converter came about that reads it from the player's own disk.
- **The direction:** what came next, one step at a time: two players over the cable, the 60 Hz mode a Lynx owner had asked for in a forum, smoother fighters, more life in the scenery, the rewrite in assembler, four of the seven ideas for version 0.9, and soft edges that can be switched off again.
- **The questions that opened new ways:** can two Lynx be tested on one PC (the double emulator)? Do the colours match the original (the new colour model)? Must the edges of the fighters be hard (the soft edges)? And won't a soft edge look wrong where the two fighters overlap?

Claude (AI)

- **The analysis:** disassembled the original into a listing that reassembles to an identical binary, worked out every routine and wrote the mechanics specification.
- **All the code:** the Lynx game (first in C, now about 10,500 lines of assembler), the data converter, the ROM Builder for Windows and Linux, the tools for the in-between frames and the trailer.
- **The technical solutions:** colours that change from line to line, fighter frames on the cartridge, the turned picture, soft edges without transparency – and proposals of its own, such as the seven ideas for version 0.9.
- **The drafts and the corrections:** the in-between frames, the animated details in the scenery, and every change the author asked for.

- **The eye:** played and watched every build and said what was wrong: palette flicker, a speech bubble without transparency, a somersault that was too slow in the dodging round, a boat that sailed too fast, a caravan walking on the Sphinx' paws. Went through all 100 new in-between frames one by one and said what had to change in about 60 of them.
- **The no:** bigger fighters, swaying branches, a sun that only changed its colours and a heat haze over the desert were tried and dropped, because they did not fit the game.
- **The players:** a tool that anybody can use with their own copy, tested against every release, and the contact with Lynx players in the forums.
- **The tests:** the double emulator, the Lua scripts, frame-by-frame comparisons with the original and with earlier versions, the timing of every frame.
- **The paperwork:** this summary, the website, the README, the screenshots, the videos and the trailer.

> **In numbers:** from 15 to 23 October the author wrote 212 prompts for this project, about 4,800 words in all.

The AI did the bulk of the work, and it worked fast. But it did not know what this game should feel like on a Lynx. Many first attempts were technically correct and still wrong – a foot missing in an in-between frame, a kick that did not turn far enough, a caravan in front of the palm trunks instead of behind them. The tests show that the code does what it was meant to do; whether it is what it should be, only someone who plays the game can tell. The ideas, the taste, the patience to look at every frame once more and the decision what to leave out came from a human.

And the most important human part is forty years old: Archer Maclean's game design and graphics and Rob Hubbard's music. The port does not redraw or re-record any of it – it carries their work to another machine.

Tools

Reverse engineering	REA (rea-agents 6.5.0) driving Ghidra 12.1.4 for a first inventory of the 6502 code (142 procedures). For the detailed work, a Python toolchain written for this project: ATR disk image reader, Atari executable loader, 6502 disassembler emitting ca65 source, byte-for-byte round-trip check, call-graph generator and a hint system for names and comments.
Lynx development	cc65 suite: ca65 assembler and ld65 linker (the C compiler and the TGI graphics library built the versions up to 0.7); sp65 for sprite packing; sim65 to run 6502 code on a simulated CPU.
Build & conversion	Python 3 with Pillow and NumPy.
ROM Builder	PyInstaller (one .exe with Python, NumPy and the needed part of cc65); a small Atari 8-bit emulator in Python that runs the loaders of the different releases.
In-between frames	Python tools written for this project: nine drafting methods (from simple shape morphing to rotating the whole body or single limbs with RotSprite-style pixel-art rotation), an editable sheet with all frame pairs, a checker for stray pixels and a script with the corrections for every single frame.
Testing & recording	Felix Lynx emulator 0.6.4 with built-in Lua scripting; ffmpeg for videos; for the two-player mode a double emulator built from Felix's source code with Visual Studio 2022 (two Lynx on one ComLynx cable).

Trailer	Python with NumPy, SciPy and Pillow, ffmpeg; public-domain photos of the Lynx I and II (Evan-Amos, Wikimedia Commons); the fonts Bebas Neue and Oswald (SIL Open Font License).

