VIDEOGAMES / HARDWARE / DIGITAL CULTURE

Retromind

01

13 NOVEMBER

2026



SPECIAL / CONVERSIONS AND NEW LIVES

OUTRUN AGA / INSECTOR X / TIME PILOT

CLASSIC GAMES, NEW MACHINES

Arcades, code and creativity: classics find new ways forward.

32 **BEHIND THE PATCHES**
Tools and communities keeping games alive

41 **GOLDENEYE 007**
From the archive: the value of code

EXPERIMENTAL EDITORIAL PROJECT WITH AI AGENTS

50 PAGES / ENGLISH

WELCOME / RETROMIND IN ENGLISH

A GUIDE TO RETROMIND.

An English-language doorway into an Italian site about games, machines and the people who keep them alive.

Retromind is a predominantly Italian-language website about classic videogames, home computers and consoles, and the work that continues around them. Its focus is not limited to remembering old releases. New homebrew games, fan conversions, translations, technical discoveries and preservation projects show how these platforms remain places for making, investigating and playing.

The site follows the 8-bit and 16-bit computer and console eras, the arrival of 32-bit systems, and later generations when a story connects with that wider history. An Amiga conversion, a Saturn translation or a reconstructed Nintendo 64 program can illuminate different parts of the same subject: what games do, how their machines work, and how communities make them accessible today.

News introduces recent developments and links to the original reporting. Features examine methods, requirements and historical context. Blog essays consider tools, documentation and the people involved. Archive selections bring earlier articles back into the conversation, retaining their original dates so that a past report is not mistaken for a new announcement.

This English edition presents the complete selection from the Italian magazine dated 13 November 2026, with the same editorial illustrations and a newly added introduction. It is a translation and editorial adaptation of that issue, not an update of its news. The original selection closed on 12 November; subsequent developments are outside its scope. Links lead to the cited sources and to original Retromind articles, which are generally in Italian.

Retromind Magazine is also an experimental publishing project. AI agents select, adapt and lay out material according to editorial and production rules established by people. Human involvement in defining those rules is distinct from reviewing each issue. Any actual human revision is declared when it takes place, and generated or reworked illustrations are identified in their captions. This edition follows the same principle of transparency.

Visit Retromind: original articles, sources and the magazine archive.

INSIDE THIS ISSUE.

01 / 13 NOVEMBER 2026 / ENGLISH EDITION



The high-score table, with columns for names, routes and records. Screenshot by RoarTJ, published in AmigaGuru's review.

09 OUTRUN AGA

RetroDNA's conversion opens our special on the new lives of arcade games.

05 NEWS

The week across computers and consoles

09 FEATURES

Eight articles on the new lives of classics

32 BLOG

Patches, hardware and shared decisions

41 FROM THE ARCHIVE

GoldenEye 007 and conversion engineering

48 THE PROJECT

Transparency, credits and further reading

02 / Preface: a guide to Retromind.

04 / Editorial: machines change, ideas continue.

49 / Continue exploring.

START / ISSUE 01

MACHINES CHANGE. IDEAS CONTINUE.

A weekly magazine following classic videogames as they continue to evolve.

Videogame history is full of changes of machine. An idea born in an arcade finds a home on a domestic computer; an engine is studied, rewritten or adapted; a community carries on working where the market stopped. Retromind's first issue begins with these transitions.

OutRun AGA, Insector X and Time Pilot offer three entrances into the same workshop. Curiosity about the result sits alongside curiosity about the method: code, memory, pictures and sound reveal what bringing a game to another architecture means. Alongside conversions are Amiga patches, Soundtracker music and documents from an unfinished Psygnosis project.

The blog essays turn towards tools and people. A patch should still be usable tomorrow; hardware requirements must distinguish the machine building the software from the one running it; a decision shared across forums, chat and code should leave an understandable record.

The archive forms part of that journey. We revisit GoldenEye 007 and an essay on arcade conversions, retaining their original dates. The present of classic gaming becomes clearer when placed beside the questions that came before it.

This magazine is itself an experiment. AI agents select and lay out the material according to editorial criteria established by people. The project note explains the method, and captions distinguish illustrations from documents. Links lead back to articles and sources. Enjoy the issue.

Retromind / editorial

This edition translates the Italian issue dated 13 November 2026. Its selection closed on 12 November at 15:53 CET. The news is selected; all chosen features and blog essays are translated in full. The added preface introduces the site to English-speaking readers.

6-12 NOVEMBER / SELECTED NEWS

8 / 16 BIT

The week's developments, with direct links to the original sources.

AcidBoutique: a new music tool for Commodore Amiga

AcidBoutique is new music-production software for Commodore Amiga, inspired by a 303 synthesiser and an 808 drum machine.

11.11.2026

Source: Indieretronews

Aliens NeoPlasma reaches Amstrad Plus in a new unofficial conversion

The ZX Spectrum title comes to Amstrad CPC Plus in a new version made with Devil System and AI assistance.

10.11.2026

Source: Indieretronews

RoboCop Arcade: a new Amiga 500 conversion is in development

Ranx is working on an arcade-style RoboCop conversion for Amiga 500, inspired by Data East's 1988 coin-op.

10.11.2026

Source: Indieretronews

Vallation receives an enhanced ZX Spectrum 128K edition

The Cybernoid-style game from 2016 returns to the ZX Spectrum in an improved version for 128K machines.

10.11.2026

Source: Indieretronews

Cat Simulator 64: a new Commodore 64 game

A light, family-friendly title arrives on Commodore 64, developed by illobo/netmilk with assistance from Claude AI.

10.11.2026

Source: Indieretronews

The Great Giana Sisters comes to ZX Spectrum in a new unofficial conversion

Kirby2000 has released a new unofficial ZX Spectrum conversion of The Great Giana Sisters, bringing the classic 1980s platformer back to 8-bit hardware.

09.11.2026

Source: Indieretronews

Maldita Castilla could be coming to Amiga AGA

MainakeProject is working on an Amiga AGA port of Maldita Castilla, Locomalito's classic indie action platformer.

08.11.2026

Source: Indieretronews

6-12 NOVEMBER / SELECTED NEWS

8 / 16 BIT

The week's developments, with direct links to the original sources.

MSXdev26 #18: Tresure Chest released

A new MSXdev26 release: Tresure Chest, an MSX title presented as a mixture of puzzle and adventure.

08.11.2026

Source: MSXDev

Wipeout 2097/XL comes to Amiga in a fan-made conversion

An unofficial Wipeout 2097/XL port brings the racer to AmigaOS 3 using decompiled original code and requiring modern hardware.

08.11.2026

Source: Indieretronews

1942 arrives on Amiga in a smoothly scrolling homebrew port

Retrodnagames releases a fan-made Commodore Amiga port of 1942, with old-school graphics and scrolling.

08.11.2026

Source: Indieretronews

Fix it Felix Jr arrives on Amstrad CPC Plus

Amstrad GGP releases new homebrew for Amstrad CPC Plus and GX4000 inspired by Fix it Felix Jr.

08.11.2026

Source: Indieretronews

Pinball Illusions: The Vikings table is being brought to Amiga AGA with AI assistance

Pinball Illusions' fourth table, The Vikings, is being unofficially adapted for the AGA Amiga A1200, addressing a historical omission from the original Amiga release.

07.11.2026

Source: Indieretronews

Captain Victor Stormbeard and the Undead Treasure: a new C64 platformer demo

KD Software has released a new playable demo of its Commodore 64 platformer in development, offering a first hands-on taste of the game.

07.11.2026

Source: Indieretronews

6-12 NOVEMBER / SELECTED NEWS

32 / 64 BIT

The week's developments, with direct links to the original sources.

A guide to the Biggoron Sword trading sequence in Zelda: Ocarina of Time

A practical guide to Ocarina of Time's celebrated trading sequence, detailing the steps needed to obtain Biggoron's Sword in the Nintendo 64 classic.

11.11.2026

Source: Nintendo Everything

Ys: Vanished Omen, Saturn and Falcom Classics in focus

A feature on the Saturn version of Ys and the role of Falcom Classics, with historical context and a focus on JVC's 1997 release.

11.11.2026

Source: SEGA SATURN, SHIRO!

Renegade 2: Return to Jacob's Star, a lost DOS sequel resurfaces

Reconstructing a cancelled DOS sequel: Renegade 2: Return to Jacob's Star, the unreleased follow-up to Midnight Software and SSI's 1995 space game.

11.11.2026

Source: Games That Weren't

Examining Hebereke's Popoitto on Sega Saturn

A feature on the Sunsoft and Success puzzle game released for Saturn in Japan and Europe in 1995.

09.11.2026

Source: SEGA SATURN, SHIRO!

Three new English Sega Saturn patches with AI assistance

Three new English Sega Saturn patches have been released as part of the latest wave of fan translations for the console.

06.11.2026

Source: SEGA SATURN, SHIRO!

Saturn: QuadWorld homebrew and new fan translations in SHIRO!'s roundup

SHIRO!'s roundup covers the Sega Saturn homebrew game QuadWorld and new fan translations for the console.

06.11.2026

Source: SEGA SATURN, SHIRO!

6-12 NOVEMBER / SELECTED NEWS

PS2 / XBOX / GAMECUBE AND BEYOND

The week's developments, with direct links to the original sources.

Sega's most influential platformer deserves a major remake

A feature revisits a major Sega platformer from the Dreamcast/GameCube era and argues that it deserves a substantial remake.

09.11.2026

Source: Polygon

RetroTINK 6x CE: a restock announced for 7 November

Mike Chi had announced a RetroTINK 6x CE restock for 7 November at 11:00 New York time. The notice concerned units in stock and ready to ship.

06.11.2026

Source: RetroRGB News

Bethesda marks Skyrim's 15th anniversary with Anniversary Creations

Bethesda celebrates The Elder Scrolls V: Skyrim's 15th anniversary with a new Anniversary Creations bundle, while its community continues supporting the original game from the PS3 and Xbox 360 era.

11.11.2026

Source: Wccftech Gaming

Revisiting NASCAR 14

A return to NASCAR 14, the NASCAR racing simulation that had already established its character on the platforms of its time.

08.11.2026

Source: Old School Gamer Magazine

Mass Effect 3 multiplayer revived through a Legendary Edition mod

ME3Tweaks has released a mod restoring Mass Effect 3's cooperative multiplayer through Legendary Edition.

07.11.2026

Source: Wccftech Gaming

Wobbling Pixels compares RetroTINK 6x CE and Morph 2K

Wobbling Pixels has published a detailed comparison of Mike Chi's and PixelFX's scalers, relevant to players using PS3, Xbox 360 and other older console generations.

06.11.2026

Source: RetroRGB

FEATURES / 12.11.2026

OUTRUN AGA: RETRODNA'S CONVERSION FOR THE AMIGA 1200

A 14 MHz 68020 and 4 MB of Fast RAM: the project recreates Sega's classic with complete routes, music and sound effects playing together. AmigaGuru's review explores its achievements and compromises.



Conceptual illustration recomposed with AI to fit this issue.

An Amiga 1200 with a 14 MHz 68020 processor and an additional 4 MB of Fast RAM: this is the configuration specified for OutRun AGA, RetroDNA's conversion reviewed by AmigaGuru. Its most interesting achievement lies in the experience as a whole: the routes, the transitions between locations, and music and sound effects playing simultaneously, within the constraints of a machine without a CPU accelerator.

The review describes a game running at 320 × 224 pixels, with 32 colours and a frame rate generally around 25 frames per second. These figures explain the project more precisely than calling it a perfect conversion: OutRun AGA develops its own technical solution for recreating, on a home computer, an experience originally built around the capabilities of arcade hardware.

In OutRun, the sensation of driving depends on many elements working together. The road changes gradient and direction, roadside objects grow as they approach, and traffic demands constant adjustments. Junctions turn each run into a route the player chooses. Preserving this visual and physical continuity is essential: a convincing screenshot reveals only part of the work needed to sustain an entire game.

Amiga players also have a specific historical comparison. AmigaGuru remembers U.S. Gold's 1989 conversion particularly for the loading interruptions at junctions. In the new project, the reviewer instead reports smooth transitions between different environments. The difference directly affects the rhythm: choosing a road and continuing without that pause preserves the connection between speed, scenery and the pressure of the clock.

