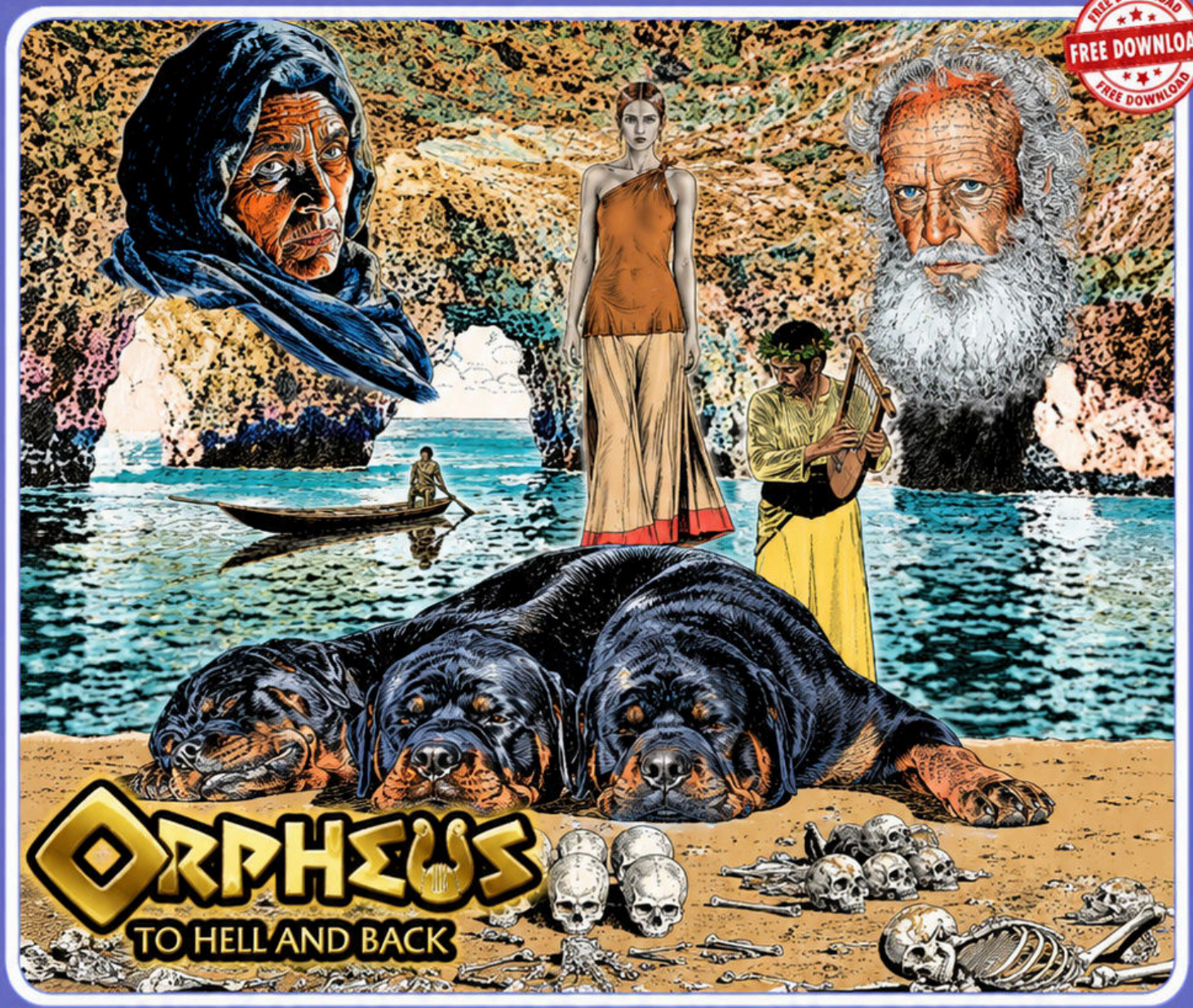
PROGRAMMAZIONE - STORIA - INTERVISTE - RECENSIONI - APPROFONDIMENTI

RETRO MAGAZINE WORLD

N°55

future days are back

www.retromagazine.net



SPECIALE AMOS: convertire AFTERMATH OF THE ASIMOVIAN DISASTER

Speciale VIC-II: Come sfruttare il VSP Bug del C64

Demoscene: intervista ad "ice00" aka Stefano Tognon

Speciale eventi retro: Pianeta Amiga e Varese Retrocomputing

10.5281/zenodo.23510697

ISSN 3103-697X

L'asticella si sta alzando!

Di cosa sto parlando? Ma della qualità degli eventi dedicati al retrocomputing in Italia. Chi ha avuto modo ultimamente di partecipare a qualcuno di questi eventi avrà probabilmente avuto la mia stessa impressione: il livello di questi raduni si è decisamente innalzato. Da eventi di nicchia, appannaggio di pochi retroappassionati, gli eventi degli ultimi mesi si stanno trasformando sempre più in appuntamenti imperdibili per chiunque: per l'appassionato che ha modo di vedere oggetti rari e incontrare persone disposte a raccontare le loro imprese e per il visitatore occasionale che ha modo di scoprire un mondo nuovo e, perché no, appassionarsi a qualcosa che fino al giorno prima non conosceva.

Ma c'è di più! Negli ultimi tempi sono stato testimone di una cosa straordinaria: intere famiglie, attratte dalla folla che riempiva alcuni di questi eventi, si sono affacciate a sbirciare e, incantate dall'ambiente che hanno trovato, hanno avuto modo di trascorrere una giornata diversa dal solito.

Che significa tutto questo? Che probabilmente gli organizzatori hanno finalmente trovato il sistema per avvicinare il grande pubblico a un pezzo di storia italiana, senza però rinunciare alla qualità comunicativa che l'appassionato di lunga data ricerca in questo tipo di occasioni.

Chi scrive ha partecipato in maniera più o meno attiva a un paio di occasioni veramente interessanti che si sono svolte in Toscana nei mesi di settembre e ottobre: il **Pianeta Amiga** di Empoli e il **Flashback Festival** di Follonica.

Quando sono tornato a casa da entrambi questi appuntamenti, ho avuto la sensazione di aver vissuto qualcosa di speciale.

Sono tornato arricchito sotto diversi punti di vista: ho conosciuto persone, ho condiviso esperienze, ho imparato cose nuove e al tempo stesso mi sono divertito come quando ero ragazzino con i miei amici di fronte al Commodore 64 o all'Amiga.

Non vi parlerò di Pianeta Amiga, in quanto potete leggere il resoconto di Carlo Savorelli in questo stesso numero, ma mi preme spendere due parole sul Flashback Festival di Follonica.

L'evento si è svolto a Follonica nelle giornate di sabato 3 e domenica 4 ottobre. Sin dall'arrivo era chiaro che si trattasse di qualcosa di speciale: una decina di auto, tra cui la DeLorean e KITT di Supercar, ci accoglievano all'ingresso. All'interno della magnifica location della Fonderia 1, esposizioni di calcolatrici e computer storici, cabinati arcade e console di tutti i tipi facevano bella mostra di sé. Il tutto arricchito da conferenze tenute da relatori d'eccezione: Leonardo Vettori e Lorenzo Mancini, Federico Gori e Gastone Garziera, giusto per nominarne alcuni...

Ah, dimenticavo. Non perdetevi l'articolo di Filippo Occulto dedicato al **Varese Retrocomputing**, un altro appuntamento incredibile a cui purtroppo non ho potuto prendere parte.

In sintesi, il retrocomputing italiano sta dimostrando di saper unire memoria, divulgazione e spettacolo.

Se questo è il livello, non resta che augurarsi che l'asticella continui a salire.

Francesco Fiorentini

SOMMARIO

	PAG.
◊ Le occasioni mancate	Pag. 3
◊ Il mio primo computer: SV-318	Pag. 9
◊ Costruire un'avventura testuale con un narratore AI	Pag. 14
◊ GentleOS - Un OS moderno per PC a 16/32 bit	Pag. 17
◊ L'arte di forzare l'hardware: il VSP bug del C64	Pag. 20
◊ LigthTreeFM - un file manager per AmigaWB con l'IA	Pag. 28
◊ Reinventare un classico senza tradirne l'anima	Pag. 36
◊ Il cavalier de Mérè (C64 principianti)	Pag. 42
◊ Deluxe Paint: quando dipingere coi pixel era magia	Pag. 48
◊ Dove ho messo quel tal programma?	Pag. 50
◊ Tutto su un floppy! HamsterOS	Pag. 54
◊ POKE64 - il C64 su... iPad!!!	Pag. 59
◊ Mean Hamster Software - speciale 40 anni	Pag. 61
◊ La storia della System 3: 4 decenni controcorrente	Pag. 65
◊ Shadow Snare, la promessa dell'83 che torna sul C64	Pag. 72
◊ Intervista a Michael Cranford	Pag. 74
◊ Demoscene - Intervista ad Ice00 aka Stefano Tognon	Pag. 77
◊ eSports, le origini: conversazione con B. Baumgart	Pag. 90
◊ Varese Retrocomputing: dalle origini ad oggi	Pag. 104
◊ Pianeta Amiga: la conferenza di Michele Battilana	Pag. 108
◊ Razing Core (Sega Master System)	Pag. 110
◊ Xibalba 64 (Nintendo 64)	Pag. 114
◊ Rogucraft DX (PC Windows)	Pag. 117
◊ L'Abbaye des Morts (ZX Spectrum Next)	Pag. 120
◊ Orpheus: To Hell And Back (Game Boy)	Pag. 122
◊ Kaboomania (Amiga, PC/XBox, PS, SwOS, macOS)	Pag. 124
◊ Mappy (C64, C64U)	Pag. 126
◊ Worms: The Director's Cut 1.5 (Amiga AGA)	Pag. 128
◊ Tetris Basic (Amstrad CPC)	Pag. 130
◊ Space Panic DX (Vic-20)	Pag. 132
◊ Super Mario Bros. (SMS)	Pag. 134
◊ Bubble Fight (Master System, SG-1000, SC-3000)	Pag. 136
◊ Lemmings Reloaded (C64, C64U)	Pag. 139
◊ Rogul (Atari ST/STE)	Pag. 142
◊ Final Whiistle (XBox, PC Windows)	Pag. 144
◊ Pitfall II (Vic-20)	Pag. 146
◊ Snapman-20 (Vic-20)	Pag. 148
◊ Binky: Frozen in Time (C64)	Pag. 150
◊ Yokoxevi (MSX)	Pag. 152
◊ Side Arms (Mega Drive)	Pag. 154
◊ Worm King (Game Boy)	Pag. 156
◊ Cannon Fodder (Atari STE)	Pag. 159
◊ Hyper Runner (Amiga AGA)	Pag. 162
◊ Tiny Quest (Amstrad CPC, Atari ST/Falcon)	Pag. 165
◊ Wolf64 (C64)	Pag. 168
◊ Babylonian Twins (Amiga)	Pag. 171
◊ Dracula's Castle (Super Game Gear)	Pag. 174
◊ Ruote Fumanti (C64)	Pag. 175
◊ Void Lancer (C64)	Pag. 176
◊ Wolf (GBC)	Pag. 177

• Hanno collaborato:
Francesco Fiorentini, **Carlo Del Mar Pirazzini**, **Tommaso Piitturru**, **Eugenio Rapella**, **Jari Jalonen**, **Takahiro Yoshihoka**, **Roberto Lanciotti**, **Ingrid Poggiali**, **Chiara Carrara**, **Alex Fridman**, **Daniele Brahimi**, **David La Monaca**, **Marco Figaro**, **Danllo Savioni**, **Felice Nardella**, **Christopher Drum**, **Giampaolo Moraschi**, **Roberto Del Mar Pirazzini**, **Filippo Occulto**, **Marta Rossmann**, **Leonardo Vettori**, **Carlo Savorelli**, **Lorenzo Mancini**

• Immagine di copertina:
Giuseppe Mangini
(soggetto ispirato al gioco Orpheus To Hell And Back - Game Boy)

• Layout di copertina:
Carlo Del Mar Pirazzini
(un omaggio alla rivista Videogiochi n.15 - maggio 1984)

Le occasioni mancate

Anni '80: il decennio delle scelte sbagliate

di Tommaso Pitturru

Attenzione, questo articolo prosegue il filone "ucronico" (ipotesi che un evento storico sia andato in modo diverso rispetto alla realtà) dei precedenti contributi di Tommaso, allargando lo sguardo all'intero decennio. Le immagini dell'articolo sono generate con tool AI.