How the work was done

1. **Understanding the original.** The game is a single executable of about 40 KB. It was disassembled into a listing of about 9,000 lines that reassembles to an identical binary. With that guarantee, routines, tables and variables could be renamed and annotated freely without breaking anything. The analysis was split into topic notes (moves and animation, hits and scoring, computer opponent, game flow, status display, bonus rounds, sound) and condensed into a mechanics specification, which became the blueprint for the port.
2. **Build pipeline first.** Before any game code was written, a minimal Lynx program, a reliable build script and the emulator were set up. Input handling and flicker-free double-buffered display came next.
3. **Data converter.** A Python converter reads the disk image and generates C and assembler sources: the backgrounds, 53 fighter animation frames, the status display, the referee and his speech bubbles, the bonus-round objects, all game tables, the music data and the six sound samples. The first plan had been to redraw the graphics. When the author asked for a true one-to-one port, the converter became the bridge between the original disk and the Lynx build.
4. **Rebuilding the engine.** Routine by routine in C: the move engine with its animation tables, joystick mapping relative to the fighter's facing direction, automatic blocking, hit detection and scoring, clock and time bonus, the computer opponent with its difficulty levels, demo mode, both bonus rounds, game over and the high score table with name entry. Source comments cite the original addresses, so every part can be traced back to the listing.
5. **Checking against the original.** The music player ran in sim65 and its output was compared with a register dump of the original for all 13,516 frames of the tune: identical. Sound effects were checked sample by sample in WAV recordings from the emulator. Dropped frames, interrupt timing and palette switches were measured with Lua scripts inside the emulator.
6. **Improving the picture with the author's feedback.** An enhanced build improved the picture quality. Every change was checked with recordings: the emulator writes out each displayed frame with its palette, and a script turns that into a pixel-sharp video with synchronised sound. The author's comments on these videos (palette flicker, a speech bubble without transparency, a score overlapping the bubble, the stars over a knocked-down fighter, the missing credits line) drove the next rounds of work.
7. **All seven backgrounds.** The first version was based on a single-file release that only contains the Sydney scene. The disk versions load one of seven scenes after every bonus round. That loader was analysed and rebuilt, including its side effects on the game flow. The author then asked whether the colours matched the original. They were checked against screenshots of the original for every scene, which led to a much better colour model (see below). Later the colours were refined once more, with palette changes from line to line.
8. **The ROM Builder.** A Windows program for everybody who owns the game. It recognises each release by a fingerprint over exactly the bytes the converter reads, and at the end of every build it is tested against 42 different releases and file formats. A final audit unpacks the finished .exe completely and checks that it contains no data of the original game.

9. **Two players over the cable.** The original's two-player mode needs two joysticks, and the Lynx has only one D-pad – but it has the ComLynx port. The author asked whether this could be tested on one PC. Felix emulates a single Lynx, so Claude built a small double emulator from Felix's source code: two Lynx running side by side on one simulated cable, driven by scripted inputs. With it the cable protocol, the two-player game logic from the original listing and the synchronisation of both handhelds were developed and tested.

10. **60 Hz and smoother fighters.** A Lynx owner in a forum asked for the faster speed of the game on NTSC machines, so the frame rate can now be switched in the pause. Then the author wanted smoother fighters: the original shows a new pose only every fourth frame, 10 per second. The new Smooth mode runs at 60 Hz with the timing of World Karate Championship on NTSC machines, which keeps the original speed, and adds two in-between frames to every step: 30 poses per second. Claude drafted the 100 new frames with several methods, and the author reviewed them one by one – missing feet, odd heads, legs that bent the wrong way, a kick that had to turn a little further. About 60 frames got corrections of their own until every movement looked right.

11. **Living backgrounds.** The author asked for more life in the scenery, one location at a time. The pictures stay the original ones; small animated details are drawn into them: water that ripples, clouds that drift behind the temple and the skyscrapers, cherry blossoms, a shimmering sun, window lights that go on and off, and now and then a sailboat, a gull, a flock of birds, a ferry, a hovering helicopter, a camel caravan, or cars and a train on the Harbour Bridge. Every idea was checked in a recording first, and the author's verdict decided: swaying branches looked unnatural, a sun that only changed its colours did not convince, and the heat haze over the desert went again. The boat had to sail slower, the caravan lower – at first it walked on the Sphinx' paws – and behind the palm trunks instead of in front of them, until it was right.