OutRun AGA is the work of Agermose, who had already worked on the conversion of 1942 mentioned by the source, as part of RetroDNA. AmigaGuru's account follows a lengthy development process through preliminary builds and later revisions. The author had access to

those builds and describes substantial improvements in speed and presentation. His account therefore offers a close view, with his personal involvement explicitly acknowledged.

This work must also be distinguished from ReAssembler's separate OutRun conversion. According to the review, the version of that project under discussion required a 68060 and had no audio. These details refer to the version the author recalls and are chiefly useful for avoiding confusion between two independent initiatives. RetroDNA's defining choice is its hardware target: retaining the Amiga 1200's standard processor.

Additional memory remains part of the requirement, however. Describing it simply as a stock Amiga 1200 would obscure the required 4 MB of Fast RAM. In Amiga terminology, Fast RAM is memory available to the processor, separate from the Chip RAM shared with the custom chips. This distinction matters when assessing a real configuration: having an AGA system satisfies only part of the requirements, alongside the specified expansion.

Targeting the 68020 makes the blitter's role particularly interesting. This is the Amiga circuit specialised in operations on blocks of graphics data, capable of taking some work off the CPU. AmigaGuru attributes the game's behaviour to its intensive use and reports little performance benefit from faster processors. The review does not document the internal algorithms, but it does show how strongly the project depends on the balance between processor, memory and graphics hardware.

Resolution and colour count must likewise be understood as choices made for this conversion. A 320 × 224 display with 32 colours defines a particular workload for rendering the road, cars and backgrounds. Without technical documentation from the developers, it would be inappropriate to attribute specific redrawing or compression techniques to the engine. The available figures nevertheless reveal a clear priority: sustaining the action while keeping the scenery recognisable.

For smoothness, the source reports approximately 25 frames per second under normal conditions, dropping to around 17 in more demanding sections. The reviewer considers these slowdowns relatively unobtrusive. That positive judgement must remain distinct from the performance measurements: the phrase 'at full speed' used in the initial news summary therefore needs this qualification. The port has its own frame-rate target and varies under load.

Audio makes a substantial contribution to the recreation. The original's three selectable tracks, Passing Breeze, Magical Sound Shower and Splash Wave, are included, along with Last Wave for the high-score screen. The review credits the music work to DJMetune and TAWS and highlights simultaneous music and effects: the engine and squealing tyres accompany the soundtrack throughout the drive.

AmigaGuru describes three channels for music and three for effects. This count must be distinguished from the Amiga's four physical audio channels: six simultaneous sound parts require additional software handling rather than simply assigning one voice to each hardware channel. The source does not explain the implementation. For the player, the described result is



OutRun AGA's starting line on the coastal stage. Screenshot by RoarTJ, published in AmigaGuru's review.

tangible: the soundtrack remains present while the sound effects convey information about the action.

The package also includes four difficulty levels, each with its own separately saved high-score table. The review presents the conversion as complete in its routes and principal elements, while noting a possible omission in some crash animations. The author does not recall seeing the sequence in which the passengers are thrown from the car and the characters subsequently quarrel, and explicitly leaves this open for verification. It is therefore a detail that still needs confirmation.

The reported tests include a CD32 equipped with an SX-1 expansion and configurations with more powerful processors, up to a 68060 system. Positive behaviour on those machines is useful information for owners of expanded hardware, but applies only to the configurations actually tested. For the CD32 in particular, the reference is to the expanded console used by the reviewer: the text does not document operation on an unexpanded machine.

In practical terms, AmigaGuru presents the game as a free download. The body of the review discusses version 1.0, while the download reference names 1.0.1. Free availability describes the distribution method reported by the source; any terms of use and installation instructions should be checked in the material accompanying the project. Anyone wishing to try it should first confirm that their system has AGA and enough available Fast RAM.

The comparison with the 1989 conversion consequently takes on a more precise meaning. The two productions differ in their target hardware, required resources and development context. The contemporary result provides a way to study how distributing work differently among the Amiga's components can affect the final experience. Automatically treating it as evidence of what could have been achieved under the production conditions of the time would erase important distinctions.

OutRun AGA is interesting as a concrete example of present-day development for a historical platform: a defined hardware target, measurable compromises and attention to the continuity of play. For people still using an Amiga, it offers a new conversion to assess on their own machines. For those following its technical history, it shows how the scene's accumulated knowledge is being applied today to one of the most demanding arcade experiences to bring into the home.

Sources

OutRun AGA delivers the arcade dream on classic Amiga hardware



A stretch of road through stone arches, one of the conversion's environments. Screenshot by RoarTJ, published in AmigaGuru's review.

OUTRUN AGA / 4 OF 4

CONTINUED

IN PICTURES / OUTRUN AGA



The continuity of the driving experience depends on the road, traffic and transitions between locations. Conceptual illustration generated with AI for Retromind.



OutRun AGA's music selection screen, showing Passing Breeze. Screenshot by RoarTJ, published in AmigaGuru's review.

OUTRUN AGA

RetroDNA's conversion for the Amiga 1200.

68020 / 14 MHz

4 MB FAST RAM

FEATURES / 12.11.2026

INSECTOR X ON AMIGA: AN AGA CONVERSION BUILT WITH STATIC RECOMPILATION

Caiusarcade's first build translates the 1989 arcade game's code into native 68020 instructions. The method, requirements and performance of a project still in development.



Conceptual illustration recomposed with AI to fit this issue.

Bringing an arcade program to the Amiga means dealing with different processors and audiovisual systems. The first build of caiusarcade's unofficial Insector X conversion tackles this through static recompilation: according to Indie Retro News, code from the arcade machine's two Z80 processors is translated into native Motorola 68020 instructions. The project targets AGA Amigas and remains in development.

The starting point is the horizontally scrolling shooter published by Taito in 1989. The conversion draws its game logic, graphics and audio from the original, with the stated aim of reproducing its behaviour on Commodore's machine. This connection to the arcade software makes the work interesting: the adaptation must preserve the game's rules while moving their execution into a different technical environment.

Understanding the method begins with the meaning of static recompilation. In general terms, it is translation performed before execution: instructions written for one processor are transformed into code another can use. In the case described by the source, the destination is the Amiga's 68020. The processor can therefore execute instructions that have already been translated, following a different path from interpreting the original instructions while the game runs.

Native code still leaves the work of adapting the game to its new platform. Operations performed by the original program must produce pictures and sound through the Amiga's resources while behaving consistently with the arcade machine. Processor speed, memory management and screen updates become parts of the same problem. Translating an instruction correctly and obtaining a faithful sequence of gameplay are checks at different scales.

The project description reports frame-by-frame comparisons of code behaviour and pixel-by-pixel checks against the arcade graphics. This is a significant approach because it centres on observable results: an

object's position or the moment an action occurs can help identify differences in the conversion. Fidelity remains the stated goal; the available build represents one stage in that development.

The process also reportedly uses artificial-intelligence assistance for conversion and verification. This is an element of the production method described by the source. Assessing the result for players then involves concrete questions: correspondence with the original game, build stability and performance on supported configurations. The project's technical interest lies in the move from arcade software to a program executable on Amiga hardware.

The hardware target is clearly defined. An Amiga 1200 or another AGA-based machine is required, together with a hard drive and Kickstart 3.x. The source specifies approximately 1.3 MB of free Chip RAM and at least 4 MB of Fast RAM, with 8 MB recommended. These requirements belong together: AGA support identifies the compatible family of machines, while free memory specifies the resources the program must actually have available.

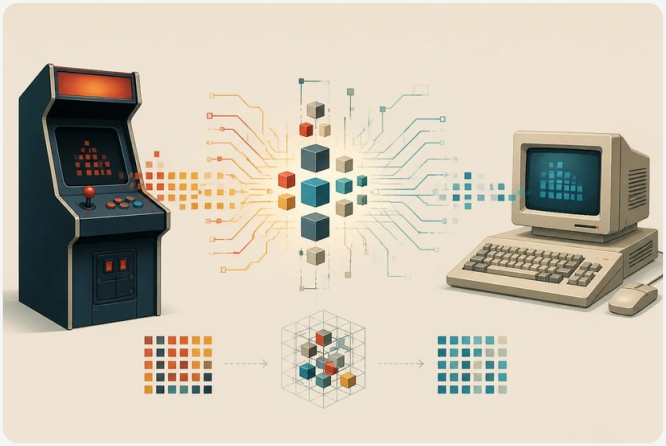
The distinction between Chip RAM and Fast RAM is particularly helpful for anyone approaching these configurations today. Chip RAM is also accessible to the Amiga's custom chips and takes part in audiovisual operations; Fast RAM provides processor memory that is not shared with those circuits in the same way. Adding Fast RAM therefore does not replace the required amount of Chip RAM. The word 'free' also matters when preparing the environment from which the game will launch.

The reported performance also shows why game-logic speed must be distinguished from the frequency of visual updates. According to Indie Retro News, on a standard 14 MHz 68020 the game and music run at full speed, while the screen updates less frequently in busy situations. Both can be true: the action continues to be calculated at its intended rate, while the player sees a less fluid representation of it.

With a 50 MHz 68030 or a faster processor, the source reports better performance through much of the action. The practical benefit therefore chiefly concerns sustaining the presentation during more demanding situations. The account provides no detailed measurements that would establish a constant frame rate for every configuration: the available information describes the build's behaviour rather than offering a uniform guarantee.

Preparing the files also deserves attention. The described process assumes access to the original ROMs and supplies a simplified tool for generating the build on Windows 10 or 11. Linux and macOS instructions specify Python 3 and vasm. A modern computer therefore participates in preparing the software, while the final program runs on the required Amiga configuration. Anyone trying it needs to consider both stages, alongside the target machine's requirements.

The preliminary state is also evident in the note accompanying the news video's audio. Indie Retro News observes that it appears to have problems and suggests a possible connection with the early version. That observation concerns this particular demonstration and



From arcade logic to native Amiga instructions. Conceptual illustration generated with AI for Retromind.

INSECTOR X / 3 OF 3

CONTINUED



Source: Indie Retro News

must be kept separate from the general statements about music and performance. Practical testing should therefore follow the project's current files and instructions, remembering that the conversion remains a work in progress.

For the Amiga scene, Insector X is an interesting example of contemporary development built around a historical game's code. Static recompilation provides a specific route for transferring arcade behaviour, while AGA and the available CPUs determine its practical presentation. This first build already reveals the relationship between the two: preserving the arcade logic requires adaptation whose results can be assessed against the characteristics of the machine running it.

Sources

Insector X comes to Commodore Amiga in an unofficial fan conversion

FEATURES / 08.11.2026

TIME PILOT ON AMIGA: TRANSLATING KONAMI'S ARCADE GAME FROM Z80 TO 68000

RMJ and JoeJoe bring the 1982 shooter to the Amiga 500 and 1200. Their conversion provides a way to understand what transferring arcade code to another architecture involves.



Conceptual illustration recomposed with AI to fit this issue.

Bringing Time Pilot to the Amiga from the arcade program's instructions is the work attributed to RMJ and JoeJoe by Indie Retro News, which reports a release for the Amiga 500 and Amiga 1200. The most interesting detail is the method: according to the account, the original machine's Z80 code was translated, instruction by instruction, into 68000 assembly. The process brings the logic of Konami's 1982 shooter into the Amiga ecosystem.

Time Pilot builds its identity around airspace that can be crossed in several directions, with continuous scrolling and battles spanning different historical periods. The player flies a jet, while each stage changes the historical setting of the confrontation. Free movement and time travel give the action a recognisable shape: the battlefield must be read by tracking threats arriving from different directions, rather than following compulsory progress along a single axis.

This structure helps explain why a faithful conversion remains interesting. Time Pilot's identity depends on the relationship between movement, enemy appearances and the rhythm of combat. Its graphics make it immediately identifiable; the continuity of its behaviour during play determines how closely the experience resembles the arcade game. For a project described as a '1:1' conversion, managing those relationships is an essential part of the work.