La storia dell'informatica viene quasi sempre raccontata come una marcia trionfale in cui le tecnologie migliori sono sopravvissute, mentre quelle deboli si sono estinte. È una favola consolatoria, ma priva di fondamento: se si esaminano da vicino gli anni Ottanta, l'evoluzione del personal computer assomiglia molto più a un campo di battaglia in cui a prevalere non sono stati i progetti più eleganti o visionari, ma quelli sostenuti dalla forza delle politiche industriali, dalle guerre dei prezzi o da spietate, e spesso miopi, strategie commerciali.

Dietro la piatta omologazione del mondo IBM-compatibile e dell'interfaccia Windows – destinati a dominare il decennio successivo mentre una Apple in affanno tentava faticosamente di difendere la propria nicchia –, rimangono tra le macerie progetti rivoluzionari e occasioni colpevolmente sprecate.

Ripercorrendo quegli anni con la lente del what-if, emerge la sensazione che, per cambiare il corso degli eventi e inaugurare un'epoca informatica infinitamente più stimolante, non servissero chissà quali miracoli scientifici. Per questo motivo, per ciascuno dei casi esaminati in questo viaggio, abbiamo dedicato un paragrafo intitolato «Sarebbe bastato...». Niente di particolarmente speculativo, ma l'individuazione di quel singolo POD, point-of-divergence corrispondente a quella specifica scelta progettuale, commerciale o manageriale che avrebbe potuto riscrivere per sempre il corso della storia informatica.

Il sabotaggio interno: Apple IIGS (1986)

Tra tutti i paradossi degli anni Ottanta, forse il più clamoroso non fu consumato da un'azienda al collasso, ma dalla Apple di John Sculley. Nel settembre del 1986 vide la luce l'Apple IIGS, una macchina che incarnava il punto d'incontro ideale tra la sterminata eredità software dell'Apple II e il futuro del personal computing multimediale.

Abbandonato l'ormai vetusto 6502 a otto bit di MOS Technology (e il suo derivato 65C02), si decise di puntare tutto sul ben più performante Western Design Center 65C816: un microprocessore a 16 bit con spazio di indirizzamento a 24 bit (capace di gestire fino a 16 MB di RAM) e piena compatibilità col predecessore. Il comparto grafico, gestito dal Video Graphics Controller (VGC), offriva una palette di 4.096 colori con risoluzioni fino a



640×200 punti. Ma il vero capolavoro risiedeva nell'audio: Apple integrò il chip Ensoniq 5503 DOC (Digital Oscillator Chip), un generatore wavetable a 32 oscillatori indipendenti progettato da Bob Yannes — lo stesso creatore del SID del Commodore 64. Il IIGS disponeva di un sonoro a campionamento digitale a 15 voci che faceva sembrare il "beeper" standard dell'IBM PC e i segnali a onda quadra dei concorrenti reperti storici.

Completato da un ambiente WIMP a colori con finestre, icone e menu a tendina già visti sul Macintosh (al lancio su ProDOS 16, dal 1988 sul più evoluto GS/OS a 16 bit), il IIGS si presentava come la macchina da sogno per l'intrattenimento, la didattica e la creatività personale.

All'interno di Cupertino, tuttavia, infuriava la guerra tra la divisione Apple II e quella Macintosh. Il Mac, nel 1986, era ancora una macchina monocromatica, chiusa, limitata a uno schermo integrato da 9 pollici in bianco e nero e audio monofonico. La dirigenza temette che un Apple II così avanzato ed economico potesse cannibalizzare le vendite del neonato Macintosh aziendale. Approccio che Apple non ha mai del tutto abbandonato, basta vedere le temute sovrapposizioni tra iPad e MacBook che almeno finora hanno frenato in questi ultimi l'adozione di un touch screen OLED. Il 65C816, già disponibile a 4 MHz e destinato a crescere rapidamente di frequenza, venne tenuto a 2,8 MHz per motivi tecnici probabilmente superabili, l'espandibilità fu frenata e il marketing fu ridotto ai minimi termini per contenerne la diffusione a discapito del più prestigioso, e costoso, Macintosh.

Sarebbe bastato... non castrare il 65C816 a 2,8 MHz, montare una CPU a pieno clock (4 MHz già al lancio) e avere il coraggio industriale di puntare sull'Apple IIGS come piattaforma multimediale a colori di massa, e lasciare che il Mac crescesse in base alle richieste degli utenti. Il mercato consumer americano ed europeo avrebbe incoronato Apple regina dei personal computer 16 bit prima ancora che Commodore consolidasse l'Amiga 500, anticipando di un decennio l'informatica multimediale su scala globale.

L'Arcade irraggiungibile: Sharp X68000 (1987)

Mentre l'Occidente si divideva nello scontro tra Amiga 500 e Atari 520ST, in Giappone nel 1987 la Sharp presentava un computer che mostrava in un colpo solo di cosa fosse capace la tecnologia nipponica dell'epoca: l'X68000.

Racchiuso nel celebre chassis a doppia torre con maniglia a scomparsa, l'X68000 montava un clone Hitachi del Motorola 68000 con frequenza a 10 MHz, 1 MB di RAM e, soprattutto, un sottosistema video completamente separato con 1 MB di VRAM indipendente. La macchina offriva risoluzioni non interlacciate fino a 768×512 punti, una palette a 16 bit da 65.536 colori totali, più o meno visualizzabili a seconda della risoluzione scelta, e un generatore hardware capace di muovere 128 sprite a 16 colori contemporaneamente, su piani multipli e a scorrimento





fluido. Il comparto audio univa la sintesi FM a 8 canali dello Yamaha YM2151 (lo standard assoluto delle sale giochi) all'OKI MSM6258 ADPCM a 10 bit per i campionamenti vocali.

L'architettura era talmente vicina all'hardware da sala giochi che l'X68000 venne utilizzato come piattaforma di sviluppo per le schede arcade: capolavori come Ghouls 'n Ghosts e Final Fight ricevettero conversioni domestiche inarrivabili in Europa e negli Stati Uniti.

L'X68000 rimase tuttavia un gioiello inaccessibile. Sharp compì tre errori fatali: fissò il prezzo a oltre i 3.000 dollari dell'epoca, isolò l'hardware attorno ai costosi monitor a frequenza multipla (15/24/31 kHz) e alle Kanji ROM, inutili fuori dal Giappone, e affidò l'esperienza utente a Human68k, sistema in stile DOS accompagnato solo dal rudimentale gestore grafico Visual Shell, rimandando il vero ambiente a finestre (SX-Window) al 1989.

Sarebbe bastato... stringere un accordo con un partner internazionale (come Sony o Philips), dotare la macchina di una ROM internazionale, integrare nativamente in ROM l'ambiente grafico GEM di Digital Research (esattamente come fece Atari), o analogo, e commercializzare una versione monoblocco – Amiga e Atari style - intorno ai 1.000 dollari. L'X68000 avrebbe cancellato il divario tra sala giochi e salotto, e avrebbe seriamente messo in pericolo il monopolio Apple nel Desktop Publishing professionale a colori.

La miopia di Commodore: Amiga e l'illusione 2D

Il Commodore Amiga (1985) incarnava il capolavoro architetturale di Jay Miner: i chip custom Agnus, Denise e Paula scaricavano la CPU da ogni compito multimediale grazie a Blitter, Copper, a una palette di 4.096 colori e all'audio a quattro canali stereo campionati via DMA, il tutto orchestrato dall'AmigaOS con un vero multitasking preemptivo.

La tragedia di Amiga fu la gestione aziendale di Commodore. Paralizzata da lotte intestine e da un management privo di visione tecnica, l'azienda lasciò invecchiare l'architettura OCS per anni, limitandosi nel 1990 al modesto aggiornamento ECS. Il computer rimase ostaggio di un'architettura a RAM condivisa — dove l'attività dei coprocessori video sottraeva cicli di clock preziosi alla CPU 68000 — e di una struttura grafica a bitplane, perfetta per i titoli 2D a scorrimento orizzontale parallattico ma punitiva per il calcolo tridimensionale che avrebbe dominato gli anni Novanta. Quando nel 1992 arrivò tardivamente il chipset AGA, la finestra storica si era chiusa: la mancata evoluzione verso una grafica poligonale nativa trascinò Commodore al fallimento del 1994, lasciando campo libero ai PC equipaggiati con schede VGA e acceleratori grafici.

Sarebbe bastato... che la dirigenza Commodore non congelasse la ricerca e sviluppo tra il 1987 e il 1990, finanziando tempestivamente la transizione a un'architettura



a 32 bit pensata per il 3D (per la quale esisteva già il progetto del chipset AAA, avviato nel 1988). Amiga avrebbe guidato la rivoluzione del 3D poligonale e del video digitale prima che sui cloni IBM basati su Windows approdassero le prime schede 3dfx, combinazione destinata a monopolizzare il mercato.

La standardizzazione tradita: il miraggio dell'MSX3 (1990)

Nel primo articolo di questa serie ucronica avevamo già esplorato l'universo MSX, immaginando uno scenario fanta-ucronico nel quale un prototipo segreto a doppio Z80 presentato a Kazuhiko Nishi nell'autunno del 1984 avrebbe traghettato lo standard verso i 16 bit e il multitasking prima dell'Amiga. In quell'occasione ci eravamo fermati alle soglie dell'MSX2, senza addentrarci in ciò che accadde davvero al tramonto del decennio: la vera occasione mancata dello standard, ovvero il progetto dell'MSX3.