12. **Everything in assembler.** The C version had used up the RAM (300 bytes free) and spent time on the compiler's parameter and pointer handling. So the whole Lynx program was rewritten in 6502 assembler, module by module. In the double emulator, test builds of both versions ran the demo over all backgrounds, one-player games with random input, both bonus rounds and two-player matches, and the pictures, the game state and the writes to the sound registers were compared frame by frame: identical. The code shrank from 25,325 to 14,367 bytes, 13,017 bytes of RAM became free, and a frame takes about a fifth less time.

13. **A trailer, made in code.** A 90-second trailer, cut to the original music, without a video editor: the double emulator recorded long runs with a switch that logs the fighters' moves and positions in every frame, a script picked a hit for every move and the moment in each scene when its animated details are visible, and the rest – the Lynx's pixel grid, public-domain photos of both Lynx models with the game on their screens, titles, the sound effects of the original on every hit – was put together frame by frame in Python.

14. **Version 0.9: for the player.** Asked for ideas for the next version, Claude proposed seven; the author picked four: an overview of all moves in the pause, the picture turned round for left-handed players, saving in the cartridge EEPROM and a sailboat in Rio. The turned picture was compared frame by frame with the normal one, saving was tested with emulator runs that switch the Lynx off and on again, and long demo runs in every scene were compared with version 0.8: apart from the new sailboat, every frame is the same. At the author's request a fifth addition followed: soft edges for the fighters, switched with A in the pause. With hard edges, every frame is still the same as before.

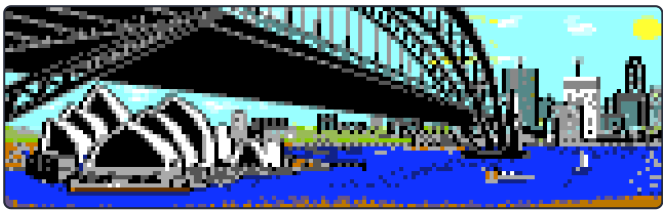
Technical challenges

Screen size: 192 lines into 102

The Atari picture is 160 pixels wide and up to 192 lines high; the Lynx screen is 160 × 102. The width matches, but everything has to be halved vertically. Dropping every second line destroys thin details, so the background combines each pair of Atari lines into one Lynx row and gets its own set of 16 colours for that, chosen by clustering in a perceptual colour space. The fighters are halved the same way, with mixed colours where two different colours meet.



Atari: the background as stored on the disk, 160 × 98 lines.



Lynx: 160 × 49 rows, line pairs blended, colours changed from line to line.

> Pixel-exact fighters at full height, and later at double width with smoothed contours, were also built as experiments. Next to the halved background they looked too coarse, so the final version keeps fighters and background on the same grid.

Colours: 128 per line vs. 16 per screen

The original changes colours from line to line with display-list interrupts and picks from 128 colours; the Lynx shows 16 colours at a time out of 4,096. Halving the picture height adds to the problem: blending each pair of Atari lines creates 18 to 51 different mixed tones per scene. A single Lynx row, however, needs only 7 to 19 of them, and neighbouring rows share almost all. So the enhanced build changes colours from line to line as well. For every row the converter plans which palette entries to reload for the next one, and it only reloads entries that the current row does not show. That makes the timing uncritical: the change may happen anywhere in the row. A timer interrupt carries out these loads, and the converter groups them so that 6 to 16 interrupts per frame are enough. The scenery keeps almost all of its mixed tones; most of the remaining difference to the original screenshots is the Lynx's 4,096-colour palette itself. In the busiest scene, Mount Fuji, up to 49 colours are on screen at the same time, more than the 18 of the original picture (45 before the shimmering sun of version 0.8 added its own tones). Status lines, scenery and fighting area still start from palettes of their own, and when the scene changes, the new colours take effect in exactly the frame that first shows the new picture.

Seven scenes and their raster effects

Each scene has three small interrupt routines of its own: colour changes, sky gradients over three or four lines, even a player-missile object that is moved halfway down the picture. The converter runs these routines in a tiny 6502 interpreter and so knows on which line which colour changes. Compared with a screenshot of the original for every scene, the model now matches pixel for pixel, and 20 more Atari colours were calibrated from those screenshots. The sunset over New York, the red temple roofs and the blue lake below Mount Fuji only appeared after this step.

Graphics model

The Atari draws the fighters as characters in a 16 × 12 character box and the bonus-round objects as hardware player-missile graphics with collision registers. On the Lynx everything becomes a Suzy sprite, run-length packed to save memory, and collisions are computed in software by decoding the packed sprite at the pixel in question.