Early home versions had already taken the game to very different machines: the source mentions the Atari 2600, ColecoVision and MSX. Each move brought the original design into contact with specific capabilities and limitations. Its arrival on Amiga now adds another technical interpretation to that history, with a significant distinction: the account describes work based directly on the arcade program's instructions.

Understanding that choice requires separating the game from the instructions used to execute it. The

TIME PILOT / 2 OF 3

CONTINUED

Z80 and the Motorola 68000 family belong to different architectures: machine code intended for the former must be transformed before the latter can run it. Program operations therefore need corresponding implementations on the destination architecture, preserving the meaning of calculations, conditions and data accesses. Assembly makes it possible to work at this level, close to the operations the processor actually performs.

The instruction-by-instruction translation described by Indie Retro News concerns how the software's logic is transferred. In the project as presented, this happens during conversion and produces code for the target architecture. An emulator instead reproduces, during execution, the behaviour of the machine for which the program was written. These are different approaches to running the same game on another system, each with its own implementation and verification requirements.

The processor is only part of the question. An arcade game also depends on hardware that displays images, produces sound and receives input. Transferring CPU logic requires connecting it to the resources of the machine hosting the conversion. In practical terms, the result must keep the program's decisions, the picture on the screen and the player's possible actions consistent with one another. This is where an apparently simple description such as 'code translation' meets the broader work of adaptation between systems.

The named platforms are the OCS-based Amiga 500 and the AGA-based Amiga 1200. These are important reference points for the project's audience, placing the conversion across both generations of Amiga hardware. The compatibility claim tells users which machines the work targets; naming the chipset alone does not describe the graphics and sound solutions adopted. The report focuses chiefly on code translation and fidelity to the arcade game.

The phrase '1:1' must also remain tied to its source: it is the description reported by Indie Retro News. Independently assessing correspondence with the original would require comparisons of gameplay behaviour, audiovisual presentation and control response. Translating the instructions gives a precise technical insight into the project's approach; overall fidelity remains a property of the playable result, to be considered as a whole.

Choosing the 1982 Time Pilot also keeps the conversion's subject clearly defined. Its successor, Time Pilot '84, released in arcades in 1984, adopted an entirely futuristic setting. RMJ and JoeJoe's project instead returns to the first game's formula, in which the changing historical period accompanies progression. That relationship between game structure and time travel forms part of its historical identity, alongside the freedom of multidirectional combat.

For preservation, a conversion of this kind offers another way to access a game on different hardware. Keeping the arcade program and documenting the original machine retain their separate roles; a port makes it possible to study how that behaviour is transferred elsewhere. When the conversion method is described, the distance between the architectures also becomes a subject of interest: it helps us read



Source: Indie Retro News

TIME PILOT / 3 OF 3

CONTINUED



From arcade code to the Amiga architecture: a technical transition between different systems. Conceptual illustration generated with AI for Retromind.

a videogame as rules implemented through specific technical resources.

This is why Time Pilot on Amiga merits attention beyond its availability announcement. The project connects a 1982 arcade game, two generations of Commodore computers and contemporary work on code. For followers of the Amiga scene, it adds a title to the software available for historical machines; for those interested in videogame technology, it provides a concrete case of transfer between architectures. Its significance comes from the meeting of those possibilities: playing on an Amiga and understanding how the original's behaviour can be reconstructed on different hardware.

Sources

Time Pilot arrives on Amiga in a direct conversion of Konami's arcade game

FEATURES / 07.11.2026

AMIGA: VESURI IMPROVES FRAME RATES AND COMPATIBILITY IN CLASSIC GAMES

From Road Rash to Wing Commander, WHDLoad-based patches address rendering, game timing and bugs, with assistance from artificial intelligence.



Conceptual illustration recomposed with AI to fit this issue.

Making an Amiga game smoother also requires attention to what happens between frames: vehicle movement, control response, collisions and timers must continue to behave consistently. Vesuri is tackling this collection of problems through updates to classics including Road Rash, Wing Commander and Test Drive. The changes use WHDLoad's infrastructure and aim to make better use of expanded Amigas.

According to Indie Retro News, Vesuri is publishing versions with improved frame rates and specific fixes on GitHub, combining AI tools with his experience in the Amiga demoscene. The source points to an earlier optimisation of Stunt Car Racer to 50 frames per second, achieved through manual programming. That result forms the background to work now extending to several games, each with different needs.

The problem will be familiar to owners of an Amiga accelerator. A more powerful processor offers additional computing resources, but software must be capable of using them. The source mentions Apollo 1240 expansions for the Amiga 1200, illustrating the gap that can exist between hardware capabilities and the behaviour of programs designed for earlier configurations.

Within the Amiga architecture, the CPU also works alongside custom chips and memory. Increasing its speed can ease some calculations, while other operations remain constrained by how the game organises its work. Patches therefore need to address the program's actual limits: waits, drawing routines, synchronisation and dependencies on the machine's characteristics.

A crucial distinction is between frame rate and simulation speed. The former describes how many images are produced each second; the latter determines how quickly the game world progresses. If movement and logic depend directly on how often the scene is drawn, faster rendering can alter driving or flying. Producing

AMIGA PATCHES / 2 OF 3

CONTINUED

more images while preserving consistent timing requires changes to timekeeping as well as graphics performance.

Test Drive makes this relationship particularly clear. Enhanced framerate mode promises smoother driving by changing the presentation of scenery and traffic and making race timing and average-speed scoring more consistent. It also fixes sideways jumps in the road, steering response in bends and nearby vehicles disappearing before a collision. These changes directly affect the readability of the road and the predictability of the controls.

In Road Rash, the update introduces a smoother mode synchronised with the display refresh, intended for Amigas with Fast RAM. This memory can help the processor without competing for the same accesses needed by the machine's custom chips. In the notes quoted by the source, the improvement is accompanied by consistent simulation speed and a reduction in visible steering lag.

The Road Rash work also addresses opponents' lane positions, flickering bikes in the distance and in the mirror, and vertical jumps on hills. On compatibility, the notes identify a fix for a loading freeze on 68030 machines with the cache disabled. This is a concrete example of the variety of configurations a contemporary Amiga patch must consider.

4D Sports Driving presents another case. Alongside faster rendering and correct race timing, the notes describe an experimental A4000 simulation mode intended to make driving smoother and more responsive on faster PAL Amigas. It is selected automatically on 68040 or 68060 systems, with simulation settings that can be saved.

An important practical consequence emerges here: replays depend on simulation behaviour. The update supports saving and playing back A4000-mode replays and retains support for original replays; older A4000 replays, however, are incompatible with the described changes. Anyone keeping recordings of their races therefore needs to read the notes for the version in use, because compatibility can vary even within the same game.

In flight simulators, consistent timing involves a different collection of elements. Their Finest Hour is reported to receive faster rendering, no tearing and full terrain detail on fast Amigas, alongside changes to aiming, hits, bombing and the game's AI behaviour. The stated aim is to maintain a constant game speed as the frame rate changes.

The changes also include adjustable mouse sensitivity and 128-second flight recordings. New films saved in the enhanced mode require that same mode for playback through the updated slave, while original files remain loadable. As with 4D Sports Driving, information about formats and playback conditions forms part of the update's technical value.

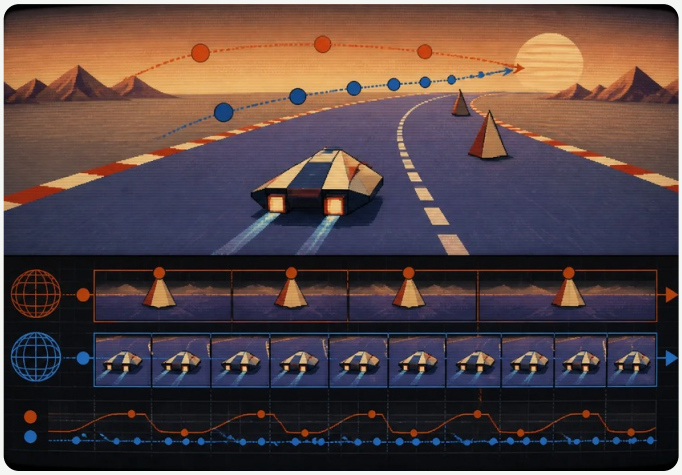
Wing Commander extends the work to the sequences surrounding flight. The source lists new timing for the introduction, briefings, and take-off and landing animations, along with smoother camera movements and spacecraft effects. Improvements to projectiles and hit detection involving asteroids and mines are also included.



Source: Indie Retro News

AMIGA PATCHES / 3 OF 3

CONTINUED



Frame rate and simulation speed do not always coincide: smoother rendering may require different timing. Conceptual illustration generated with AI for Retromind.

A particularly interesting detail concerns Wing Commander's disk edition: the notes describe faster arithmetic and collision checks in every mode, including on the Amiga 500, while preserving the game's behaviour. These changes therefore also benefit a basic configuration, although the gains and conditions differ from those of accelerated machines. Red Baron, meanwhile, receives CPU-dependent rendering changes collected under a single Enhanced framerate option.

The operational link between these projects is WHDLoad, the system that launches many Amiga games from a hard drive through game-specific components known as slaves. A slave handles loading and applies the adaptations the program needs. Over time, the scene has accumulated valuable knowledge at this level about compatibility, hardware access and games' internal workings.

Vesuri's updates build on work by authors including JOTD, Psygore, StingRay and CFou!. According to the source, adopting the new versions involves replacing the relevant slaves. The project thus joins an established practice of community maintenance, in which later changes can draw on technical foundations built by other developers.

Anyone wishing to try them must assess the conditions game by game. Enhanced framerate is already enabled in the supplied icons in several cases; some installations use CUSTOM2=1. The notes also describe ways to return titles such as Wing Commander and Their Finest Hour to their original modes. CPU, available memory and replay or recording instructions remain decisive: the same label covers changes with different requirements.

AI's role must be understood within this development process. The report presents it as assistance enabling Vesuri to tackle more quickly work previously done manually. The described results concern specific software changes: more efficient routines, corrected timing and resolved compatibility problems. Experience with the Amiga architecture remains the technical context in which these tools are used.

The value of this activity is clearest in the precision of the changes. A correctly behaving lap counter, a collision detected at the right moment or a recording that can be replayed makes a game more reliable in everyday use. Original modes and enhanced options also let us observe how the same program responds to different configurations and technical choices. For the Amiga scene, this is maintenance and research with directly usable results: new ways to run, study and play familiar software.

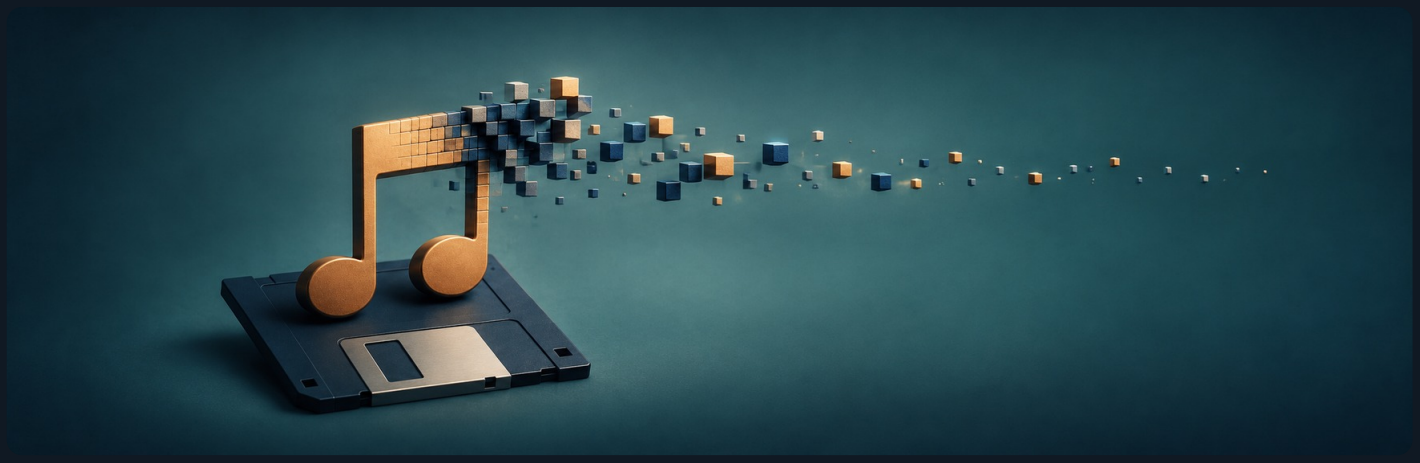
Sources

Enhanced-framerate Amiga releases with AI assistance and Vesuri

FEATURES / 06.11.2026

KARSTEN OBARSKI, SOUNDTRACKER AND THE MUSIC OF AMIGA CRACKTROS

The reuse of music from Amegas and Crystal Hammer reveals how compositions, tools and credits travelled between commercial games and the Amiga scene.