Nato nel 1983 dall'intuizione di Nishi per dare all'informatica un ecosistema aperto e compatibile sostenuto da colossi come Sony, Panasonic, Toshiba e Philips, l'MSX si era evoluto regolarmente con l'MSX2 (1985) e l'MSX2+ (1988). Al posto di quest'ultimo il programma iniziale prevedeva che alle soglie degli anni Novanta lo standard sarebbe evoluto verso il mai nato MSX3.

Per superare i limiti dello Z80 senza spezzare la retrocompatibilità con l'immenso parco software preesistente, e in assenza di un successore adatto allo scopo del celebre processore prodotto da Zilog, ASCII Corporation fu costretta a creare una CPU proprietaria straordinaria: l'ASCII R800. Pilotato da un master clock a 28,636 MHz, l'R800 lavorava a una frequenza operativa reale di 7,159 MHz ma, grazie a una pipeline interna ottimizzata che eseguiva molte istruzioni in appena uno o due cicli, garantiva l'efficienza computazionale di uno Z80 virtuale spinto a quasi 29 MHz. Dotato di un moltiplicatore hardware a 16 bit integrato su silicio, l'R800 poteva tenere testa, in molte operazioni, a un Motorola 68000 a 7 MHz.

I problemi maggiori, tuttavia, riguardarono il processore video. Per l'MSX3, ASCII e Yamaha avevano progettato il chip Yamaha V9978: un integrato con DAC a 15 bit (palette di 32.768 colori), due piani indipendenti per le modalità a pattern con scorrimento hardware multidirezionale, un motore hardware per il bit-block transfer e il disegno rapido di primitive, e una gestione di 128 sprite (fino a 16 per scanline a 15 colori più trasparenza). Ma Nishi pose una condizione inderogabile: la totale retrocompatibilità a livello di registri interni con i vecchi processori video V9938 e V9958.

Yamaha incontrò enormi difficoltà nel far coesistere la logica legacy a 8 bit con la nuova architettura a 16 bit sullo stesso die: i registri generavano conflitti di temporizzazione interni e i macchinari produssero tassi di scarto insostenibili, facendo slittare i tempi di consegna oltre la fine del 1990. Panasonic, vincolata agli impegni commerciali presi, fu costretta a un compromesso al ribasso: nell'ottobre 1990 presentò il Panasonic FS-A1ST

"MSX turbo R", accoppiando il formidabile processore R800 al vecchio VDP V9958 del 1988, che equipaggiava l'MSX2+, e limitando l'audio al PSG e alla sintesi FM MSX-MUSIC.

Quando nel 1992 Yamaha completò il nuovo processore grafico, ribattezzato V9990 E-VDP, la retrocompatibilità con i registri storici era stata eliminata. Senza la piena compatibilità all'indietro, il V9990 non poté essere ratificato come MSX3, finendo relegato a schede di espansione per un manipolo di appassionati e chiudendo la parabola storica dell'MSX.

Sarebbe bastato... che Nishi accettasse una modalità di commutazione architetturale duale, o si affidasse a tecnologie di emulazione, invece di pretendere la compatibilità a livello di singoli registri sullo stesso circuito. L'MSX3 avrebbe invaso il mercato nel 1991 con un VDP a 32.768 colori, audio FM OPL3 avanzato e una CPU equivalente a uno Z80 da 28 MHz, minacciando da vicino il mercato di best seller come l'Atari ST e contrastando il declino degli home computer orientali.

Le altre occasioni mancate del decennio

Queste vicende non furono isolate, ma fecero parte di una sequenza ininterrotta di deviazioni storiche che hanno scandito l'intero decennio:

Texas Instruments TI-99/8 (1983): Erede del TI-99/4, il primo home computer a 16 bit della storia (1979), il 99/8 non arrivò mai sul mercato: cancellato nell'ottobre 1983, prevedeva una CPU TMS9995 a 10,7 MHz, 64 KB di RAM e bus seriale Hex-Bus.

Sarebbe bastato... non farsi prendere dal panico per le perdite operative della guerra dei prezzi con il C64 e superare i conflitti interni tra la divisione consumer e quella dei sistemi professionali. Il TI-99/8 avrebbe sdoganato i 16 bit nel mercato consumer con due anni di anticipo su Amiga e Atari ST.

Sinclair QL, la maturità negata e il progetto Loki (1984-1986): Presentato nel gennaio 1984, il QL era una macchina sorprendente per l'epoca: sistema operativo QDOS multitasking, SuperBASIC strutturato e 128 KB di RAM a un prezzo popolare. Ma Sir Clive Sinclair lo lanciò in fretta, prima ancora di disporre di un prototipo funzionante: le prime macchine arrivarono mesi dopo, alcune con parte della ROM su una scheda esterna aggiunta in extremis, "castrate" dal bus a 8 bit del 68008, con chip ULA pieni di bug e gli inaffidabili Microdrive a nastro. Parallelamente, Martin Brennan e John Mathieson avevano concepito il Loki, un SuperSpectrum da 256 colori, accelerazione blitter e sonoro digitale tramite DAC a 8 bit. Dopo la cessione ad Amstrad della linea di computer Sinclair, nell'aprile 1986, il progetto rimase congelato; i due fondarono Flare Technology, e le loro idee migrarono senza successo verso altre realtà produttive, fino all'Atari Jaguar.



Sarebbe bastato... sei mesi: un tempo sufficiente per definire meglio il progetto, costruendolo attorno al Motorola 68000, sceso di prezzo proprio in quel periodo, con bus a 16 bit pieno, e più tempo per sviluppare ULA senza bug. Ne sarebbe nato un QL Plus monoblocco, con un drive floppy da 3,5 pollici integrato a destra della tastiera al posto dei Microdrive – o perfino da 2,8/3", per distinguersi dalla concorrenza – e l'ambiente WIMP grafico ICE a gestire il multitasking del QDOS. In un'altra linea temporale, sarebbe bastato che Alan Sugar, dopo aver rilevato la linea di computer Sinclair, portasse avanti il Loki: in questo scenario avrebbe ridefinito i limiti del computer entertainment europeo.

Atari ST (1985): Una macchina a suo modo semplice ed economica, spinta dal Motorola 68000 a 8 MHz con un bus a 16 bit pulito, porte MIDI integrate di serie che ne fecero lo standard della produzione musicale mondiale e l'ambiente grafico GEM (su disco nei primissimi esemplari, poi in ROM). Contrariamente a un diffuso pregiudizio, l'ST non era una macchina esclusivamente da ufficio: il chip video Shifter gestiva sia il colore per i videogiochi (320×200 a 16 colori e 640×200 a 4 colori su palette di 512), sia una straordinaria modalità monocromatica ad alta definizione (640×400 a 71 Hz). Bastava collegare il monitor dedicato SM124 perché il sistema commutasse automaticamente sull'alta risoluzione a scansione progressiva, regalando una nitidezza e un'assenza di sfarfallio per il Desktop Publishing che nessun concorrente poteva offrire a quel prezzo.

Sarebbe bastato... che Jack Tramiel reinvestisse una frazione dei margini iniziali nello sviluppo tempestivo di coprocessori custom dedicati che potessero combattere ad armi pari con quelli dell'Amiga, anziché centellinarli tra il blitter del Mega ST (1987) e il tardivo e interlocutorio STE del 1989. Avrebbe neutralizzato sul nascere il principale vantaggio architetturale made in Commodore, blindando il primato dell'Atari ST sia negli studi professionali, dove veniva preferito, sia nel gaming domestico.

Acorn Archimedes e l'occasione Olivetti (1987): Il primo home computer al mondo basato su architettura RISC a 32 bit. Il processore ARM2 a 8 MHz toccava 4 MIPS (contro il singolo MIPS scarso del 68000 e i 1-2 MIPS dell'Intel 80286), con un sistema operativo istantaneo residente in ROM (Arthur, sostituito nel 1989 dal RISC OS multitasking) e grafica a 256 colori senza sforzo. Ma la vera beffa risiedeva nella proprietà: dal 1985 la britannica Acorn era controllata a maggioranza dall'italiana Olivetti, che deteneva quasi l'80% delle quote azionarie. Invece di comprendere che nei laboratori di Cambridge era nata la piattaforma di calcolo del futuro, la dirigenza di Ivrea trattò Acorn come un fornitore secondario da cui pescare il BBC Master Compact per rimarchiarlo nel modesto Olivetti Prodest PC 128 S a 8 bit per il mercato casalingo italiano. Olivetti si ostinò a inseguire l'universo dei cloni IBM x86 (la linea M24, M28 e poi Prodest PC1 e PCS), lasciando che l'Archimedes rimanesse confinato nelle aule scolastiche d'Oltremanica.

Sarebbe bastato... che Olivetti riconoscesse il valore della tecnologia RISC ARM sviluppata dalla propria controllata, industrializzando l'Archimedes su scala globale e distribuendolo attraverso la propria capillare rete commerciale europea e le partnership americane al posto dei soliti cloni compatibili. L'architettura ARM e il RISC OS si sarebbero imposti sui desktop del mondo con dieci anni di anticipo sulla storia.

Gli anni Ottanta sono stati il vero crocevia della storia dell'informatica personale: un decennio in cui quasi tutti i paradigmi tecnologici moderni (il calcolo RISC a 32 bit, i coprocessori multimediali dedicati, la sintesi wavetable, le interfacce grafiche WIMP, il multitasking) sono stati concepiti, testati e in molti casi drammaticamente sprecati.

L'esplosione globale della piattaforma PC/Windows non ha premiato l'architettura migliore, ma la struttura più adatta all'omologazione industriale. Rileggere queste storie attraverso le tante occasioni mancate non serve a riscrivere il passato, ma a conservare la consapevolezza tecnica di quanto quel decennio sia stato vicino a regalarci un futuro tecnologico molto più vario, audace, affascinante e colorato che soltanto in tempi recenti, a più di trent'anni di distanza, abbiamo iniziato a sperimentare.



Il mio primo home computer: SV-318

di Jari Jalonen

Prima, uno sguardo breve e conciso al mercato dei computer a 8 bit in Finlandia negli anni '80. Il Commodore 64 e, più tardi, l'Amiga 500 conquistarono la Finlandia alla fine degli anni '80. Il Commodore divenne noto come il "Computer della Repubblica", poiché in Finlandia vendette più unità pro capite che in qualsiasi altro luogo al mondo. Prima che il Commodore raggiungesse questo status, la serie SV-3x8 di Spectravideo, basata su Z80 (SV-318 e SV-328), era una solida seconda scelta. Il 318 e il 328 differiscono molto nell'aspetto esteriore, ma internamente erano essenzialmente macchine identiche. Entrambi avevano capacità sonore identiche e i loro giochi erano compatibili tra loro. L'unica differenza era la quantità di memoria. L'SV-318 aveva 16 KB di RAM, mentre l'SV-328 ne aveva 64 KB. In Finlandia avevamo in vendita gli stessi computer a 8 bit del resto d'Europa, come Dragon 32/64, Salora Fellow/Manager, ZX Spectrum, Apple, Sharp e molti altri dispositivi. Mentre Commodore vendeva in quantità enormi, queste altre macchine rientravano chiaramente nella minoranza. Tra tutte queste, i dispositivi Spectravideo erano chiaramente la seconda scelta immediata accanto alla marcia trionfale di Commodore. O almeno lo erano all'inizio, ma alla fine degli anni '80 i dispositivi Spectravideo scomparvero gradualmente dal mercato, proprio mentre le vendite di Commodore crescevano e raggiungevano nuove vette.