Sound: POKEY to Mikey

Rob Hubbard's music player was transcribed one to one and its POKEY settings mapped onto Mikey's channels. One subtle trap: writing Mikey's channel control register restarts the channel's timer, and until only the reload value was updated on pitch changes, the vibrato sounded about 3 % flat. The six original sound effects are 4-bit samples at 3.95 kHz, played through Mikey's DAC from a timer interrupt. For a while the enhanced build streamed them directly from the cartridge to free 4.3 KB of RAM; since version 0.7 the fighter frames live on the cartridge instead (see below), and the samples are back in RAM.

Memory

The Lynx has 64 KB of RAM, and two 8 KB frame buffers come out of that. Code, tables, packed sprites, music and the stack have to fit into the rest. The seven backgrounds and their colours stay on the cartridge; at a scene change the new picture is read into one shared buffer within about two frames. The colours of the next scene are read straight from the cartridge in the vertical blank before its first frame, because there is no room to keep them twice. The two-player mode needed another 2.7 KB. A measurement showed that the game never uses more than 14 bytes of its 1 KB C stack, because the compiler keeps local variables static, so the stack shrank to 256 bytes. The speech bubbles are now stored as one empty bubble plus the text of each message, as the original does it; a 512-byte frequency table for the music is calculated instead, with identical results for all 256 values; and the cable code was written in assembler. That left 716 bytes. The Smooth mode needed far more: the 100 new frames alone would not fit. So version 0.7 moved all fighter frames, the 53 original ones and the new ones, onto the cartridge as well. The assembler version of 0.8 freed 13 KB; the living backgrounds – sprites, patches, occluders, the copies of the colour tables for fading – use most of it. The additions of version 0.9 – move overview, turned picture, saving and soft edges – took another 3.3 KB, and 0.6 KB are still free.

A toolchain bug

The Lynx interrupt handler that ships with cc65 acknowledges all pending interrupts after its handlers have run. An interrupt that arrives in the meantime is lost: about one vertical blank in a hundred while sound effects were playing, visible as occasional palette flicker. The enhanced build has its own interrupt entry that acknowledges only what was pending on entry. Over 3,000 test frames, missed palette switches dropped from 30 to 0.

Colour changes and drawing time

Every interrupt halts Suzy, the graphics chip, for as long as it runs, so the extra colour changes cost drawing time. A profile of every frame, measured inside the emulator, showed where the time really went: not into Suzy's drawing, but into the C code that prepares a frame. Almost a quarter of every frame was spent rebuilding the score lines, which change about once a second. They are now only rebuilt when something changes, the floor is drawn together with the background, the scenery above the fighting area is only redrawn after a scene change, and the code that places the fighters and draws the score lines, the speech bubbles and the high score table was rewritten in assembler. A frame now takes about half as long to prepare. In a one-minute test with all sound effects playing no frame is dropped any more (1.7 % before); only loading a new scene and the very first high score table after switching on still take a little longer than one frame. Long test runs, including both bonus rounds, were compared frame by frame with the old drawing code: every frame is identical. One emulator detail also had to be handled: Felix does not raise an interrupt again if it arrived while another one was being handled, so the interrupt entry now looks for new requests before it returns. Later, slow-motion jumps in the dodging round showed one more bottleneck: while an object was in the air, every second frame was dropped. The Atari detects these collisions in hardware; the Lynx version checked them pixel by pixel and rebuilt the object graphics in slow C code. Both were rewritten in assembler with identical results, and every frame is drawn again.

In-between frames at 60 Hz

The fighters use 53 animation frames in a fixed box of 64 × 96 Atari pixels; the 35 moves step through 72 different pairs of frames. For each pair the Smooth mode has two new frames, at one third and two thirds of the way. The difficulty is timing: when a new pose appears, the game does not know the next one yet – the player may hold the move, reverse it or start another. So the in-betweens always lead from the previous pose to the current one: for two frames the first, for two frames the second, then the pose itself. Every pose is still shown, only 67 milliseconds later, and nothing is ever guessed wrong. When a fighter turns around or moves far in a single step, the new pose follows directly, as in the original.

All 153 frames together are far too large for the RAM, so they now lie on the cartridge, one 1 KB block each, and a frame is copied into a buffer of its fighter whenever it changes. The copy loop handles eight bytes per pass and was tuned to 12 cycles per byte, because in Smooth mode both fighters may need a new frame in the same 1/60 second. If the score lines change in that very frame, they are redrawn one frame later, so that every in-between frame stays on screen for its full 1/30 second. The game logic itself is untouched: over more than 5,500 pose changes, Classic, Speed and Smooth produced exactly the same sequence, and the Classic mode was compared frame by frame with the version before: identical.