Conceptual illustration recomposed with AI to fit this issue.

A Red Sector crack intro associated with Crystal Hammer used the music from Amegas: two games connected by compositions from Karsten Obarski, the creator of The Ultimate Soundtracker. It is one of the episodes revisited by an AmigaGuru article, offering a view of how closely musical tools, commercial productions and unauthorised distribution networks were intertwined in the Amiga's early years.

The account leaves a question open: what relationship Obarski had with those reusing the tracks. The material described chiefly documents their circulation and the credits appearing in intros. Any involvement by the musician in the named groups' activities remains unproven. Keeping that distinction allows us to read the story without turning an intriguing coincidence into an established biography.

Understanding the story requires distinguishing the objects involved. A cracktro is a short introduction added to a modified copy of a game, used by groups to present themselves through graphics, messages and music. A trainer instead changes how a game operates, providing advantages or modifications to its rules. These elements could accompany the software and acquire identities of their own, particularly when players encountered them before the game's opening screen.

AmigaGuru describes two cases involving Crystal Hammer, released in 1987. Red Sector reportedly chose Amegas's music for its intro, credited to Obiwan, an alias the source associates with Obarski. A trainer distributed under the Blizzard name instead reportedly used Crystal Hammer's own music, credited to Obi. The distinction is interesting: presentations surrounding a single game drew on different compositions by the same author.

A musical credit helps reconstruct an attribution; by itself it proves neither permission, collaboration nor

SOUNDTRACKER / 2 OF 3

CONTINUED

membership of a group. Establishing those relationships would require direct testimony or further documents. The source raises questions precisely at this point without providing a conclusive answer. The episode's strongest historical significance therefore concerns the reuse of music and the way the composer's name travelled with it.

Obarski's place in Amiga culture also involves the software used to construct that music. The Ultimate Soundtracker provided a sample-based composition environment: digitised sounds could be triggered and organised through sequences of notes and commands. The tracker approach made a piece's structure visible as a grid of events, connecting musical writing with control over the computer's sound output.

The Paula audio chip supplied the machine's sound capabilities. Soundtracker gave musicians a way to use them, while sample libraries provided the starting material. These are three distinct layers of the same experience: hardware, composition tool and available sounds. Considering them together helps explain Obarski's influence, because a technical possibility also becomes a widespread practice through the tools that make it accessible.

AmigaGuru emphasises the ST-01 and ST-02 disks attributed to Obarski. Its account states that the first contained 126 samples on a single floppy. The number suggests a collection already varied enough to support different experiments and compositions. Such a library made a practical contribution: it allowed people to begin with ready-made sounds and concentrate on selecting them and organising a piece.

The source then describes the ST series expanding through musicians' and enthusiasts' work, giving a total of 116 disks over roughly six years. This is a count reported by the blog and should be treated as part of its reconstruction. The phenomenon it describes is nevertheless clear: the initial disks prompted community production of sound collections that expanded composers' available vocabulary.

A distinction useful to listeners emerges here. Sharing a sample means starting from the same sonic material; reusing an entire composition means reproducing its melodies, rhythm and arrangement. Libraries could create tonal similarities between independent works.

In the intros cited by AmigaGuru, however, the recognisable material consists of specific Obarski pieces transferred away from the games for which they were created.

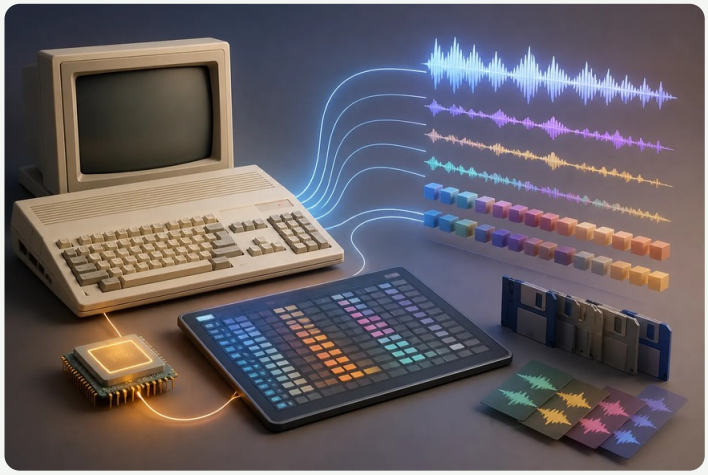
The blog presents Amegas's soundtrack as the first commercial use of music made with Soundtracker. In this reading, the game made the new tool's results audible, providing a concrete example of what combining samples and sequences could achieve. Later reuse in intros widened the piece's circulation further, connecting Obarski's name both to the composition program and to instantly recognisable music.

The overlap between these environments needs precise treatment. Creating tools, composing for games and participating in cracking groups are different activities, even when they share references and material. Commercial music appearing in a cracktro demonstrates that the track crossed a



Source: AmigaGuru - Amiga

SOUNDTRACKER / 3 OF 3 CONTINUED



Hardware, composition tool and samples: three layers of the same musical experience. Conceptual illustration generated with AI for Retromind.

distribution boundary. Determining who transferred it, through which personal relationships and with whose consent requires a separate investigation.

Another part of the account concerns players' memories. AmigaGuru's author recalls Red Sector's intro with the Amegas music still circulating in 1989 and cites The Great Giana Sisters among the examples. In his view, some players came to associate that music with the game that followed. This retrospective testimony needs the caution appropriate to personal memories, but describes a plausible effect of listening order: what precedes a game can become a lasting part of its perceived identity.

This has practical consequences for preservation. Studying the experience of a particular copy benefits from separating the commercial program, the added introduction and any modifications. A recording of startup may document music encountered by many users without proving that it belonged to the original soundtrack. Preserving the recording's context helps avoid mistaken attributions and allows different experiences of the same title to be compared.

Today's habits also shape what becomes visible again. The source's author observes that he mostly uses WHDLoad and now seldom encounters crack intros. This is a personal observation, not a characteristic to generalise to every installation. It nevertheless reminds us to document the version actually running when comparing memories, videos and preserved copies: a game's title alone can omit a significant part of the experience.

AmigaGuru's account therefore supplies a concrete line of research. Comparing the named intros, transcribing their credits and checking the music could establish the chronology of reuse more precisely. Clarifying Obarski's involvement would require evidence of a different kind. Meanwhile, Amegas and Crystal Hammer already reveal two forms of his legacy: compositions travelling between contexts and tools giving other musicians the means to create new music.

Sources

Karsten Obarski, Soundtracker and the cracking side of the Amiga scene

FEATURES / 10.11.2026

SPECIAL FORCES: THE PSYGNOSIS PROJECT REDISCOVERED IN DOCUMENTS FROM 1997

A document recovered by The Retro Collective describes a tactical action game planned in different forms for PC and PlayStation. One account connects its history with Andy McNab's Team SAS.



Conceptual illustration recomposed with AI to fit this issue.

Four operatives and a planning mode on PC; two characters and more action-oriented missions on PlayStation. Special Forces, proposed by Psygnosis Stroud in 1997, envisaged two interpretations of the same military project, adapted to their respective platforms. Its intentions can be reconstructed from a design document recovered by The Retro Collective, digitised by Neil Thomas and discussed by Games That Weren't.

The recovered material concerns the game's proposal. The report does not document the recovery of a playable Special Forces prototype: in an update added to the article, Stuart Maine states that the project remained at the internal-pitch stage, although he contributed some level design. His account also introduces a significant connection: the proposal reportedly led to Andy McNab's Team SAS, already recovered by Games That Weren't through Maine's own donation.

The original document was included in a donation to the museum. Dated 13 May 1997 and marked Draft 1.6, it names Paul Walker, Matt Wilkinson and Derek Ward. It contains a general game description and two separate technical sections for PC and PlayStation. This separation shows how the authors addressed differences between the versions from the proposal stage.

The starting idea was a third-person 3D action adventure with missions approachable in different ways. The stated reference was exploration and combat in games such as Tomb Raider, combined with managing military operations. Players would lead an elite international unit working for the United Nations on covert assignments around the world.

Proposed operations included hostage rescues, reconnaissance and the destruction of enemy installations. One more unusual scenario involved infiltrating an opposing facility equivalent to Area 51. This variety was intended to support operational freedom: infiltration

SPECIAL FORCES / 2 OF 3

CONTINUED

and direct assault could be combined according to the objective, resources and the team's abilities.

On PC, players would choose four members from a pool of eight, drawn from units such as the SAS, US Rangers and Spetsnaz. Each had a personality and specialist skills that could improve through completed missions. During the action, players would directly control one character while the others followed under computer control, reacting to the player's initiatives.

The strategic component would let players prepare attacks by assigning targets, routes and equipment to individual operatives. Timing their interventions was also planned. Such a system's value depends on coordinating different actions: choosing a teammate's position becomes meaningful when their intervention can be synchronised with that of the controlled character. Here the proposal sought to give the squad a tangible role in mission structure.

Enemy forces were also described as part of a responsive environment. The plan called for increased patrols after player actions, repairs to damaged installations and changes in behaviour. Technical documentation proposed daily routines for soldiers and a system of interconnected agents: software entities managing coordinated behaviour. The aim was to make military bases places with activities of their own, observable during exploration.

These statements must be read as design objectives. Their presence in the draft does not demonstrate implementation or testing. They do help explain why character intelligence was among the proposal's most demanding aspects: autonomous teammates, reactive opponents and planned orders would have to operate in the same environment. The game's quality would also depend on players being able to understand their reactions.

For PlayStation, the authors outlined a more contained version. Missions would be smaller, enemy behaviour simpler and the squad reduced to two characters. Detailed strategic planning was excluded in favour of a more arcade-oriented approach. Games That Weren't relates these choices to the console's hardware limitations, without providing enough evidence to identify a particular technical bottleneck.

The differences would directly shape the experience. A smaller squad changes the number of actions to coordinate; more compact environments alter the space available for flanking and alternative routes. Removing detailed preparation also shifts decision-making towards events during direct control. The document therefore allows multiplatform adaptation to be studied through design decisions, even before considering possible graphical differences.

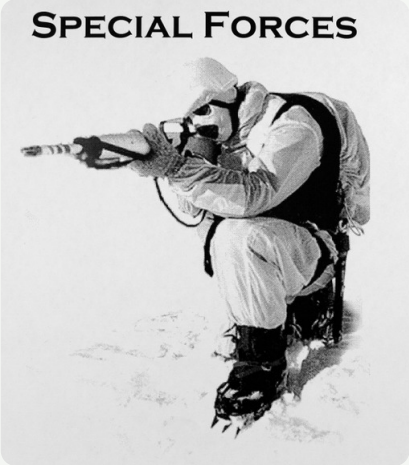
Around this framework, plans included motion-captured animation, realistic military equipment and both daytime and night-time environments. A laptop-style interface would collect maps, orders and tools for preparing operations. The proposal also included network games for up to sixteen participants and a separate four-player cooperative mode. These features,



A recovered design document examined in the archive. Conceptual illustration generated with AI for Retromind.

SPECIAL FORCES / 3 OF 3

CONTINUED



Source: Games That Weren't

too, belonged to the intended programme of work described in the documents.

The documentation preserves an equally useful production perspective. Psygnosis proposed an initial three-month design and planning phase beginning in June 1997, followed by full development from September. Completion was scheduled towards the end of 1998, with a possible release in early 1999. The estimated budget was approximately £1.19 million.

The proposed team included six programmers and four artists, supported by a game-design coordinator and a producer. Music and motion capture would require additional resources. These figures describe the organisation envisaged for making Special Forces; they do not establish actual expenditure or production proceeding to that timetable. They are valuable because they connect the design's ambitions to the people and time expected to realise them.