L'SV-318 e io

Avevo già preso confidenza con l'SV-318 durante una visita a mia cugina. Lei e suo marito avevano un paio di figli più o meno della mia età. Eravamo in visita da loro, e i bambini mostrarono a me e a mio padre come usare il computer. Digitarono alcuni brevi programmi in BASIC e un paio di giochi. C'erano pochissimi giochi disponibili per il sistema, così volevano un nuovo computer perché "tutti gli altri ne avevano uno". Alla fine presero un Commodore 64, che presumibilmente tutti nella loro classe possedevano — anche se poi si scoprì che solo tre



Jari Jalonen con l'SV-318 all'eta' di 11 anni - Foto scattata dal padre Y.O. Jalonen

persone della classe avevano realmente un Commodore 64. La situazione fu una sconfitta per i loro genitori, ma una vittoria per me, dato che non avevano più bisogno del loro vecchio SV-318 e ci prestarono la macchina. Era il 1986 e a giugno avevo compiuto 11 anni. Il computer arrivò a casa nostra nel settembre 1986.

Il processore della macchina era uno Z80, usava l'SV-Basic e aveva solo 16 KB di memoria. I giochi venivano caricati da cartucce o cassette, come era pratica standard con quasi tutti i computer dei primi anni '80. Si potevano acquistare espansioni di memoria, un'unità a dischi floppy, una stampante e altre periferiche per il sistema. Tuttavia, le periferiche e le cartucce di gioco erano costose, quindi in pratica il registratore a cassette (datassette) e i giochi su cassetta erano il formato più comune da possedere, grazie al loro costo contenuto.

Per noi la macchina fungeva da console di gioco, ma fu anche la mia prima porta d'accesso all'apprendimento della programmazione. Il computer veniva fornito con un grande manuale finlandese SV 318-Basic formato A4, che conteneva una vasta quantità di nuove informazioni per me. Studiai la programmazione con diligenza. Comandi come SCREEN, CIRCLE, LINE, GOTO, GOSUB, INPUT e PRINT divennero così familiari che quando acquistai il mio SV-318 nel 2023 sapevo ancora programmare cerchi e linee dopo una pausa di 40 anni, con solo un po' di pratica. Oltre al manuale Basic, le riviste di computer in Finlandia — come ovunque nel mondo negli anni '80 — erano piene di listati di programmi per ogni computer immaginabile, il che significa che ce n'erano moltissimi anche per lo Spectravideo. Una rivista locale chiamata MikroBitti era una di queste fonti, da cui digitai molti programmi di giochi. MikroBitti era la rivista più diffusa rivolta agli appassionati di computer in Finlandia durante gli anni '80. Altre riviste includevano Printti e Tekniikan Maailma.



Gioco Kuumailmapalloilua

L'inizio, però, non fu facile. Avevo 11 anni e quelle erano le mie prime settimane con un computer, quindi né io né mio padre sapevamo praticamente nulla di questi dispositivi. Il primo codice che mio padre, Olavi (21 dicembre 1938 – 17 giugno 2005), digitò nel computer fu un gioco chiamato Minotauruksen labyrintti (Il labirinto del Minotauro). Mio padre digitò l'intero gioco nel computer e, prima che avesse la possibilità di salvarlo, io digitai RUN e il computer si bloccò. C'era un bug nel codice, ma non avevamo capito che bisognava salvare il gioco su cassetta almeno una volta. Ricordo mio padre che digitò di nuovo lo stesso gioco una seconda volta, ma se ben ricordo il connettore di alimentazione proprietario a 4 pin era allentato, la connessione cadde, e anche questo tentativo svanì nel cestino dei bit. Non ricordo se mio padre tentò mai una terza volta. Ricordo però che chiedemmo consiglio ai proprietari del computer. Si scoprì che il programma poteva essere salvato in qualsiasi momento. Avevamo supposto che il programma dovesse essere completamente finito prima di salvarlo su cassetta. Imparai che si poteva salvare un programma non finito su nastro dopo ogni singola riga, se se ne aveva voglia, permettendoti di riprendere da dove avevi lasciato se un problema cancellava la memoria del computer. Non ricordo più quante volte provammo o se semplicemente rinunciammo a quel gioco. Invece ricordo vividamente il primo gioco che fu un successo. Si chiamava Kuumailmapalloilua (Volo in mongolfiera).

Volo in mongolfiera e la proposta di Albert

Ricordo quanto fu bello quando finalmente digitai il programma da MikroBitti e lo trasferii al computer funzionante. Non ricordo se funzionò subito o se c'erano errori, ma la cosa più importante era che alla fine riuscì. Volo in mongolfiera è una tappa importante anche perché fu grazie a questo gioco che iniziai a imparare il Basic a livello pratico. Ricordo di aver cambiato un numero casuale. Ero un ragazzo e non sapevo ancora leggere il codice Basic; sapevo solo copiarlo dalla rivista. A causa di una piccola modifica, la dimensione della mongolfiera divenne enorme. Alla fine imparai a modificare il codice del gioco per renderlo più facile da giocare.



Hello RetroMagazine World! in Basic

Volo in mongolfiera è un gioco interessante. Il suo concetto centrale non implica velocità o spari, né un tempo di reazione rapido è di alcuna utilità. Lo schermo è diviso orizzontalmente in tre sezioni: a sinistra c'è la mongolfiera, al centro una montagna con una bombola del gas di riserva, e a destra il traguardo. Verticalmente, c'è un'alta montagna ben sagomata al centro e delle frecce sul lato sinistro. Alcune frecce puntano a destra e altre a sinistra. L'idea del gioco è intelligente: premi la barra spaziatrice per far salire la mongolfiera. Quando la mongolfiera raggiunge una freccia, si sposta a sinistra o a destra a seconda della direzione in cui punta la freccia. L'obiettivo principale è andare a prendere la bombola del gas di riserva e volare fino al bordo destro, dopodiché parte una fanfara di vittoria. All'inizio il gioco è difficile, ma una volta capiti i trucchi diventa facile. Il gioco entra interamente nella memoria dell'SV-318, quindi è un gioco molto piccolo con un solo livello. Una volta imparato a superarlo, puoi completarlo 11 volte di fila senza alcuna difficoltà aggiuntiva.

Lo stesso numero della rivista presentava anche un altro gioco chiamato Albertin Kosiomatka (Albert in cammino per fare la proposta), che fu il secondo gioco che riuscii mai a finire di digitare. L'obiettivo è percorrere un sentiero tortuoso pieno di pericoli dal lato sinistro dello schermo fino a destra. Il sentiero è stretto e lungo il percorso ci sono molti pericoli diversi. Il gioco è divertente e interessante, ma estremamente difficile. A testimonianza della sua difficoltà, non riuscii mai a superare questo gioco.

C'erano innumerevoli giochi che creai dai listati dei programmi. Uno dei più importanti era Qbertin junior, che, com'era prevedibile, era un clone quasi identico di Qbert*, anche se secondo me più facile dell'originale. La cosa fantastica del gioco era che includeva un editor di livelli, che permetteva di creare nuovi livelli e salvarli su cassetta. Avevo una quantità enorme di livelli miei, ma purtroppo sono tutti andati perduti nel tempo. I listati dei programmi e le cassette di raccolta vendute nei negozi erano una parte enorme del fascino del computer, semplicemente perché la macchina veniva fornita con pochi giochi originali ufficiali. Pertanto, i giochi offerti dai listati dei programmi erano un passatempo estremamente prezioso.

Giochi originali

Il computer veniva fornito con quattro cassette di giochi, una delle quali era una cassetta bundle Spectravideo contenente vari programmi di esempio in Basic. Oltre a questa, c'erano le cassette di giochi Spectron, Nomis e Armoured Assault. Tra questi, Spectron è stato il gioco più importante per me. Spectron era un clone di Space Invaders. In esso, lo Spectron si muove in basso e spara a creature o navi che arrivano dall'alto. Anche se Spectron era un clone, si distingueva molto dalla versione originale perché era molto più avanzato. Oltre alle navi in stile Invaders, c'erano diverse varianti di mezzi nemici. Il gioco inizia in modo tradizionale, ma la velocità aumenta livello dopo livello e, oltre ai nemici precedenti, appare una nuova versione che può piombare in basso per molestare il giocatore nella parte inferiore dello schermo. Queste creature possono scivolare sotto il giocatore e generare lì una piccola prole. Una volta che abbastanza di questi "piccoli" si accumulano sul bordo inferiore, è game over,



Gioco Albert in kosiomatka

anche se al giocatore restano ancora delle vite.

Nomis era più semplice di Spectron, ma impegnativo a modo suo. L'idea del gioco era mettere alla prova la memoria umana. Inizialmente offre un paio di numeri; se li ricordi correttamente, il gioco continua. Poi ne arrivano tre, e se li memorizzi ne appaiono quattro. I numeri continuano ad arrivare finché il giocatore non commette abbastanza errori, punto in cui il gioco finisce. Il mio record personale di un paio di anni fa era di 12 numeri. Non ricordo che tipo di record stabilii negli anni '80.

Armoured Assault era un gioco di carri armati per due giocatori. Entrambi i giocatori controllano un carro armato. Ci sono più livelli, e chi riesce a distruggere per primo tre vite dell'avversario vince la partita. Tra questi giochi originali, Spectron era quello che giocavo di più all'epoca. Ho una quantità immensa di ricordi affettuosi di quelle innumerevoli sessioni di gioco. L'SV-318 rimase in uso solo per un anno, finché non presi l'SV-728, ma questa è un'altra storia.

Ritorno agli anni '80

Quando mi allontanai dal lavoro nel maggio 2023, avevo molto tempo libero. Mi ero interessato al retro computing alla fine degli anni '90 e nei primi anni 2000, ma l'hobby era stato completamente abbandonato per un motivo o per l'altro. Nel 2023 avevo molto tempo libero e un forte bisogno di distogliere la mente da un rapporto di lavoro fallimentare. Trovai in soffitta ogni sorta di cose che avevo comprato intorno al cambio del millennio. Trovai il Fellow menzionato nei miei testi precedenti, un Commodore 64G, un SV-328 e vecchi computer tascabili Sharp. Inoltre c'erano due computer VIC-20, un paio di modelli più vecchi di Commodore 64 e molto altro. Alcuni dispositivi funzionavano, altri no.