Animation on a picture that is not redrawn

To save time, the scenery above the fighting area is only drawn after a scene change; every other frame starts from what the frame buffer still holds from two frames earlier. Anything that moves there must clean up after itself. The converter cuts every moving object out of the picture, fills the hole with neighbouring pixels and stores the object as a sprite; a small patch of the cleaned picture is drawn first, and only in the two frames after the object has moved. Objects that pass behind the temple, the Statue of Liberty or a palm trunk get an "occluder": the picture's own pixels, drawn once more after the object. Cars and trains on the bridge are single-pixel sprites that give the pixels behind them their original colour back. All of them may only use colours that their picture row already shows, because the palette changes from line to line. One trap: one-bit sprites load a palette byte of their own, and every sprite drawn after them inherited it – white pixels appeared in the cypresses until a zero-size sprite reset the palette.

Fading with changing palettes

A fade on the Lynx means rewriting the colour registers – but here the colours change from line to line during the picture. So the tables for the line interrupts are kept twice: the original and a copy in which every colour is scaled by the current brightness. A new scene takes over the current brightness when its colours are read from the cartridge, so it starts dark and brightens in eight frames. Fading out only changes colours and does not draw a new picture: right after the high score table the two frame buffers differ, and redrawing them made the picture flicker. The same tables dim the picture during the pause, while the status lines with the pause text keep their full colours.

Many releases, one builder

The game exists in many forms: single files, boot disks with their own loaders, a slightly different build of the program on the original disk, a 130XE version that keeps the backgrounds packed in extra memory, a cartridge with the disk inside, cassettes, and a copy-protected original disk whose loader reads sectors that exist twice on the track. The ROM Builder runs each loader on a small emulated Atari until the game starts, reads the backgrounds from wherever the release keeps them and tells the user which release it found and what that means for the Lynx version.

Small hardware details

A transparent sprite pixel on the Lynx is colour 0 after the sprite's pen mapping, not pen 0 itself. This mattered for the referee's speech bubble, which overlaps the water.

Controls

The Atari version uses a joystick with one fire button plus keyboard keys. On the Lynx the D-pad is the joystick, A and B are fire, Option 1 starts a game – with a second Lynx on the ComLynx cable a two-player match, as SELECT did on the Atari – Option 2 switches the music on and off, and Pause pauses. While paused, left and right on the D-pad choose the mode: 50 Hz Classic, 60 Hz Speed or 60 Hz Smooth; down shows an overview of all moves, Option 2 turns the picture round for left-handed players, and A switches the edges of the fighters between hard and soft.

Two Lynx, one game

Every frame, the Lynx that started the match sends its buttons over the cable, and the other one answers at once from its serial interrupt with its own. A frame is only computed when both are there, so both handhelds compute exactly the same game. Random numbers come from a common seed, the computer opponent no longer takes its chance from the raster line, the high score table is copied at the start, and every frame advances the game by exactly one tick on both. The invitation also carries the mode, and switching it in the shared pause changes both handhelds in the same frame. The invitation also carries an identifier: a Lynx whose ROM would compute the game differently, for example one built from a release with a single background instead of seven, does not join. In the double emulator the complete game state of both handhelds was compared after every frame: through whole matches, bonus rounds, the winner's solo game and the name entry, also with the two screens out of phase, with clocks running 200 ppm apart, with pause and with the cable pulled. It was identical on both every time. The emulated serial port needed two corrections first: Felix sent bytes eight times too fast, and its receiver lost some bytes that were sent back to back. On real hardware the mode has not been tested yet.

Turning the picture round

Lynx games for left-handed players turn the picture with a bit in the display hardware. The Felix emulator does not support that bit, and the colours that change from line to line would have to be turned round anyway, so the game turns the picture itself, which also works in emulators without that bit. Before Suzy draws a sprite, its two mirror flags are switched and its position is reflected; Suzy then draws the same pixels from the opposite corner, exactly in their turned place and at no extra cost. Sprites that never move – score lines, background, floor, referee – are turned once when the player switches, and the moving details of the scenery stay in turned screen positions for as long as the picture is turned. For every scene the converter writes a second table of colour changes that runs from the bottom of the picture to the top, and it simulates both tables line by line when the game is built. The D-pad is turned as well, before the buttons go over the cable, so a left-handed and a right-handed player can play each other. In every scene, 6,000 frames of the demo, turned back by a script, are identical to the normal picture pixel for pixel. Outside scene changes, which take up to three frames longer, the turned picture drops no frames.