Internal comments reproduced in the document show interest in the project while identifying specific difficulties. These included AI complexity and reliance on technology being developed by other Psygnosis studios. The proposal thus also records a production discussion: imagined features had to be assessed alongside the tools needed to build them and the planned collaborations.

Stuart Maine's later contribution clarifies the proposal's fate relative to uncertainties in Games That Weren't's initial account. Maine recalls helping another team with some level designs and says Special Forces never progressed beyond the pitch. The connection to Andy McNab's Team SAS suggests continuity worth investigating, without allowing every feature in the 1997 draft to be automatically attributed to the later project.

Another piece of information received with the document remains separate: former soldier and writer Andy McNab was reportedly intended as the protagonist, with plans to involve him in motion capture. Games That Weren't states that it found no confirmation in the surviving documents. The later connection with Team SAS makes the report interesting, but its evidential status differs from that of material readable in the proposal itself.

For anyone studying videogame development, the recovery reveals decisions a finished executable could scarcely explain: how features were divided between platforms, staffing estimates and internal managers' observations. Neil Thomas digitised the complete document; Games That Weren't presents selected scans. Through these papers and the collected testimony, Special Forces now gains a more precise place in the history of Psygnosis projects.

Sources

Special Forces: rediscovering Psygnosis's PC and PlayStation project

FEATURES / 11.11.2026

MEGA MAN LEGENDS: A 'NEW MEDIA' LISTING AND THE POSSIBILITY OF A COLLECTION

A report from Ontario's film-production portal puts Capcom's subseries back in the spotlight. What is known, and what a possible return might mean.



Conceptual illustration recomposed with AI to fit this issue.

Mega Man Legends appears in a listing on Ontario's film-production portal, classified as 'new media'. Reported by Nintendo Everything, the discovery renews discussion of a possible rerelease of the PlayStation classic. The project's nature remains unclear, however: that classification does not automatically identify a videogame, let alone a remaster.

According to the report, the listing gives a filming start date of 14 September 2026 and an expected conclusion on 1 December. These dates describe the production period recorded in the document, not a release window. The report's origin also deserves attention: this is a portal connected to audiovisual production, rather than the shops and classification bodies that often reveal advance information about videogame releases.

The phrase 'new media' leaves several interpretations open. Nintendo Everything explicitly considers the possibility of a television project as well. Already treating the listing as evidence of a collection or remaster would therefore go beyond the information reported.

Interest in a possible videogame return nevertheless has a clear context. Capcom has released several collections from the Mega Man catalogue, revisiting the main series and branches such as Mega Man X and Battle Network. Zero and ZX have received similar treatment. Legends occupies a particular position in this history: according to the source's account, it remains the principal subseries absent from this wave of collections.

Comments by Capcom's Shingo Izumi, quoted in the same article, make the comparison relevant. Discussing future possibilities, Izumi named a Mega Man Legends collection among candidates worth considering for new projects. This expressed openness in conditional terms. The source establishes no connection between those comments and the Ontario listing: the two elements must be assessed separately.

MEGA MAN LEGENDS / 2 OF 2

CONTINUED



A videogame rerelease or an audiovisual project: the possibilities remain open. Conceptual illustration generated with AI for Retromind.

A collection and a remaster would also address different needs. The first primarily concerns selecting works and presenting them together in a shared edition. Remastering instead implies changes to a game's presentation, whose extent depends on the project. A release could combine both approaches, but no information establishes such work on Legends. The 'new media' classification specifies neither content, platforms nor technical changes.

For a production associated with the PlayStation era, a rerelease could also offer an opportunity for reassessment. Collections place individual games within a series' development and help explain its different branches. In Legends' case, the editorial value of a return would also depend on giving it a recognisable place alongside the other Mega Man families already reissued.

In practical terms, contemporary accessibility would have to be judged by the work actually done. Compatibility, control response and audiovisual fidelity are decisive when an older game moves to different systems. Any changes should be documented clearly so players understand the relationship between the new edition and the original experience. These are criteria for assessing a future release, not features that can be attributed to the project revealed by the listing.

Commercial availability and preservation also require distinction. A new edition can make a work easier to access; its contribution to historical understanding depends on faithful reproduction and any contextual material included. For Retromind, this is where a Legends return could gain substance: making a game available again while providing ways to understand its place in Capcom's catalogue.

The Ontario listing therefore remains something to follow, with a meaning that is still limited. An announcement could establish what product bears the Mega Man Legends name and, if it is a videogame rerelease, which works and changes it includes. For now, Izumi's comments provide a precedent for considering a collection, while the listing documents a project whose form remains unidentified.

Sources

Mega Man Legends listing surfaces, classified as 'new media'

FEATURES / 09.11.2026

PUZZLE MANIA XXXL: AN MSX1 PUZZLE GAME WITH ROMS UP TO 128 MB

The File-Hunter's game joins MSXdev26 with adult imagery and multimedia content on NEO cartridges. The 128 MB version requires openMSX 22.



Conceptual illustration recomposed with AI to fit this issue.

Puzzle Mania XXXL combines a sliding-tile game with 16 MB and 128 MB ROMs, despite specifying an MSX1 with 16 KB as its machine requirement. Presented as entry 19 in MSXdev26, the project by The File-Hunter, a member of FONY, provides a concrete case for examining the relationship between a historical platform and cartridge formats capable of storing large amounts of audiovisual material.

The imagery needs to be explained at the outset: the game contains female nudity and a playful erotic presentation, so it is intended for adults. MSXdev's listing emphasises this through numerous double meanings. To understand the project's interest within the MSX scene, the most useful points are the puzzle structure, distribution in two formats and the different environment required by the larger version.

The described mechanic is straightforward: slide the tiles of a fragmented picture until it is reassembled. The menu lets players choose the puzzle type and size; the presentation also advertises numerous images. It is therefore a visual-reassembly formula in which the pictures to unlock or reconstruct form an important part of the attraction.

MSXdev presents Puzzle Mania XXXL as the MSX1 version of an earlier Puzzle Mania. This connection helps frame The File-Hunter's work: an existing formula is offered for the first generation of the MSX standard, accompanied by a substantial multimedia package. According to the listing, the author specifically attributes the title's XXXL suffix to the size of the ROM.

Reading the figures correctly means distinguishing the memory required by the machine from the capacity of the medium containing the game. The specified 16 KB and the 16 MB or 128 MB ROMs describe resources with different functions. Working memory is used during execution; cartridge space stores the program and

PUZZLE MANIA XXXL / 2 OF 2

CONTINUED

its content. A large ROM can therefore provide a substantial collection of data even when the computer's memory requirement remains modest.

This distinction also explains the kind of expansion Puzzle Mania XXXL offers. ROM capacity concerns the space available for pictures and audio; displaying and playing them depends on the software and execution environment. The megabyte figure alone measures how much material can be stored. Judging smoothness, sound reproduction and the quality of the experience would instead require hands-on testing.

The source explicitly connects large cartridges with the ability to reproduce moving images and sound. Puzzle Mania XXXL follows this direction in the contemporary scene: using higher-capacity formats to accompany gameplay with extensive audiovisual content. A sliding puzzle makes the relationship between a simple game rule and a potentially much larger image collection particularly clear.

The two distributed versions are named NEO16MB and NEO128MB. The most important practical distinction concerns the latter: MSXdev says it can only be used with openMSX 22 and contains larger streams. Anyone wishing to try it must therefore take this precise emulator requirement into account. The general specification of an MSX1 with 16 KB must be read alongside the note for the selected format.

The description of larger streams remains information about the 128 MB version's contents. The listing does not quantify differences in duration, resolution or audio quality, so it is best to retain the documented distinction. For possible use on original hardware, the ROM format and the medium employed must likewise be considered separately from the computer model named in the requirements.

The MSXdev page also offers online play through The File-Hunter or the MSXdev games app. For readers chiefly interested in the puzzle formula, these provide ways to approach the project; for anyone examining NEO128MB, the explicit reference to openMSX 22 remains relevant.

Puzzle Mania XXXL's interest for the homebrew scene lies in this specific combination: an author already connected to the project, an MSX1 version and two formats placing multimedia content at the centre of the offering. Its inclusion in MSXdev26 documents how contemporary development for a historical platform also involves work on storage media, execution environments and access methods. These choices directly affect what is distributed and how enthusiasts can use it today.

Sources

MSXdev26 #19: Puzzle Mania XXXL arrives

BLOG / 07.11.2026

FAN PATCHES: KEEPING AN EDITION PLAYABLE OVER TIME

Fixes, version numbers and instructions accompany a translation long after launch. Their quality allows a patch to be selected, applied and preserved even when a project slows down or ends.



Conceptual illustration recomposed with AI to fit this issue.

When a fan patch moves from version 1.0 to 1.1, downloaders need to understand which problems it fixes, which copy of the game it requires and how to update an earlier installation. That information determines the revision's practical usefulness. Work after publication therefore carries a specific editorial responsibility: making each edition identifiable and explaining the conditions under which it can be used.

A release defines an edition

A published translation brings linguistic decisions and technical solutions together in a configuration its authors consider ready for players. Feedback then arrives from playthroughs different from those of the testers: a conversation on a side route, a menu string that is too long, a term used inconsistently. Maintenance organises these reports and decides which changes warrant another distribution.

It is useful to distinguish translation coverage from an edition's stability. Translating all intended text describes the extent of the language work. Checking dialogue windows, modified routines and gameplay paths concerns the result's behaviour. These are connected dimensions, each with its own scope for revision. Accurately describing a project's state helps players interpret terms such as 'complete', 'beta' or 'revised'.

The Telefang translation's release pages offer a concrete example of this continuity, collecting fixes, variants and known-issue notes. Their value is in giving each distribution readable context. An increasing version number gains meaning through the associated information: which defects were addressed, which options are available and which limitations still matter.

Identifying the game to modify

The first condition of reliability comes before launching the game. A patch operates on specific input data that must be identified precisely. Commercial title and region can still leave different revisions; for optical-disc games, the representation of the disc

image also matters. In a project targeting a machine such as the Saturn, specifying the expected base prevents a difference in file preparation from being mistaken for a translation defect.

This is where checksums come in: values calculated from a file's contents. Comparing a copy's fingerprint with the one supplied by the authors checks that the base matches. A filename can be changed freely; the fingerprint depends on the bytes. The pkmn-azoth-translation documentation describes this practice through identifiers for both the base and the produced version, also providing a reference for checking the result.

The README should accompany the entire operation: identifying the copy, choosing the patching tool and checking the outcome. It must also explain whether a new patch applies to the original base or to an already modified version. Different distributions may use different procedures. Leaving this choice implicit forces users to guess at precisely the point where an error can compromise the result.

Tools are also part of the conditions of use. Naming the program used and any necessary settings makes the procedure repeatable. When the recommended patcher changes, updating the instructions belongs to the same maintenance work as fixing the game.

Fixes have consequences

Changing two lines of text may require broader checks than their length suggests. In historical games, writing must coexist with the available space, how strings are addressed and the rules for displaying them. Clearer wording may use more characters; a change to text layout may affect other screens sharing the same routine.

A useful report therefore describes the conditions of a problem. The patch version and execution environment place it in context; a precise route to the scene helps others reproduce it. After a change, testing includes both the reported case and nearby functions affected by the modification. This work limits regressions: defects introduced by a later fix.

Public notes should reflect the scope of the testing performed. A check on a particular emulator documents that environment. Original-hardware tests add specific information, which should be reported with the relevant conditions. Stating exactly what was checked also makes future reports more useful: someone encountering a problem can determine whether they are exploring a combination that has received little testing.

Language revision follows a similar logic. Correcting a name requires finding its other occurrences and considering its use in different contexts. An updated glossary reduces the risk of later contributions reopening settled choices or introducing new inconsistencies.

Writing for those who arrive later

README and changelog have different readers and functions. The former guides someone encountering the project today through instructions valid for the available version. The latter preserves the sequence of revisions and lets existing users assess an update. If essential information is scattered across years of announcements,



Source: Telefang translation release notes

FAN PATCHES / 3 OF 3

CONTINUED



Before applying a patch, identify the correct source copy precisely. Conceptual illustration generated with AI for Retromind.

every new user must reconstruct the current procedure independently.