Spectron era un gioco che mi mancava da alcuni anni. Avevo visto un YouTuber giocare a Spectron su un SV-328. Ero assolutamente certo che non avrei mai più potuto giocare a questi giochi che avevo giocato sull'SV-318. Quando iniziai a navigare nei mercati online locali e internazionali, come eBay, mi resi conto che c'erano molti giochi Spectron in vendita, e il prezzo non era nemmeno

alto. Comprai una versione CIB (Complete in Box) di Spectron per qualche decina di euro.

Ricordo chiaramente quando caricai il gioco sull'SV-328 per la prima volta. Non iniziai il gioco; mi limitai a immergermi nell'atmosfera, sapendo che ora avevo Spectron caricato e che il menu iniziale era visibile sul televisore collegato al computer. La sensazione era indescrivibile. E quel suono — un suono che pensavo non avrei mai più sentito da hardware reale con General Instrument AY-3-8910 senza YouTube — ed eccolo lì! Proprio davanti ai miei occhi e accanto alle mie orecchie c'era qualcosa che non vedevo né sentivo da anni. Sfortunatamente, quando provai a collegare il joystick al computer, la corrente si interruppe. Pensai che il cavo fosse rotto, ma avevo dimenticato lo stesso identico problema che si era verificato molto tempo prima con Minotaur's Labyrinth: la connessione tra l'SV e il connettore di alimentazione proprietario a 4 pin è notoriamente fragile su queste macchine. Anche il minimo urto, come inserire un joystick, interrompeva la connessione e il dispositivo si riavviava. Fortunatamente, ricaricai il gioco e ci giocai per la prima volta dal 1987.

L'SV-318 originale rimase con noi per quasi esattamente un anno. Arrivò nell'autunno del 1986 e fu sostituito dall'SV-728 nell'autunno del 1987. La mia prima esperienza di gioco nel 2023 fu pessima. Ricordavo di essere davvero bravo in quel gioco, ma nel 2023 non riuscii nemmeno a superare il primo livello. Mi arrabbiai moltissimo e decisi che non avrei mai più giocato con quella macchina. Fortunatamente, non sono uno che porta rancore, e qualche giorno dopo decisi di giocare di nuovo. Il gioco andò molto meglio, riuscii ad arrivare fino al livello 8 — un tragitto incredibilmente lungo. Tra il 2023 e il 2025 giocai intensamente a vecchi giochi a cui giocavo molto negli anni '80. Con quasi tutti, la prima esperienza di gioco dopo anni fu un disastro, ma al secondo o terzo tentativo giocare risultava molto meglio e i riflessi si risvegliavano. Tuttavia, i miei riflessi non sono più allo stesso livello di quando avevo 11-15 anni.

Quanto all'SV-318: non era nella nostra soffitta, perché fu restituito ai suoi legittimi proprietari nel 1987. Tuttavia, nel luglio 2023, un SV-318 fu messo in vendita su Huuto.net, un sito d'asta locale. Il pacchetto includeva la scatola originale, l'alimentatore (che sembra essere più raro delle stesse macchine SV-3x8 o SV-728) e il manuale originale di programmazione Basic — lo stesso identico che avevo a disposizione nel 1986. Il prezzo di partenza era di 70 euro, ma la guerra delle offerte fu accanita, e il prezzo alla fine salì a 215 euro. Vinsi l'asta e mi riportai a casa questa magnifica macchina.

Anche se giocare a Spectron sull'SV-328 fu una grande esperienza, non fu altrettanto magico come sull'SV-318. La nostalgia era così forte che l'SV-318 fu un'esperienza più importante, anche se le macchine sono internamente identiche. Queste sono quelle cose mistiche ed emotive che la ragione non può spiegare.

Nel corso del 2023 comprai occasionalmente altri giochi per SV-3x8. Più tardi acquistai giochi come Armoured Assault e Nomis. Circolano raramente in Finlandia, ma ogni tanto salta fuori un nuovo gioco. Da allora ne ho comprati molti altri, come Sasa, Minigolf, O'Mac Farmer e molti altri ancora. Tra queste aggiunte più recenti, Minigolf è stato il mio preferito in assoluto. La grafica è molto semplice, ma il gioco in sé è incredibilmente impegnativo. Devi cercare di mandare in buca una pallina minuscola di un pixel in una buca grande solo un paio di pixel, attraverso percorsi complessi. A volte perdo la pazienza e il joystick finisce contro il muro quando proprio non vuole riuscire. Ma di sfide non ce ne sono certo mancate.

Esempio di codice:

```
10 SCREEN 2,1
20 CIRCLE(128,96),90,8
30 PAINT(128,96),8
40 GOTO 40
```

(Cerchio colorato al centro dello schermo)

SVI 318/328	
Operating system	Microsoft Extended SV BASIC
CPU	Zilog Z80 A
Speed	3,6 Mhz
RAM	SVI 318 16 Kb and SVI 328 64 Kb (up to 256 kb)
VRAM	16 Kb
ROM	32 Kb
Graphics	TMS9918A SCREEN 0: Text mode 40x24 characters, two colors (background+foregrund). SCREEN 1: High resolution graphics mode: 256x192, 16 colors. Sprites are available. SCREEN 2: Low resolution graphics mode: 64x48, 16 colors. Sprites are available. The screen is addressed and formed using full 256x192 resolution.
Text Modes	40 chars x 24 lines (optional 80 x 24)
Graphic modes	256 x 192 dots
Colors	16
Sound Chip	AY-3-8910
Sound	Three channels, 8 octaves

Costruire un'avventura testuale con un narratore AI

Dalle istruzioni di un GPT a un'applicazione per macOS e Windows: l'esperienza di The Curse of the Whittaker Family

di Marco Figaro

Una porta socchiusa, una torcia quasi scarica e un comando da scrivere. Le avventure testuali costruiscono il loro rapporto con il giocatore attraverso questo scambio: il programma descrive una situazione, il giocatore decide come affrontarla e il mondo risponde.

Un narratore basato sull'intelligenza artificiale introduce una possibilità interessante: interpretare richieste formulate liberamente e generare descrizioni durante la partita. Per chi realizza l'avventura, questo significa progettare insieme una storia, delle regole e il comportamento di una voce narrante.

Ho esplorato questo percorso con The Curse of the Whittaker Family, un'avventura horror sviluppata in due forme: un GPT personalizzato all'interno di ChatGPT e un'applicazione autonoma per macOS e Windows. Le due realizzazioni affrontano lo stesso problema con strumenti diversi: dare spazio all'iniziativa del giocatore mantenendo riconoscibile il mondo che sta esplorando.

Prima del modello, il mondo

Il primo passo è definire ciò che rende l'avventura una storia precisa.

Nel mio progetto, Mark riceve una lettera del padre scomparso che lo conduce a Redgrave Manor. La villa custodisce un mistero familiare; l'esplorazione mette alla prova anche la lucidità del protagonista. Fotografie, lettere e oggetti personali forniscono punti di partenza per l'indagine.

Questi elementi diventano materiale di riferimento per il narratore. Il documento iniziale contiene sezioni dedicate alla missione narrativa, al protagonista, alle regole, alla situazione iniziale e al ciclo di gioco. Comprende inoltre comandi, inventario e storia della famiglia.

Organizzare il materiale serve a rendere esplicite le decisioni dell'autore. Chi è il protagonista? Che cosa sa all'inizio? Quali oggetti possiede? Quali eventi possono interrompere l'esplorazione? Dove deve fermarsi il narratore per aspettare il giocatore?

Una generica richiesta di raccontare una storia horror lascia tutte queste scelte al modello. Preparare un'avventura significa decidere quali risposte devono invece rimanere stabili.

Il GPT come laboratorio narrativo

Nella versione GPT, il documento è caricato come materiale di conoscenza e accompagnato da istruzioni operative.

Alcune definiscono l'identità del gioco: l'ambientazione è Redgrave Manor, le risposte devono riguardare l'avventura e ogni turno deve terminare con "Cosa fai?". Altre riguardano le meccaniche: stato mentale, inventario limitato dal peso, mappa progressiva, oggetti combinabili ed enigmi.

Il narratore riceve anche indicazioni su elementi specifici, come il vecchio pianoforte a coda coperto da un lenzuolo nel salone. È un modo per conservare particolari riconoscibili all'interno di una narrazione che può cambiare.

Una scelta centrale riguarda il controllo del protagonista. Il giocatore decide le azioni di Mark; il narratore descrive l'ambiente e le conseguenze. Se dietro una porta si sente un rumore, l'AI deve lasciare spazio alla decisione di ascoltare, entrare o allontanarsi.

Il GPT permette di sperimentare questa impostazione dentro un'interfaccia conversazionale già disponibile. Il lavoro dell'autore si concentra quindi sulla struttura dei materiali, sulle istruzioni e sulla valutazione delle risposte.

Le regole scritte non sono controlli automatici

Una difficoltà emerge quando il racconto deve ricordare fatti precisi per molti turni.

Le istruzioni possono chiedere di conservare l'inventario e aggiornare la mappa, ma il rispetto di queste richieste dipende dal comportamento del modello. Una descrizione scorrevole può contenere una contraddizione: un oggetto ricompare, una risorsa consumata torna disponibile oppure il protagonista raggiunge un luogo senza un passaggio plausibile.

Anche il salvataggio riflette questa impostazione. Nel GPT, "salva partita" richiede un riepilogo con azioni compiute, inventario, risorse, stato mentale, mappa e



The Curse of the Whittaker Family è un'opera di Marco Figaro, registrata su Safe Creative
Registrazione: https://www.safecreative.org/certificate/2608016608845-45R9PS

abilità. Il giocatore può copiarlo e incollarlo per riprendere l'avventura. È una ricostruzione testuale dello stato, la cui utilità dipende dalla completezza e dalla corretta interpretazione del riepilogo.

Questa fase insegna a distinguere due qualità: quanto una risposta coinvolge e quanto rispetta ciò che è già accaduto. Entrambe sono necessarie per sostenere una partita.

Portare l'avventura in un'app per macOS e Windows

Le versioni per macOS e Windows condividono una struttura software basata su Tauri, con un'interfaccia realizzata in HTML, CSS e JavaScript e una componente in Rust che avvia il servizio del modello. Gli eseguibili e le librerie sono specifici per ciascun sistema operativo.

Nei pacchetti sono inclusi un modello quantizzato in formato GGUF e llama-server, insieme ai componenti necessari per la rispettiva piattaforma. La narrazione viene eseguita sul computer, senza richiedere una conversazione su ChatGPT. All'avvio, il modello deve essere caricato in memoria prima di poter rispondere ai comandi del giocatore.

L'interfaccia affianca al racconto la mappa, l'inventario, il peso trasportato e lo stato mentale. Il salvataggio locale conserva i dati della partita, mentre il modello riceve le informazioni necessarie per interpretare il comando successivo.