Saving without a battery

Lynx cartridges can carry a small EEPROM of 128 bytes, and flash cartridges and emulators provide one. It is reached through the cartridge's address counter and an audio pin, and every bit is clocked by hand. The game keeps 42 bytes there: an identifier, the mode, the orientation, the six places of the high score table and a checksum. Writing a word keeps the chip busy for up to 10 milliseconds, so the game never waits for it: once per frame it sends at most one changed word and asks in the next frame whether the chip has finished. Without an EEPROM every word reads as $FFFF, the identifier does not match, and the game starts with an empty table as before. The tests found two problems. The game itself kept the audio pin high while it read fighter frames from the cartridge, so the chip took every such read for an erase command and never finished; the pin now stays low outside the EEPROM routines. And Felix leaves part of its EEPROM state uninitialised, so in about every second start the chip stays silent. The test emulator was corrected; in Felix 0.6.4 itself saving works only in the other starts.

Soft edges without transparency

The Atari draws the fighters with twice the vertical resolution of the Lynx screen, so the port mixes every pair of lines into one, and inside the fighters this has always produced in-between colours. At the outline, where a fighter pixel meets an empty one, the pixel stayed hard. Suzy knows no transparency, so a soft edge has to be a colour of its own: the fighter's colour mixed with the floor. The fighters stand in front of a floor of a single colour, so four mixed colours per scene are enough – black, skin, white and red, each with the floor – and the line interrupts load them below the picture into colour slots that only the picture uses higher up. The edge pixels become a small sprite of their own, drawn before everything else on the floor; whatever is drawn later covers them, so where the fighters overlap, the edge simply stays hard. In the fighter picture itself the edge pixels keep a pen that is transparent but still counts for the collision test of the bonus round. Both versions of every frame are on the cartridge, so with hard edges the game loads and draws exactly as before. A test compares the two settings frame by frame: every one of more than two million differing pixels is the expected mixed colour or covered. To keep enough time for the extra sprites, the status lines are now only redrawn after they change, which saves more than a millisecond per frame.

Testing without hardware

Everything was tested in Felix. Its Lua scripting made it possible to count dropped frames, switch sound effects on in the demo, jump straight into the bonus rounds or into a given scene, log the exact frame of a scene change, record the sound and capture the picture from inside the emulator, independent of what is on the PC screen. The two-player mode was tested in the double emulator described above. The Smooth mode was checked the same way: in a long demo run every in-between frame belonged to its pair of poses and stayed on screen for exactly two frames, and in two-player matches both handhelds showed the same poses and in-between frames, also after switching modes in the shared pause. 98 of the 100 new frames appeared in the tests; the other two belong to a step at which the fighter turns around, and turns never get in-betweens. Only 0.06 % of the frames in Smooth mode come late, all of them at the start of a fight or at a scene change, never during a movement. Felix runs about 1.3–1.6 % faster than a real Lynx, and the recording script corrects for that.

The finished Lynx game

The enhanced build (“Farben” is German for colours) contains the complete game: one player against the computer, or two players on two Lynx, in three modes. The ROM Builder produces this build from your own copy.



A flying kick meets its target. The fighters keep the original animation frames and timing.



FULL POINT: score pop-up, referee bubble and the stars over the knocked-down fighter.



The seven backgrounds on the Lynx: Sydney, New York, Rio de Janeiro, an East Asian temple, the pyramids, the Parthenon and Mount Fuji, each with its own animated details. A new one fades in after every bonus round.



Up close: a sailboat off Athens passing behind the cypress, a helicopter over Manhattan, a caravan behind the palm trunk, traffic on the Harbour Bridge, the lake below Mount Fuji and the sun behind the blossoms at the temple.



Bonus round 1: break the tiles. The faster the strike after GO, the more tiles break.