An effective changelog describes changes' effects in enough detail to recognise them. 'Fixed some bugs' says little. Specifying that truncated text on a particular screen has been corrected connects the revision to an observable problem. When a change requires new patching instructions or restricts a previously supported combination, that consequence deserves particular prominence.

Version numbering chiefly requires consistency. A group can distinguish language revisions, technical updates and small fixes through whichever convention it can sustain, explaining it in the documentation. What matters is that different packages remain identifiable. Silently replacing an archive's contents while retaining the same version makes reports ambiguous and complicates comparison between copies preserved by different people.

Some documentation also transfers work between collaborators. The fan-translation-template project proposes a glossary, a decision log and a task-status log. Tools like these preserve the reason for a choice alongside its result. Someone returning to the translation can understand why a term was adopted or a change left pending without depending on the original authors' memories.

Sustainable maintenance needs clear boundaries

These practices take time, often contributed by volunteers. Continuity can take different forms: periodic updates, attention only to serious defects, or a final consolidation phase. Defining the scope of support helps the group organise its commitment and lets players form realistic expectations.

Translation makes a game's content accessible; maintenance supports its use under stated conditions; preservation keeps the elements needed to identify and reconstruct that edition. The three activities support one another. Keeping the patch, instructions and file identifiers together allows the result to remain understandable after its authors have finished working.

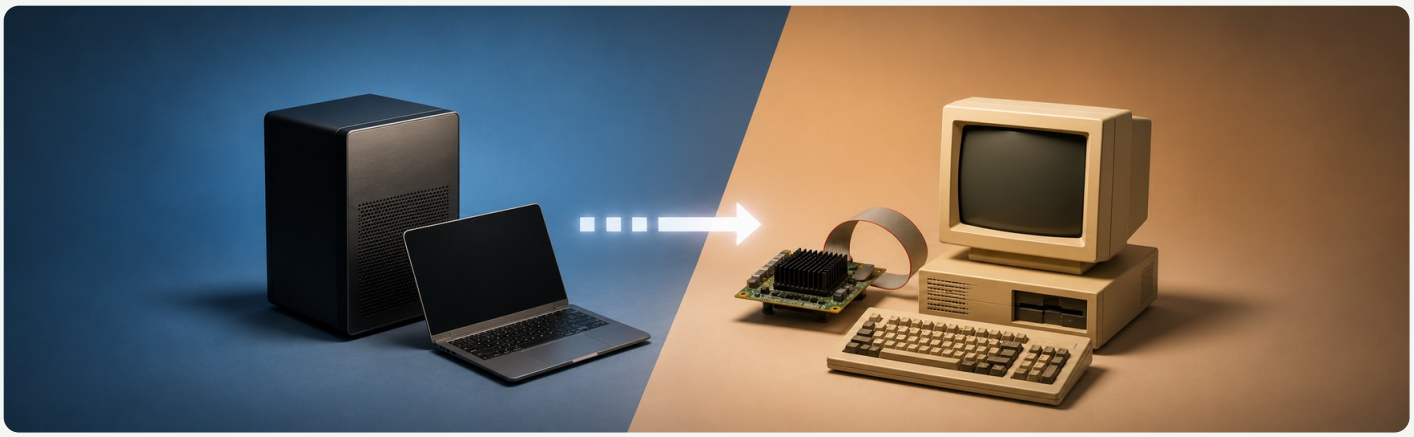
Preserving earlier versions with their respective notes also makes it possible to interpret old reports or reconstruct the project's state at a particular moment. Clearly identifying the recommended edition is enough to guide someone who simply wants a translation to use. Revision history can remain available without complicating the main route.

A project can therefore end while leaving an orderly legacy: an identified final version, self-contained instructions and explicit known limitations. If someone resumes development, they will find the decisions from which to continue. Those who only want to play will have an understandable procedure, without hunting for an answer posted years earlier in a scattered discussion.

BLOG / 09.11.2026

PROJECTS FOR CLASSIC SYSTEMS: THE ROLE OF MODERN HARDWARE IN THEIR REQUIREMENTS

A computer used to compile software and an accelerator used to run it perform different jobs. Distinguishing these roles helps explain compatibility, the cost of getting started and a project's aims.



Conceptual illustration recomposed with AI to fit this issue.

A conversion compiled on a recent PC may leave the historical machine to run the entire game. An accelerator needed during execution forms part of the configuration on which that game works. These situations create different expectations: requirements become useful when they explain where the work happens and what equipment downloaders actually need.

The computer that builds the program

Contemporary development for a historical platform can distribute work across very different systems. A PC runs tools that turn source code into a program and prepare its resources. The target machine receives the result: an executable, a disk image or another loadable format. This is the principle of cross-compilation, in which the computer producing the software differs from the one that will execute it.

This separation allows time and computing power to be devoted to preparing data. Processing performed before distribution can reduce the work required during play. The historical CPU must still handle the final program's operations within its own memory and speed constraints. The development PC's power therefore describes a specific production stage.

A documented example comes from Retro Home, a Raspberry Pi project whose build procedure requires an x86_64 Linux workstation and sufficient storage. That requirement identifies the environment used to construct the image. Someone using a prepared image follows a different procedure from someone generating it from source. Putting both under a generic 'hardware needed' label would assign the reader a cost belonging to a particular way of using the project.

Expansions that participate in execution

When an additional component works while the program is running, it changes the required platform's description. If a conversion needs an accelerated CPU, its reference configuration includes that accelerator. If it needs additional memory, the software has space that the basic machine may lack. These are essential details for understanding the project's intended audience.

Processor, memory and storage address different problems. A faster CPU can complete more calculations in the same interval; more RAM can keep more data available, provided the program can use it. A storage device retains files and exposes them through a particular interface. These resources interact but retain separate functions: adding disk space does not increase working memory.

Modern storage likewise needs precise description. A replacement medium may change loading and file management while leaving the historical machine to execute the game. A device that processes data during operation assumes an additional role. An accessory's appearance and manufacturing date reveal little about this difference; what matters is which operations it performs.

This leads to a distinction useful in editorial coverage as well. Compatibility with an expanded machine is a technical result for that configuration. Naming the model, memory and acceleration allows the achievement to be appreciated at its actual scale, avoiding a general platform label that suggests the same performance is available on the basic system.

The BIOS specifies the environment being recreated

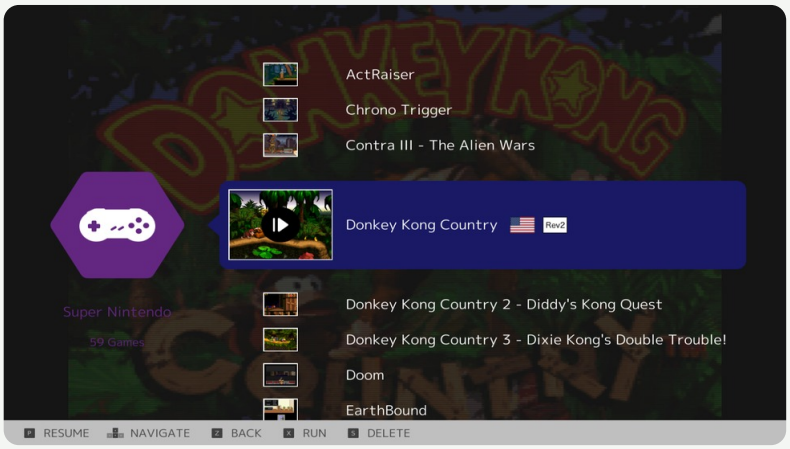
In emulation, some requirements concern the identity of the system being reproduced. The BIOS is a particularly clear case: it contains the emulated platform's firmware and, for cores that require it, participates in the intended operation. In RetroArch, a core is the component responsible for emulating a particular system. Its requirements may include specific firmware files.

Libretro's documentation highlights three practical elements: file location, name and hash. The directory identified in settings as System/BIOS provides the general reference; each core's instructions specify any required paths and conventions. The name lets the software recognise the file. The hash, a fingerprint calculated from its contents, allows comparison with the expected file: two identically named files may contain different data.

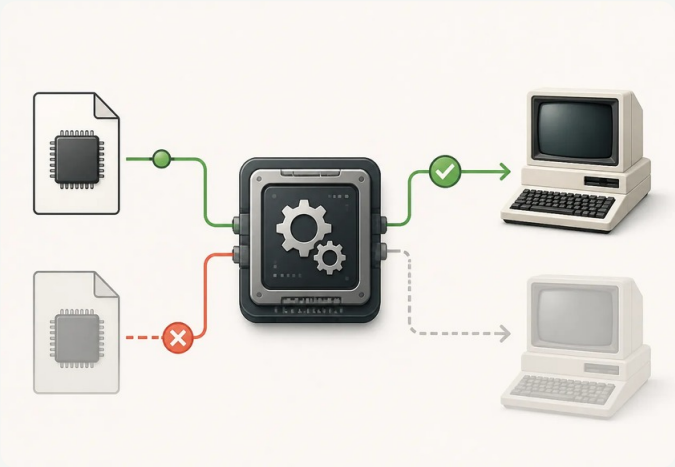
A mistake here can prevent startup even on a computer easily powerful enough to run the emulator. Diagnosis then concerns configuration and required files. Speaking vaguely of a PC being 'powerful enough' omits a decisive condition. Likewise, the BIOS requirement must be attributed to the selected core: applying it indiscriminately to every emulation solution makes the information less precise.

Preparing a system has costs of its own

Between download and startup lies a stage often given only a few lines in presentations: preparing the medium, writing the image and configuring boot. Raspberry Pi's



Source: Retro Home repository and build documentation



The correct firmware can be decisive in getting emulation to start. Conceptual illustration generated with AI for Retromind.

documentation illustrates this work through installation tools and different storage options. Raspberry Pi Imager prepares the medium; USB and, on compatible models and configurations, NVMe extend the available choices.

A list mentioning only 'Raspberry Pi' compresses many conditions. The supported model, boot medium and chosen image together define the result. A preconfigured image reduces manual steps and incorporates a selection of software and settings. Its convenience also depends on how closely that selection matches the reader's equipment.

The cost of getting started therefore includes time, skills and materials. Someone may already own the right computer for execution but still need a medium on which to write the image. Anyone compiling must also reserve space for source code and temporary build files. Separating these uses clarifies the scale of a requirement: tens of gigabytes occupied during preparation describe a different need from the space used by the installed system.

What a compatibility statement promises

The words accompanying requirements also establish the level of the promise. 'Required' identifies a necessary condition for the described procedure. 'Recommended' suggests useful headroom whose benefit should be explained. 'Tested on' documents a check limited to a configuration. Experimental support adds another qualification: the experience may require intervention and tolerance of unresolved issues.

These descriptions gain value when associated with observable conditions. Starting a program, completing a game and running consistently describe different results. The intervention users must perform also affects practical compatibility: a ready-made package and a procedure requiring recompilation reach audiences with different skills.

When covering homebrew or a conversion, each requirement should therefore be linked to its job. The historical machine's configuration explains where code executes. The development environment explains how the package is produced. Installation tools describe the route to startup. Sources may document these stages at different levels of detail; careful presentation preserves that distinction instead of extending a test beyond its documented conditions.

A modern requirement thus becomes information about a project and its audience. An author may choose an expanded configuration to support particular features, or provide a prepared image to simplify installation. Readers need to know the practical consequences: which machine they will use, which operations they must perform and what outcome they can expect. This precision makes even a technically demanding project possible to assess.

BLOG / 11.11.2026

CLASSIC-SYSTEM PROJECTS AND AI: TRACING DECISIONS ACROSS FORUMS, CHAT AND CODE

A proposal, a fix and a test may live in different places. AI can help reconstruct those steps, making work still awaiting a contributor visible.



Conceptual illustration recomposed with AI to fit this issue.

A player reports on a forum that a PlayStation homebrew game's audio disappears after pausing. Someone proposes a change on Discord; a pull request appears on GitHub; the wiki still recommends restarting the level. In this hypothetical scenario, each fragment tells part of the story. Knowing how to proceed requires understanding whether the change was accepted, who tested it and which instruction remains valid.