Il passaggio dall'idea all'applicazione richiede quindi di trasformare una parte delle istruzioni narrative in dati e controlli gestiti dal programma.

Come viene elaborato un comando

Nell'app ogni richiesta inviata al modello comprende il luogo attuale, le uscite raggiungibili, i fatti visibili, l'inventario e gli oggetti che si possono raccogliere nella stanza.

Al modello viene chiesta una risposta strutturata in JSON, un formato che il programma può leggere. La risposta contiene sia la narrazione sia informazioni come intenzione, destinazione proposta ed eventuali modifiche agli oggetti. Un esempio concreto, previsto nelle istruzioni del codice, è "salgo al piano superiore". Dal salone questa frase può indicare il corridoio del primo piano, passando attraverso la scalinata. Il programma controlla che la destinazione appartenga alle uscite consentite e che il comando esprima un movimento compatibile, prima di aggiornare la posizione. Per raccogliere un oggetto vengono verificati la sua presenza nell'elenco previsto, la stanza in cui si trova e il peso disponibile nell'inventario.

La sequenza diventa così: il giocatore esprime un'intenzione, il modello propone un'interpretazione, il programma applica i controlli disponibili e l'interfaccia mostra il risultato.



Il protagonista del gioco

Quanto controllo serve?

Questa architettura introduce verifiche concrete, ma richiede attenzione anche ai suoi limiti.

L'applicazione controlla diversi aggiornamenti dello stato; alcune conseguenze, come le variazioni dello stato mentale e il conteggio degli enigmi, dipendono ancora dalle proposte del modello entro limiti numerici.

Per sviluppare un sistema di questo tipo, occorre verificare anche l'accordo tra racconto e dati. Se il programma rifiuta uno spostamento, il testo non dovrebbe descrivere l'arrivo nella stanza richiesta. Se una porta necessita di una chiave, quella condizione deve essere rappresentata nelle regole operative.

Per gli enigmi, la stessa logica suggerisce di definire esplicitamente condizioni, oggetti necessari e conseguenze della soluzione. Il narratore può descrivere il risultato e dosare gli indizi, mentre il programma verifica che la soluzione sia stata realmente raggiunta.

Scrivere, provare, correggere

Realizzare un'avventura con narrazione AI richiede prove che vadano oltre la lettura di una bella scena.

È utile chiedere di usare un oggetto assente, tentare uno spostamento impossibile, ripetere un'azione o formulare un comando ambiguo. Questi casi mostrano quanto il sistema rispetti i confini del mondo. Sul piano narrativo, bisogna invece osservare ritmo, chiarezza e spazio lasciato al giocatore.

Nel progetto, GPT e applicazione locale sono due modi di affrontare questo lavoro. Il primo permette di sperimentare direttamente istruzioni e conversazione; la seconda consente di rendere espliciti dati e controlli, aggiungendo però la complessità di un software da distribuire e mantenere. Il compito dell'autore attraversa entrambi: costruire luoghi, stabilire conseguenze, scegliere che cosa può cambiare e proteggere ciò che rende la storia coerente. Una buona risposta deve lasciare al giocatore qualcosa su cui ragionare. Poi il narratore si ferma, e il cursore torna ad aspettare.

Cosa fai?

Sito del progetto: https://thecurseofthewhittakerfamily.kesug.com/

GPT dell'avventura: https://chatgpt.com/g/g-Yugl1z0t0-the-curse-of-the-whittaker-family

GentleOS - Un OS moderno per l'era del PC a 16 e 32 bit

di Alex Fridman

Quando si parla di recupero di PC storici, la risposta abituale è una distribuzione Linux alleggerita. GentleOS percorre invece una strada più radicale. Parliamo di un sistema operativo open source progettato espressamente per macchine x86 datate, con una GUI nativa, requisiti minimi estremamente contenuti e una variante destinata persino ai sistemi basati su Intel 80186.

L'obiettivo non è trasformare un 386 in un moderno desktop per il web, ma fornire una piattaforma autonoma, graficamente utilizzabile e tecnicamente coerente con le limitazioni dell'hardware anni Ottanta e Novanta.

Oltre la "distro leggera"

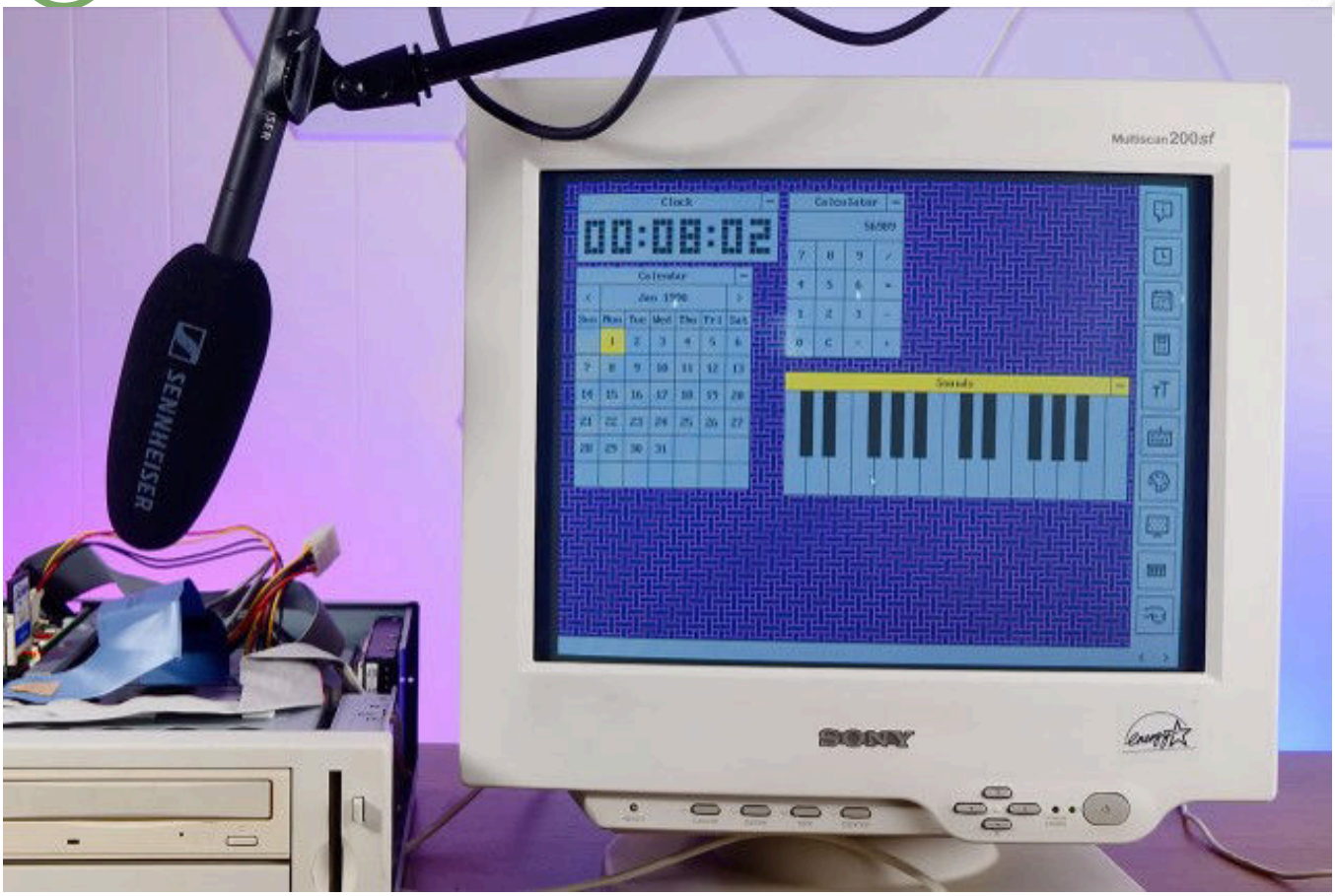
GentleOS non va interpretato come l'ennesima distro Linux ridotta all'osso. Il progetto è rivolto ai PC classici a 32 bit e nasce per consentire sperimentazione diretta su hardware reale, eseguendo applicazioni su macchine ormai fuori dal perimetro dei sistemi operativi contemporanei. La differenza è sostanziale. Un sistema Linux attuale, anche se minimale, continua a portarsi dietro un ecosistema, driver, librerie e aspettative d'uso concepiti per piattaforme molto più recenti. GentleOS parte invece dal presupposto opposto ovvero che l'hardware limitato non è un vincolo da nascondere, ma il riferimento progettuale attorno a cui costruire il sistema.

In quest'ottica, il valore del progetto è soprattutto sperimentale. Può interessare il collezionista che desidera usare davvero un vecchio portatile, ma anche chi vuole studiare le implicazioni pratiche della progettazione software sotto vincoli di memoria e potenza di calcolo oggi quasi scomparsi.

Dal 386 all'80186

La versione standard di GentleOS richiede come base una CPU Intel i386, 4 MB di RAM e un sottosistema video VGA capace di operare a 640 × 480 pixel. Sono specifiche che collocano il sistema alla portata di una larga fascia di PC





MS-DOS e compatibili dei primi anni Novanta.

L'elemento più curioso è però GentleOS/16, edizione destinata ad architetture ancora precedenti. Il riferimento all'Intel 80186 — processore introdotto nel 1982 — sposta il progetto dal semplice riuso di PC obsoleti a un territorio propriamente retrocomputing.

Va però fatta una distinzione importante. La compatibilità con CPU storiche non equivale automaticamente alla piena compatibilità con ogni PC dell'epoca. In ambiente x86, BIOS, controller video, storage, configurazioni di memoria e periferiche possono fare la differenza tra un boot riuscito e una lunga sessione di debugging. È proprio questo, del resto, uno degli aspetti più interessanti per chi ama lavorare su hardware reale.

Kernel monolitico e semplicità

GentleOS adotta un'architettura monolitica, nella quale i componenti principali del sistema operativo lavorano nello stesso spazio di indirizzamento. La scelta privilegia prestazioni e semplicità strutturale, riducendo alcuni livelli di astrazione e le complessità tipiche di architetture più modulari.

Su un sistema con pochi megabyte di RAM, l'economia delle risorse è fondamentale. Ogni servizio residente, ogni livello software e ogni buffer incidono in modo molto più evidente rispetto a una macchina contemporanea con gigabyte di memoria disponibili. Un kernel monolitico può quindi essere una soluzione sensata quando il progetto punta a mantenere basso l'overhead complessivo. Naturalmente, il rovescio della medaglia è una maggiore integrazione dei componenti che può rendere più delicata la manutenzione e meno isolati gli eventuali errori. Gli autori riconoscono infatti che il sistema è ancora in sviluppo, con bug e aspetti da rifinire nelle versioni successive.