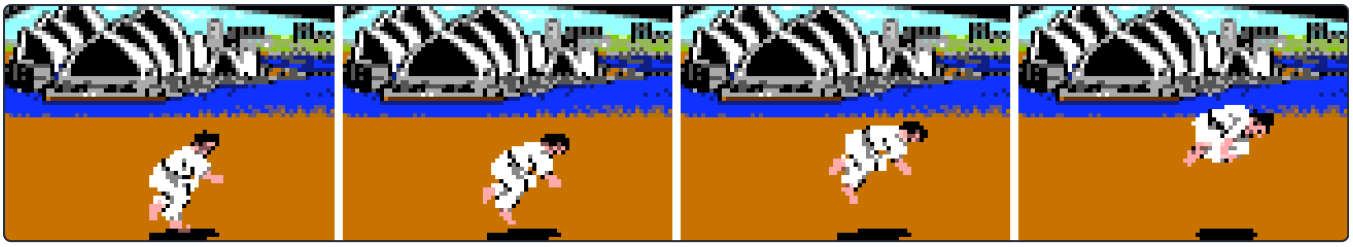
Bonus round 2: knives, spears and spinning objects fly in from both sides and have to be dodged.



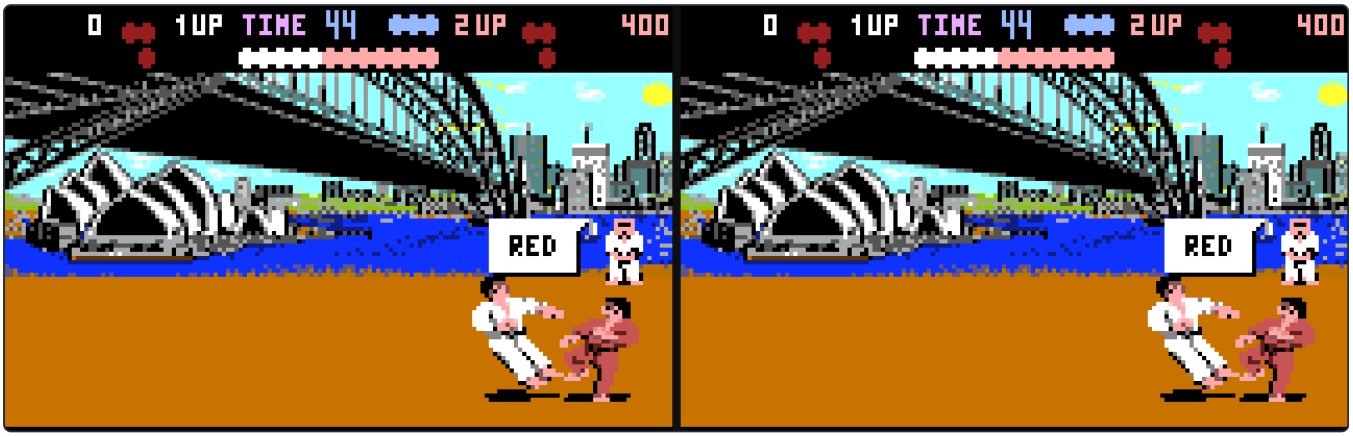
The high score table with six places, belts and three-letter names.



In the demo the bottom line alternates between the title and “Game essence by Archer Maclean”.



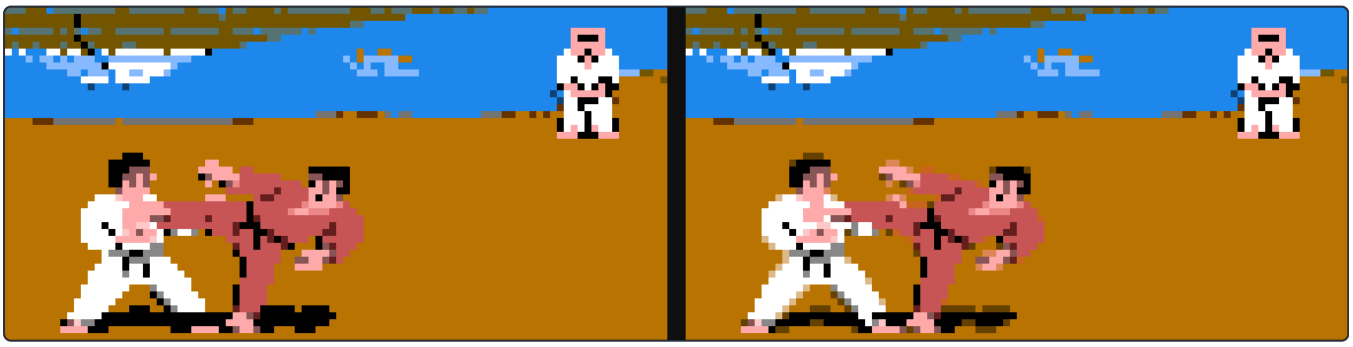
60 Hz Smooth: a somersault from the original pose on the left to the next one on the right, with the two new in-between frames shown for 1/30 second each.