Generative AI can help reconstruct these connections. Its most useful result is a verifiable account of handovers: from report to proposal, from review to testing, and finally to updated instructions. Along this route a community can lose information and accumulate unfinished work.

One conversation contains different kinds of work

A project for classic systems can bring together highly specialised skills. Someone spotting a defect during play may know little about the code. Whoever prepares a fix may need someone able to reproduce the problem on original hardware. Another person maintains the documentation. Platforms collect traces of these contributions, while coordination depends on the connections people manage to preserve.

GitHub's documentation on communication distinguishes tools with different functions: issues support discussion and tracking of specific tasks, pull requests accompany proposed code changes, and Discussions host more open-ended conversations. Forums and chat provide further spaces. Each place's actual function then depends on the group's habits.

In the audio example, a forum reply may confirm that the defect is reproducible. A technical conversation may suggest its cause. The pull request makes a change available for examination. These contributions have different operational consequences: each enables a step and leaves different questions open.

A useful summary preserves these distinctions. Combining everything under 'problem solved' would erase precisely the information a newcomer needs: which step is complete and which still requires attention.

Ask the model where the work stops

The question put to AI determines much of the result. 'Summarise the project' invites an overview. A request focused on one activity may uncover more usable details: reconstruct work on the audio defect, connect every advance to a document and identify which tests are still outstanding.

This requires bounded material. Readers can provide the relevant thread, the discussion of the change and the wiki page, retaining original dates and links. The development branch a conversation concerns may also be decisive. A limited collection makes the result easier to check and reduces noise from unrelated discussions.

The model can collect messages describing the same defect in different words, extract proposed actions and connect a request for testing with a later response. It can also highlight a break: the material contains a change ready for testing but no account of running it on the console. This is an observable documentary gap. Whether a volunteer is available to undertake that work requires separate confirmation.

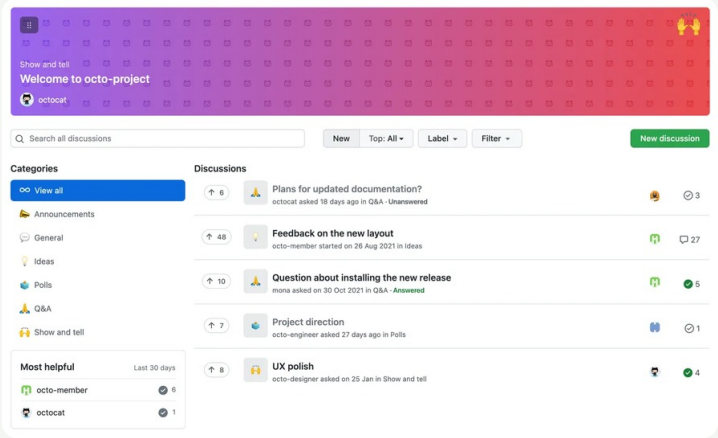
GitHub Projects best practices emphasise understandable descriptions, status updates and explicit dependencies. AI can help find these relationships when they are scattered through messages. The maintainer remains responsible for confirming them and setting priorities.

Dates make sense alongside a decision's status

A recent message may contain an attempt still awaiting assessment. An older wiki page may correctly describe the latest distributed version. Chronological order alone offers an incomplete reading of the work: the scope of each statement and its degree of adoption within the project also matter.

A pull request proposes a change. Merging it concerns a particular code branch; availability to players also depends on distributing a build that includes it. Similarly, a closed issue may refer to a duplicate report or another tracking activity. The meaning of closure emerges from the reasons and links in the discussion.

These distinctions matter particularly when testing is distributed among people with different equipment. In our example, an emulator test and a test on an original PlayStation document different execution conditions. A summary must preserve those differences and identify the tested build, avoiding a limited result being generalised to every situation.



Source: GitHub Docs - About discussions



A summary becomes useful when it returns to the right place in the project. Conceptual illustration generated with AI for Retromind.

Here AI risks making the story too linear: report, intervention, solution. A faithful reconstruction can remain open-ended. The change is recorded as merged; console testing is recorded as requested; the supplied documents contain no result. Wording of this kind lets readers distinguish a documented fact from a step still needing confirmation.

A summary must reach those who can use it

A reconstruction kept in a private conversation with an assistant has limited value to the group. Collective benefit emerges when someone checks it and returns it to the appropriate place: an issue update, a note in the technical discussion or a documentation change, according to the project's conventions.

In the hypothetical case, a contribution might explain that the fix has been merged into the development branch and requires a console test before updating the wiki. Links to the change and the test request would let a collaborator resume the activity. The short note would then have a precise function: connecting completed work with work available to undertake.

Assigning responsibility requires particular care. A very active participant may have offered an occasional suggestion. Message frequency says little about authority to approve a change or availability to perform a test. The model can extract explicitly stated assignments; any new assignment must be agreed with those involved.

Disagreements also deserve to remain visible. Two technical proposals may still be under consideration, with different consequences for the code. Preserving that open state helps whoever must decide and prevents an overly confident summary from being mistaken for group consensus.

A useful map declares its boundaries

The material available to an assistant represents only part of a project. Some discussions may take place in restricted channels; others may require authentication. If the model receives a forum thread and an issue, its conclusions must refer to those documents. The absence of a decision from the collection leaves open the possibility that it was made elsewhere.

The collection should also be limited to what is needed. Before transferring messages to an external service, consider participants' permissions and expectations and remove personal data unrelated to the analysis. Technical excerpts with original references are often enough. Dumping an entire history increases the material to inspect and may expose conversations unrelated to the problem.

Human checking can focus on the steps that actually change the work's state: acceptance of a proposal, a test's outcome and the instruction intended for users. These are the points at which an incorrect inference has consequences for other collaborators.

For maintainers of homebrew or an emulator, the practical benefit is the ability to formulate a specific request: this change awaits this test, with these references. Readers can understand how to contribute without restarting the entire investigation. AI will have done its job well if that handover becomes easier to verify and pass to the next person.

FROM THE ARCHIVE / 19.08.2026

GOLDENEYE 007 REACHES 100% DECOMPILATION: WHAT CHANGES FOR THE NINTENDO 64 CLASSIC

After nine years of reverse engineering, reconstructed code has achieved a complete match with the original game. The result opens possibilities for technical study, preservation and derivative projects.



Conceptual illustration recomposed with AI to fit this issue.

After nine years of work, the GoldenEye 007 decompilation project has reached a 100% match with the Nintendo 64 game's code. As reported by Time Extension, the updated public tracker indicates that all tracked code can now be rebuilt consistently with the original executable.

This is an important achievement for the community around Rare's shooter, but it needs qualification: complete decompilation does not mean the source code written by the developers in the 1990s has been found. It is a reconstruction starting from the compiled program, produced through lengthy analysis, verification and comparison.

What reaching 100% means

A game cartridge contains instructions already transformed into machine code, ready for the hardware to execute. Compilation loses many elements present in the development environment: comments, the original names of many functions and variables, file organisation and some information useful for understanding the authors' intentions.

A decompilation project attempts to reconstruct a readable source base, generally in C with possible assembly sections. The code is compiled using tools and settings compatible with those of the period, then compared with the program preserved in the ROM. A function is considered matching when compilation produces the same machine instructions as the original.

The process demands extreme precision. A different declaration, operation order or apparently equivalent choice can cause the compiler to generate different instructions. Visible behaviour may be identical while binary comparison still detects a mismatch. Reaching 100% means resolving these remaining differences

throughout the portion of code examined by the project.

The achievement remains distinct from recovering Rare's internal source. Names assigned during reconstruction may be modern or inferred from the program's behaviour, and original comments remain absent. Its technical value comes from being able to reproduce the software and study its structure in an understandable form.

Why GoldenEye is a particularly interesting case

Released in 1997, GoldenEye 007 turned the film into a shooter built around objective-based missions, difficulty levels that changed assignments and celebrated local split-screen multiplayer. The game played an important role in establishing first-person shooters on consoles at a time when the genre was still associated chiefly with PCs.

Its production was closely tied to the Nintendo 64's characteristics. The console used a MIPS-based processor and components dedicated to graphics and audio. Developers worked within specific limits of memory, cartridge capacity and available power, organising code around the hardware and development tools of the time.

Reconstructing such a program therefore requires expertise across several levels: Nintendo 64 architecture, compiler behaviour, data formats and internal engine logic. GoldenEye 007 involves complex systems such as character AI, objective management and environment rendering. A complete match makes these components more accessible to technical analysis.

Nine years of software archaeology

The project's duration helps convey the scale of the undertaking. Decompilation proceeds function by function. Contributors identify instruction blocks, propose possible C versions, compile them and observe the differences. Each correction brings the reconstructed code closer to the original sequence.

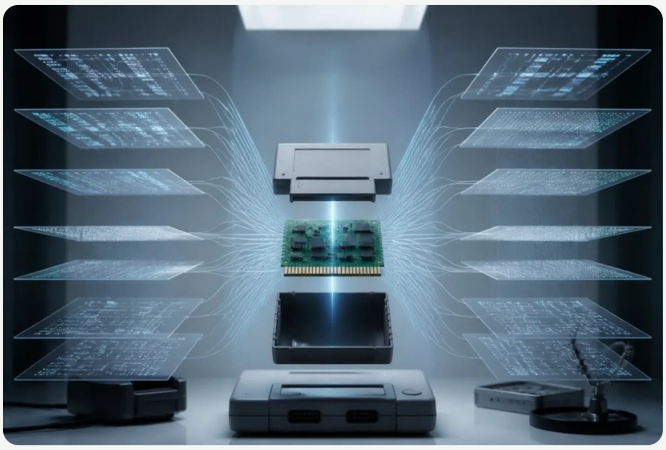
The work also includes defining data structures, reconstructing relationships between subsystems and finding the correct toolchain configuration. Unfinished parts tend to become progressively harder: unusually optimised functions, hardware-specific routines or sections where compiler behaviour is less intuitive remain.

The 100% figure therefore represents thousands of repeatable checks. It is also a collective achievement supported by procedures and tools developed by the Nintendo 64 decompilation scene. In recent years this field has produced increasingly accurate reconstructions, turning reverse engineering into an organised form of technical documentation.

Its value for preservation

Keeping a ROM preserves the digital object distributed to the public. A matching source base adds another layer of preservation: it reveals how the software organises its functions, communicates with the machine and solves platform-specific problems.

This information can help researchers, emulator developers and scholars of videogame production. Code makes details hidden during



Complete decompilation shows how reconstructed code can match the original program.

GOLDENEYE 007 / 3 OF 3

CONTINUED

play verifiable and provides concrete material for analysing Rare's techniques. It becomes possible to trace an input, understand character handling or examine the conditions governing a mission.

Decompilation also helps preserve knowledge of development tools. Reproducing the original result often requires precise compiler versions and configurations compatible with those used in the 1990s. Documenting that chain preserves part of the production process alongside the finished game.

From binary matching to future projects

A complete base can facilitate native ports, modifications and analysis tools. Developers now have a readable representation of the game's logic instead of having to work directly on ROM instructions. This reduces the reverse engineering needed to experiment with new resolutions, control systems or technical fixes.

Reaching 100% does not automatically produce a PC version or an enhanced edition. Porting requires further work: calls to Nintendo 64 hardware must be adapted, rendering moved to modern interfaces, and every change tested on different platforms. Decompilation supplies a much stronger foundation for tackling those steps.

The official rereleases available since 2023 on Nintendo Switch and Xbox consoles serve a different need. Those versions make the game accessible again through contemporary channels; the decompilation project offers a reproducible tool for study that the community can potentially extend.

Rights in the work, brand and audiovisual material remain in force. In projects of this kind, reconstructed code is normally kept separate from assets extracted from the game, and building a working version may require a ROM copy supplied by the user. Technical correspondence of the code does not change the original content's legal status.

A new level of access to Rare's history

GoldenEye 007 is chiefly remembered for its impact on players and multiplayer sessions. Decompilation lets those memories sit alongside a more precise analysis of development choices. The game can be studied as a product of a particular hardware architecture, a specific toolchain and solutions devised to fit an ambitious design within the Nintendo 64's limitations.