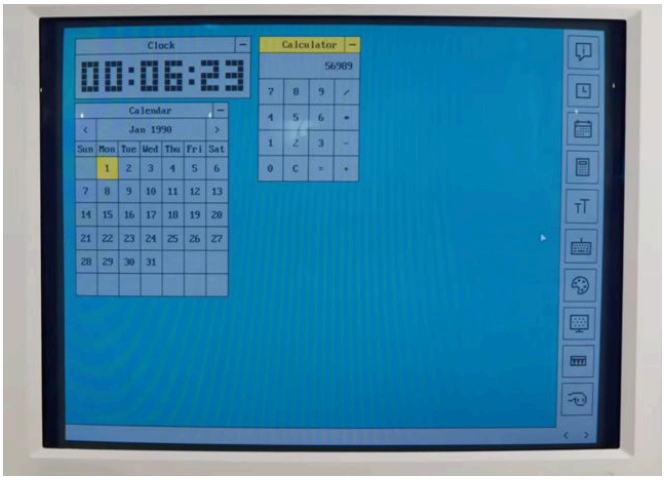
Un desktop, non solo un boot loader

Il pregio di GentleOS è che non si limita a dimostrare l'avvio di una macchina d'epoca. Il sistema propone una GUI essenziale e un corredo software sufficiente a trasformare il PC in una postazione utilizzabile, almeno

Componente	Requisito dichiarato	Significato pratico
Processore	Intel i386	Intel i386
Intel i386	4 MB di RAM	Vincolo molto inferiore agli standard dei sistemi moderni
Video	VGA 640X480	Pensata per il PC Classico



entro i limiti del suo progetto.
Tra le applicazioni incluse compaiono diverse utility di base, come orologio, calendario e calcolatrice.
Piccoli strumenti grafici e programmi di disegno.
Giochi classici, tra cui Solitario, Snake e Memoria.
Passatempi e software leggero, coerenti con l'impostazione minimalista del sistema.



L'elenco può sembrare modesto se valutato con criteri contemporanei, ma non è quello il punto. Su una piattaforma con i386, 4 MB di RAM e VGA, il fatto stesso di disporre di un ambiente grafico integrato e coerente rappresenta una dichiarazione d'intenti ovvero riportare l'utente davanti a un computer che risponde direttamente, senza strati superflui e senza pretendere risorse che non possiede.

Un laboratorio su hardware reale

GentleOS può essere letto in due modi. Da una parte è un progetto di recupero: una possibilità per sottrarre vecchi PC al magazzino o alla discarica elettronica. Dall'altra, e forse più interessantemente, è un piccolo laboratorio per capire come cambia il software quando CPU, RAM e grafica non sono risorse virtualmente illimitate.

Per il retro-programmatore, macchine di questo tipo sono una palestra concreta. Risoluzione fissa, memoria ridotta, architettura x86 storica e un OS che mira alla semplicità obbligano a ragionare in termini di footprint, layout dei dati, accesso alle risorse video e robustezza del codice.
GentleOS non sostituirà un sistema moderno né promette di rendere un 386 adatto alle esigenze della rete contemporanea. Ma non ne ha bisogno perchP il suo interesse sta nel dimostrare che un computer di trent'anni fa può ancora essere una macchina completa, non soltanto un reperto da esporre.

Link Utili

Link GentleOS32
- https://github.com/luke8086/gentleos32

Link GentleOS16
- https://github.com/luke8086/gentleos

Website
- https://luke8086.dev/gentleos16

L'Arte di forzare l'Hardware: il VSP Bug sul Commodore 64

di Felice Nardella aka Kimono

1. Introduzione

Fra le tante tecniche che hanno contribuito a spingere il Commodore 64 (abbreviato C64) ben oltre i limiti immaginati dai suoi progettisti, figura il Variable Screen Positioning (VSP), una tecnica che consente di ottenere scrolling estremamente fluidi, evitando il costoso spostamento della memoria video da parte della CPU.

Tuttavia, la sua attuazione nasconde un'insidia: un comportamento instabile a livello di bus di memoria, noto come VSP Bug, che è capace di corrompere la RAM e bloccare la macchina.

Il VSP non è semplicemente un trucco utilizzato da demo e videogiochi per ottenere scrolling fluidi e veloci: dietro questa tecnica si nasconde, in realtà, uno dei comportamenti più misteriosi e controversi del VIC-II, mai documentato dalla Commodore e rimasto inspiegabile per oltre vent'anni. Nato come un comportamento completamente accidentale dell'hardware, è stato poi trasformato dalla creatività dei programmatori, in uno strumento capace di realizzare effetti grafici che sembravano impossibili su una macchina del 1982!

In questo articolo cercherò di spiegare che cos'è il VSP, come funziona e perché su alcune macchine corrompe la memoria e manda in crash la CPU.

2. Un po' di storia del VSP

Pare che tutto sia nato per caso negli anni '80: secondo la tradizione, mentre qualcuno stava sperimentando con l'FLD (Flexible Line Distance) si è imbattuto nell'effetto, senza volerlo. Manipolando con precisione millimetrica, al singolo ciclo di clock, il registro $D011, mentre il VIC-II sta per generare una determinata riga, si riesce a ingannare il chip grafico costringendolo a iniziare il fetch della memoria video con un ritardo arbitrario, e quindi a spostare il Video Counter [6].

La prima apparizione documentata del trucco la si trova nella demo "VSP&IK+" (1987) dei Meanteam [5][9], che diede alla tecnica il suo nome: "Variable Screen Positioning" (anche se il nome tecnicamente più corretto sarebbe "DMA delay"). Mentre nel software commerciale, proprio per i problemi di stabilità di cui parleremo, restò merce rara, da lì in poi il VSP divenne un classico delle demo. I titoli che ebbero il coraggio di usarlo si contano sulla punta delle dita: "Mayhem in Monsterland" (1993), dove il VSP dà il meglio di sé alle velocità di scorrimento più alte e "Creatures" (1990), ma anche "Another World" (1991) e la recente conversione homebrew di "Super Mario Bros" (2019) [5][6].

3. Architettura dei Bus e Display Window: Come lavora il VIC-II

Per comprendere come sia possibile "ingannare" il C64, occorre innanzitutto analizzare il modo in cui il chip grafico VIC-II (MOS 6569/8565) e la CPU (MOS 6510) condividono le risorse di sistema.

3.1. La condivisione del System Data Bus

In ogni singolo ciclo di clock della macchina, sia il VIC-II sia la CPU accedono alla memoria prendendo a turno il controllo del System Data Bus. Ciascun ciclo di clock è suddiviso rigorosamente in due fasi [1]:

• 1a Fase (fase bassa del clock): il VIC-II effettua i suoi accessi alla memoria in sola lettura.

• 2a Fase (fase alta del clock): la CPU effettua i propri accessi in lettura o scrittura.

Durante ogni ciclo di clock, il VIC-II genera 8 pixel alla volta sul segnale video.

3.2. Righe di schermo, Text Mode e Bitmap Mode

L'area visibile dello schermo (la Display Window) è strutturata in precise unità di misura:

• 8 Raster line (o scan line) formano una Riga di Schermo.

• Modo Testo (Character Mode): di default la Display Window è formata da 25 Righe x 40 Colonne. Ogni carattere visualizzato è costituito da un blocco matrice di 8x8 pixel.

• Modo Grafico (Bitmap Mode): le 25 righe per 40 colonne vengono gestite indirizzando i singoli pixel per un totale di 320x200 pixel (corrispondenti a 64.000 bit di memoria, ossia 8 KB). Anche in Bitmap Mode, la struttura sottostante rimane organizzata in blocchi di 8x8 pixel.

3.3. Il concetto fondamentale: Standard Line vs. Bad Line

In una riga di raster standard (composta in totale da 63 cicli di clock in ambiente PAL), il VIC-II e la CPU convivono pacificamente. Tuttavia, la prima raster line di ogni riga di schermo si comporta diversamente.

3.3.1. La Bad Line

La prima raster line di ciascuna riga di schermo "dura" per la CPU soltanto 23 cicli di tempo macchina (25 nello standard NTSC). I restanti 40 cicli vengono "sequestrati" dal VIC-II tramite il segnale BA (Bus Available), fermando la CPU per effettuare accessi in memoria aggiuntivi, oltre a quelli ordinari.

Nel Text Mode il VIC-II accede alla Video Matrix (Screen RAM) per leggere i 40 codici carattere della riga attuale di schermo, e contestualmente accede alla Color RAM per leggere i codici colore dei caratteri.

Le raster line in cui il VIC-II blocca la CPU per 40 cicli prendono il nome di Bad Line e si verificano solo all'interno della Display Window.

3.3.2. La Bad Line Condition

Affinché il VIC-II generi una Bad Line, devono verificarsi le tre seguenti condizioni:

a. i 3 bit meno significativi (bit 0-2) del registro raster $D012 devono coincidere esattamente con i 3 bit più bassi (bit 0-2) del registro $D011 (YSCROLL);

b. il bit DEN (Display Enable) del registro $D011 è impostato a 1 (bit 4 di $D011);

c. il raster è compreso tra le linee $30 e $F7.

Il valore di default nei bit 0-2 di $D011 è 3 (%011). Di conseguenza, in condizioni la prima Bad Line del quadro è quindi la raster line 51 ($33, %00110011), che corrisponde esattamente alla prima linea visibile della Display Window, mentre l'ultima è la 243 ($F3, %11110011), seguita dalle sue 7 raster line fino alla 250 ($FA), l'ultima visibile. Ogni gruppo di 8 raster line consecutive (la prima delle quali è una bad line) forma, insomma, una delle 25 righe della Display Window [1].

3.3.3. Display State, Idle State e tipi di accesso

Gli accessi con cui il chip legge, durante una Bad Line, i 40 puntatori ai caratteri della riga corrente (e la relativa informazione colore) si chiamano c-access; quelli con cui, su ogni raster line della Display Window, legge i pixel veri e propri dal generatore di caratteri o dalla bitmap si chiamano g-access [2].

In altre parole: la Bad Line è la linea dei c-access, mentre i g-access avvengono su tutte le standard raster line e anche nelle Bad Line. A questi si aggiungono, per completezza, gli accessi di refresh della DRAM e, se sono presenti sprite, quelli dedicati alla lettura dei loro puntatori e dei loro pixel.

Il VIC-II, inoltre, può trovarsi in ogni istante in uno di due stati.

Nel Display State (attivo lungo le linee della Display Window) il chip esegue regolarmente i suoi accessi alla memoria video.

Nell'Idle State, tipico del bordo superiore e inferiore e del vertical blanking, il chip non accede né alla matrice video, né al generatore di caratteri: si limita a leggere sempre la stessa locazione, l'ultima del banco (bank) da 16 KB corrente (offset $3FFF), il cosiddetto ghost byte [2].

L'indirizzo $3FFF va inteso come offset relativo al banco: per ricavare l'indirizzo assoluto del ghost byte bisogna sommare, banco dopo banco, i 16 KB (16.384 byte, ossia $4000) che costituiscono la dimensione di ogni VIC bank. Di conseguenza, il ghost byte risiede in:

a. $3FFF nel Bank 0,

b. $7FFF nel Bank 1,

c. $BFFF nel Bank 2,

d. $FFFF nel Bank 3.