Two players, two Lynx: the same match on both screens. The status line shows both players, the fights won and the tug-of-war bar; Red has just scored.



The pause in version 0.9: down on the D-pad shows all 16 moves, on the left without and on the right with fire, each as the pose it starts (left picture); Option 2 turns the picture round for left-handed players (right picture). The status line shows the edge setting and the mode.



Soft edges (A in the pause): on the left the original hard outline, on the right each edge pixel blended with the floor – at the heads, arms and shadows. Where the fighters overlap, the edge stays hard.

What is in the game

- All seven backgrounds of the disk version – Sydney, New York, Rio de Janeiro, an East Asian temple, the pyramids, the Parthenon and Mount Fuji. A new one appears after every bonus round, in the demo after every

fourth fight, and every new game starts in Sydney again. Releases that contain only one background keep it, as on the Atari.

- Fights for half and full points against the computer opponent, with speech bubbles (BEGIN, RED, WHITE, HALF POINT, FULL POINT), score pop-ups, a 30-second clock and a time bonus.
- A computer opponent that gets harder as the levels go up; belt colours by score: white, yellow, green, purple, brown, black (from 40,000 points).
- The original cycle: three bouts, the tile-breaking round (100 points per tile), four more bouts, the dodging round (8, 15 or 20 objects, 2,000 points if none of them hits). One lost bout ends the game.
- Two players on two Lynx linked by a ComLynx cable: 60-second fights decided by the tug-of-war bar, a best-of-three match with both bonus rounds in between; the winner then carries on alone against the computer.
- Game over and the high score table with name entry; the demo mode with the original title and credit lines.
- Rob Hubbard's music and the original sound effects, as on the Atari.
- Three modes, chosen in the pause: 50 Hz Classic like the PAL original (the default), 60 Hz Speed like the European version on an NTSC Atari (20 % faster), and 60 Hz Smooth at the original speed with two new in-between frames for every step of the fighters.
- Living backgrounds: rippling water, drifting clouds, a shimmering sun, window lights, cherry blossoms, the Sphinx opening its eyes, and passers-by from a sailboat to a camel caravan; scenes fade out and in, and the pause dims the picture.
- An overview of all 16 moves in the pause: for every direction of the D-pad the pose it starts, without and with fire.
- Left-handed play: picture and D-pad turned round, chosen in the pause; it also works in two-player matches against a right-handed player.
- The high score table, the mode, the orientation and the edge setting are kept in the cartridge EEPROM (flash cartridges and emulators with EEPROM support).
- Soft edges for the fighters as an option, switched in the pause: where a fighter stands in front of the floor, each edge pixel blends the fighter's colour with the floor; where the fighters overlap, the edge stays hard.
- Stereo sound on the Lynx II: the music voices spread across the stereo field, the hits coming from the side of the fighter.

30	49	13,516	0	42
fighter poses per second in Smooth mode (10 in the original)	colours on screen at once (16 per Lynx palette)	music frames identical to the original	pixels different from the original screenshots in all 7 scenes	releases and file formats tested with the ROM Builder

About 10,500 lines of assembler make up the Lynx game, 2,800 lines of Python the data converter, 2,600 lines the ROM Builder, 2,500 lines the tools for the in-between frames and 2,600 lines the trailer. A plain build without the colour enhancements (42,571 bytes, Sydney only) is kept as well. Use the ROM Builder only with your own copy of the game and do not share the built .lnx file.

International Karate © 1986 System 3, World Karate Championship © 1986 Epyx. Game design and programming: Archer Maclean. Music: Rob Hubbard. Lynx port: unofficial, non-commercial fan project, developed with Claude Code; not affiliated with or endorsed by the rights holders. Screenshots: recorded inside the Felix emulator from the finished Lynx build; the two-player picture, the Smooth sequence, the living backgrounds, the pause pictures and the edge comparison in the double emulator built from Felix's source code; original graphics decoded by the project's analysis tools.