The project's completion also offers a useful basis for comparison with Rare's subsequent productions, beginning with Perfect Dark. The evolution of game systems and internal technology can be reconstructed through verifiable evidence, combining historical accounts with direct observation of software.

After nine years, the most enduring result is this new readability. The ROM preserves the game published in 1997; reconstructed code allows a deeper understanding of how it works and gives the community tools to continue studying, documenting and developing initiatives around one of the Nintendo 64's most representative titles.

Sources

GoldenEye 007 for Nintendo 64 is now 100% decompiled

FROM THE ARCHIVE / 16.08.2026

BRINGING ARCADES TO AMIGA AGA: THE CREATIVE ENGINEERING OF FAN CONVERSIONS

An arcade conversion for Amiga AGA must translate graphics, memory use and the rhythm of play into a profoundly different architecture.

Projects such as Double Dragon reveal how much technical work lies behind the pursuit of fidelity.



Conceptual illustration recomposed with AI to fit this issue.

Double Dragon brings together almost every challenge an Amiga conversion must face: scrolling backgrounds, several animated characters, frequent collisions, graphics priorities and control response that determines the rhythm of each encounter. Reproducing this structure on Amiga AGA means rebuilding the game according to hardware rules different from those of the arcade board.

A fan conversion thus becomes a form of technical translation. The contemporary Double Dragon project provides a concrete entry into a wider phenomenon in which fidelity grows from decisions about memory, graphics, execution timing and DMA-channel load. The result remains interesting while in development because it reveals the process of studying a historical machine and pursuing goals that commercial conversions of the period achieved with widely varying success.

Two architectures, two ways to build the same scene

An arcade board is designed around a defined set of needs. It may have specialised circuits for backgrounds, tiles, sprites, priority layers and scrolling. The game code knows and uses that configuration directly. A home-computer conversion must break down each function and find a plausible equivalent on the target hardware.

For a scrolling beat-'em-up, similarity between individual frames is only part of the job. Enemy count, animation, hitboxes, AI, character-overlap order and continuity of the scenery must all be managed. These elements share resources and must fit the time available to produce each frame.

AGA considerably expands graphics capabilities compared with earlier Amiga chipsets. It offers a larger palette, more bitplanes in indexed modes and more

flexible data-fetch modes. The underlying architecture nevertheless retains the Amiga's classic organisation. Graphics, audio, sprites and blitter operations depend on Chip RAM and its DMA accesses. The chipset's greater capability provides headroom; conversion quality still depends on how that headroom is managed.

Chip RAM is also a scheduling problem

Amiga memory concerns both available space and when each component can access it. The custom chips read screen bitplanes, audio data, Copper structures and sprite information from Chip RAM. The blitter also operates on data in this area. The CPU therefore shares the bus with activities taking place while the image is being generated.

A colourful background requires more bitplanes. Each additional plane increases the data read to construct the display and affects available bandwidth. More animations and objects require further memory for graphics frames, masks and game structures. If the chosen configuration includes Fast RAM, some CPU-only code and data can move outside shared memory. Defining the minimum machine therefore remains a decisive design choice: targeting a basic Amiga 1200 imposes different criteria from an accelerated, expanded configuration.

Planar format adds another translation stage. A pixel's colour comes from bits at the same position across several separate planes. Many modern assets and arcade systems are more conveniently described through complete pixels or tile structures. Before drawing anything, developers must convert palettes, images and masks into data suitable for the Amiga chipset. Memory layout can make the difference between efficient updates and a sequence of costly copies.

Blitter, sprites and Copper as staging tools

The blitter can copy memory blocks, combine sources through logical operations and assist with drawing graphics. Its ability to operate in parallel with the CPU makes it valuable, provided work is scheduled with bus contention in mind. A long series of blits started at the wrong moment can take bandwidth from other activities. Careful planning instead allows the CPU to handle logic while the chipset prepares parts of the image.

Hardware sprites can ease some operations, but their number, size, colour and positioning limits call for selective use. The Copper changes graphics registers at precise positions in the video frame, enabling palette changes, split screens and other techniques. Each gains value within an overall strategy: backgrounds, characters, interface and audio must coexist within the same memory and time budget.

For Double Dragon, choices also concern how fighters are represented. Large, well-animated characters consume space, bandwidth and compositing operations. Reducing animation frames makes management lighter but changes the readability of attacks. Reducing enemy numbers simplifies the scene while changing the pressure on the player. A convincing conversion seeks a resource balance that preserves the game's recognisable behaviour.



An arcade conversion's complexity comes from coordinating animation, collisions and graphics priorities.

Fidelity has several layers

Comparisons with the arcade game tend to focus on screenshots. Colours, character sizes and background detail are immediately visible. Stricter criteria emerge during play: movement speed, animation duration, the interval between input and action, collisions, enemy numbers and the progression of scrolling.

Visual fidelity can come at an excessive cost if each frame requires too many operations. A slightly simplified presentation may sustain a rhythm closer to the original. Audio belongs to the same balance because DMA channels and sample memory contribute to the overall budget. Porting becomes a series of connected judgements in which each local improvement is assessed against the complete experience.

There is also perceptual fidelity. Players recognise Double Dragon through the cadence of encounters, opponents' positions, the weight of blows and continuity of progress. A fan conversion can reconstruct these relationships even with internal solutions far removed from the arcade board. Code changes, data is reorganised and some assets are adapted; the aim remains a response consistent with the original model.

Modern development changes the tools

Contemporary conversions have cross-compilers, graphics editors, asset-conversion scripts and automated systems for producing disk images or bootable builds. Emulators with debuggers help examine registers, memory and execution timing. Testing on real hardware remains essential for checking video, audio, peripherals and memory configurations.

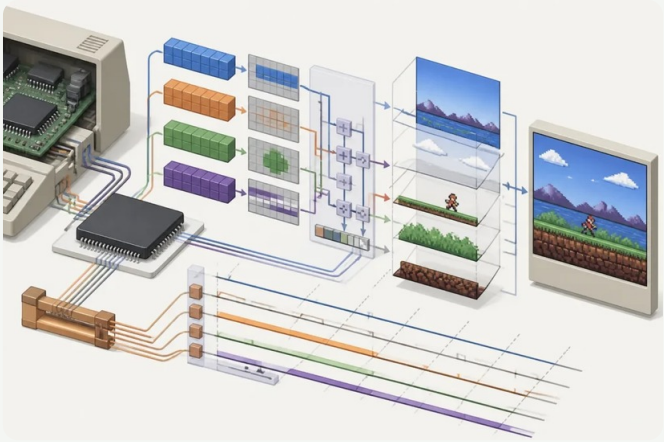
This toolchain reduces repetitive operations and shortens the cycle between changing and testing. The knowledge required remains deeply tied to the machine. An image converted to the correct format may still be too large, badly placed or expensive to update. An engine apparently smooth in an empty scene may slow down as enemies, effects and audio samples accumulate. Optimisation grows from observing these concrete cases.

Progressive releases also serve a technical purpose. Demonstrations and betas allow projects to be tested on different expansions, hardware revisions and video modes. Community feedback reveals incompatibilities and situations difficult to reproduce in a development environment. Double Dragon's project page documents contemporary work and its evolution; implementation details should be attributed to the developer when they become available.

A working archive of Amiga expertise

The terms fan conversion, remake and enhanced version often overlap. A conversion recreates a title for another platform; a remake may rewrite its technology and content; an enhanced version modifies an existing base. Combinations are common in homebrew because a project can reuse assets, replace the engine and introduce improvements designed for expanded configurations.

Their cultural value comes from making the work observable.



Blitter, sprites and Copper work together to compose the image within the available time.

ARCADE CONVERSIONS / 4 OF 4

CONTINUED

68000 code, planar graphics, blitter scheduling and Chip RAM organisation become applied skills again. Historical documentation, recent tools and collective testing converge in an executable object. Preservation thus also includes the ability to design for the hardware, alongside keeping disks and manuals.

A successful AGA fan conversion interprets an arcade game through the Amiga's particular possibilities. Compromises reveal which aspects are considered essential and which machine resources can support them. Double Dragon is a particularly clear case because so many systems must work together. Behind each scrolling background and group of opponents lies a discipline of measurement, priorities and verifiable solutions: a workshop in which the Amiga architecture continues to be understood through use.

Sources and further reading

Amiga Hardware Reference Manual - Amiga Memory System

Amiga Hardware Reference Manual - What is the Blitter?

Amiga Hardware Reference Manual - DMA Control

Double Dragon by jotd666 on itch.io

Indie Retro News - Double Dragon AGA beta update

FROM CODE TO EXPERIENCE

CODE | **HARDWARE** | **TESTING**

Archives, tools and communities keep the history of games and machines open.

THE PROJECT AND SOURCES.

Retromind is an experimental editorial project. AI agents select, adapt and lay out this PDF according to editorial criteria defined by people. Content comes from Retromind and the cited sources. AI-generated or reworked images are identified in their captions. The declared human involvement concerns the definition of editorial criteria; any human review of an individual issue is stated only when it has actually taken place.

This English edition translates the complete Italian selection and adds one introductory preface. Cover artwork is shared with the Italian edition of the same issue. Conceptual illustrations come from Retromind material or have been recomposed with AI as captioned. Screenshots and documents retain their original credits. No editorial image is repeated within this edition.

09 OutRun AGA: RetroDNA's conversion for the Amiga 1200
12.11.2026

13 Insector X on Amiga: an AGA conversion built with static recompilation
12.11.2026

16 Time Pilot on Amiga: translating Konami's arcade game from Z80 to 68000
08.11.2026

19 Amiga: Vesuri improves frame rates and compatibility in classic games
07.11.2026

22 Karsten Obarski, Soundtracker and the music of Amiga cracktros
06.11.2026

25 Special Forces: the Psygnosis project rediscovered in documents from 1997
10.11.2026

28 Mega Man Legends: a 'new media' listing and the possibility of a collection
11.11.2026

30 Puzzle Mania XXXL: an MSX1 puzzle game with ROMs up to 128 MB
09.11.2026

32 Fan patches: keeping an edition playable over time
07.11.2026

35 Projects for classic systems: the role of modern hardware in their requirements
09.11.2026

38 Classic-system projects and AI: tracing decisions across forums, chat and code
11.11.2026

41 GoldenEye 007 reaches 100% decompilation: what changes for the Nintendo 64 classic
ARCHIVE / 19.08.2026

44 Bringing arcades to Amiga AGA: the creative engineering of fan conversions
ARCHIVE / 16.08.2026

ISSUE 01 / ENGLISH EDITION

Source issue: 13 November 2026. Selection closed: 12 November, 15:53 CET. English edition prepared: 14 November 2026.

B.03 layout: Barlow Condensed, Atkinson Hyperlegible Next and Chakra Petch. Two- and three-column galleys, embedded fonts and English hyphenation. A4 PDF, 50 pages.

Complete translated contributions; original publication dates and source links retained.
Original articles, documentation and full credits: www.retromind.it.

BEYOND THE LAST ARTICLE

CONTINUE EXPLORING.

A few paths beyond the page: projects to follow, sources to explore and another issue to look forward to.

01

PLAY

Follow the projects behind the pages.

Explore community releases and development reports. Check each project's notes for compatible hardware, versions and installation requirements.

Double Dragon AGA / project page >

Insector X / development report >

02

DISCOVER

Go deeper into code and machines.

Return to the documentation and reporting behind this issue: how the Amiga shares its memory, and what reconstructing GoldenEye 007 means.

Amiga Hardware Reference Manual / memory >

GoldenEye 007 / decompilation report >

03

FOLLOW RETROMIND

Keep a place for the next issue.

Read the site between issues and revisit the magazine archive. The weekly magazine is published on Fridays at 09:00 Italian time.

English edition / previews and downloads >

Retromind / news, features and blog >

Many original Retromind articles are in Italian. External links open the original project, documentation or reporting; availability and requirements may change.

13 NOVEMBER 2026

THE STORY STAYS OPEN.

From media to code, from memory to a new world.



Original back-cover illustration generated with AI for this issue.

Retromind

WWW.RETROMIND.IT

EXPERIMENTAL EDITORIAL PROJECT WITH AI AGENTS