Anche quando è nello stato di inattività, il VIC-II continua a eseguire accessi g su ogni linea raster appartenente ai bordi verticali, leggendo sempre il ghost byte.

Va precisato che nell'Idle State, in condizioni normali, i g-access non si fermano: il controller video continua ad effettuarli su tutte le raster line dei bordi verticali, sempre e soltanto verso la locazione del ghost byte.

A prescindere dall'effettivo bit pattern del ghost byte, il sequencer video lo interpreterà come se tutti i bit fossero azzerati. Di conseguenza, nei bordi verticali non verrà acceso alcun pixel, che assumerà quindi di default il colore del bordo ($D020).

Con la tecnica dell'apertura dei bordi verticali, invece, i bit a 1 accendono i pixel corrispondenti, visualizzati con la luminanza minima (nero) per mancanza dell'informazione colore che viaggia con i c-access.

Ben diversi sono gli idle access che il VIC-II effettua in alcuni cicli delle raster line del Display State: anche in

quel caso viene letto il ghost byte, ma il valore non viene nemmeno preso in considerazione e viene immediatamente scartato (letture "dummy").

Il dettaglio più interessante è che l'Idle State può presentarsi anche all'interno della Display Window: è ciò che accade ogni volta che una Bad Line viene ritardata o soppressa. La regola precisa chiama in causa un altro registro interno del chip, il Row Counter (RC), che conta da 0 a 7 quale delle otto raster line della riga di testo corrente si sta disegnando: quando RC raggiunge il valore 7 e non si verifica più alcuna Bad Line Condition, la display logic passa in Idle State; viceversa, torna in Display State non appena una nuova Bad Line Condition si presenta.

Modificando i bit 0-2 di $D011 nel bel mezzo della generazione di un quadro si governa quindi, di fatto, proprio questo interruttore. Tenete a mente questo concetto, perché è la chiave di lettura di tutto quello che segue.

4. Il limite del VIC-II e la nascita del VSP

Il VIC-II offre un supporto nativo allo scrolling fluido (fine scroll) tramite i bit 0-2 di $D016 (orizzontale) e $D011 (verticale).

Chiunque abbia mai programmato il VIC-II sa bene che lo scrolling hardware nativo del C64 si ferma a soli 8 pixel. I registri $D016 e $D011 permettono di spostare finemente lo schermo all'interno di una finestra di un solo carattere (da 7 a 0); oltre quel limite, tocca alla CPU farsi carico del pesante lavoro di ricopiare intere porzioni di memoria video.

E non si tratta solo dei 1000 byte della matrice: per uno scroll a colori pieni bisogna spostare anche i 1000 byte della Color RAM. In un fotogramma PAL, sull'hardware originale, semplicemente non si fa in tempo a spostare tutto.

Come detto, con il VSP si inganna il chip grafico: manipolando con precisione al singolo ciclo di clock il registro $D011, proprio mentre il VIC-II sta per generare la prima Bad Line, si modifica "al volo" la Bad Line Condition. La Bad Line scatta in ritardo di n cicli: il VIC-II salterà n colonne di fetch e inizierà a leggere la Screen RAM spostato di n caratteri. Di conseguenza, l'intero schermo viene letto come se iniziasse da un'altra posizione di memoria.

Sostanzialmente, il VSP è una tecnica per ottenere un offset orizzontale dello schermo, senza di fatto copiare i caratteri: l'intera matrice grafica del C64 può essere scrollata orizzontalmente, senza dover copiare fisicamente tutti i caratteri dello schermo. In altre parole, con il VSP si cambia rapidamente la colonna dalla quale il VIC-II comincia a disegnare lo schermo.

E poiché l'inganno agisce sul contatore interno del chip, e non su una particolare modalità video, la tecnica funziona tanto in Text Mode quanto in Bitmap Mode: diventa così possibile scrollare molto velocemente uno schermo di caratteri o addirittura un'intera bitmap.

È importante sottolineare che tutto questo avviene senza spostare nemmeno un byte della Screen RAM. L'intero effetto nasce esclusivamente dal comportamento interno del VIC-II.

Il grande vantaggio del VSP è che lo spostamento applica l'effetto sia alla Screen RAM che alla Color RAM contemporaneamente: il VIC-II infatti, le legge entrambe usando il medesimo contatore interno (VC - Video Counter).

5. L'origine hardware del Bug: Meta-stabilità e DRAM

Per decenni il VSP è stato imprevedibile: il medesimo codice funzionava su un C64 e ne causava il crash immediato su un altro. All'inizio, si pensava a differenze tra revisioni del VIC-II (6569 contro 8565), ma solo nel 2013, l'hacker, musicista e programmatore Linus Åkesson (LFT) ne ha dimostrato la causa fisica a livello hardware [3].

Quando si invia l'istruzione STA $D011 nel momento critico del clock, il VIC-II cambia improvvisamente l'indirizzo di lettura della RAM per forzare lo scroll.

Questo sfasamento crea uno stato di meta-stabilità sui bit del bus di indirizzo multiplexato della DRAM (righe e colonne). Durante quella frazione di microsecondo:

a. Alcuni bit della riga DRAM selezionata possono essere scritti con valori errati.

b. Questa corruzione non avviene su tutta la RAM, ma si concentra esclusivamente nella pagina o blocco di memoria che il bus sta indirizzando in quel preciso istante o nelle celle adiacenti.

c. Se in quella pagina di memoria risiede il codice del programma, la CPU leggerà un'istruzione corrotta e il programma andrà inevitabilmente in crash.

Le locazioni soggette a questo fenomeno sono i cosiddetti byte fragili: gli indirizzi che terminano in $7 o $F. La corruzione, quando avviene, consiste nel copiare il contenuto di un byte fragile in un altro byte fragile della stessa pagina di memoria.



Fig. 1: il "VSP Lab" di LFT in azione su un C64 affetto da VSP bug

L'evidenza raccolta negli anni dalla comunità, per quanto aneddotica, suggerisce che i C64C siano le macchine più esposte al crash da VSP; e se per qualche tempo si è sperato che i C128 ne fossero immuni, anche alcuni di loro sono stati colti in flagrante.

Come ha osservato lo sviluppatore Kodiak, quella del VSP è forse l'unica vulnerabilità in ambito 8-bit che un fisico definirebbe l'output di un sistema complesso: il concorso di alimentatore, resistenze parassite della scheda, revisione del VIC e temperatura rende ogni esemplare un caso a sé [6].

Chi volesse scoprire se il proprio C64 (reale) è tra quelli vulnerabili, può ricorrere a VSP Lab, una utility scritta dallo stesso LFT per analizzare sistematicamente il fenomeno e mappare le fragilità di una specifica macchina [11].

LFT aveva persino proposto di patchare i vecchi giochi che usano il VSP, e, in effetti, una patch del genere esiste per "Mayhem in Monsterland" [6][12].

Un avvertimento, però: la vulnerabilità dipende dal tempo trascorso dall'accensione, ovvero dalla temperatura interna del VIC-II e degli altri componenti. Una macchina che supera il test a computer freddo, potrebbe non superarlo mezz'ora dopo, o in una stanza più calda.

Per questo, un rilevamento automatico ha un valore relativo: il verdetto vale per quella macchina, in quel momento, a quella temperatura [6].

6. Esempio pratico: come il VSP "sposta" lo schermo

Supponiamo di avere una schermata di testo in memoria:

```
Screen A  ($0400)
HELLO WORLD
***********
.................
.................
```

Normalmente il VIC-II legge la Screen RAM partendo dall'indirizzo selezionato dai bit del registro $D018. Con il VSP, invece, non viene modificato $D018.

Il trucco consiste nel far credere al VIC-II che la Bad Line successiva debba avvenire in un momento diverso da quello previsto. Se il registro $D011 viene modificato esattamente durante la fase di fetch dei dati video, il contatore interno del VIC-II viene incrementato in modo anomalo, saltando in avanti di n posizioni. È come se il VIC-II iniziasse improvvisamente a leggere la memoria da una posizione diversa:

```
Lettura normale
$0400
$0428
$0450
$0478
....

Dopo il VSP
$0401
$0429
$0451
$0479
....
```

L'intero schermo appare così traslato di un carattere, senza che la CPU abbia copiato nemmeno un byte di memoria. Ripetendo questa operazione ad ogni frame si ottiene uno scrolling completamente gratuito dal punto di vista del processore. Va ricordato che il Video Counter è un contatore a 10 bit: il fetch resta quindi confinato nel blocco di 1 KB selezionato dal registro $D018 e, giunto in fondo, "avvolge" ripartendo dall'inizio del blocco stesso. Nei giochi è proprio in quel momento che vengono ridisegnate le colonne che entrano dal bordo dello schermo.

6.1. Le tre vie per domare il raster

Qualunque implementazione del VSP si scontra con lo stesso ostacolo: la scrittura in $D011 deve avvenire al ciclo di clock esatto, ma un normale interrupt raster arriva con un jitter di 0-7 cicli (caso di una IRQ Raster routine vettorizzata in ROM $FFFE-$FFFF).

Il meccanismo è il seguente: quando il VIC-II raggiunge la linea raster programmata, alza la richiesta di interrupt

immediatamente: da quel lato non c'è alcuna imprecisione. Ma la CPU non può mollare tutto all'istante: deve prima portare a termine l'istruzione che sta eseguendo in quel momento, perché le istruzioni sono indivisibili. E le istruzioni del 6510 non hanno tutte la stessa durata: si va dai 2 cicli di un NOP o di un LDA immediato, ai 3-4 degli accessi in memoria, fino ai 6-7 delle istruzioni read-modify-write su indirizzi indicizzati. Solo dopo aver finito, la CPU esegue la sequenza di servizio dell'interrupt (quella è a durata fissa, 7 cicli, e non contribuisce all'incertezza) e salta alla routine.

Dunque, tra l'istante in cui il VIC-II "chiama" e l'istante in cui la prima istruzione parte, passa un tempo che cambia da un'occasione all'altra, a seconda di quale istruzione stava girando e a che punto della sua esecuzione si trovava. Nel caso migliore l'interrupt arriva proprio sull'ultimo ciclo dell'istruzione corrente: ritardo aggiuntivo 0 cicli. Nel caso peggiore arriva sul primo ciclo di un'istruzione lunga: ritardo aggiuntivo 7 cicli.

La scena ha sviluppato negli anni diverse tecniche per ottenere la precisione al singolo ciclo; le tre più rappresentative sono le seguenti:

a. Il polling puro. Il metodo più semplice: la CPU legge in un loop stretto il registro $D012 finché non compare la linea desiderata. Il jitter residuo è pari alla durata del loop stesso e va poi raffinato con ulteriori confronti; in compenso non servono interrupt, né vettori. È l'approccio tipico dei programmi di test e degli esperimenti rapidi, al prezzo di tenere la CPU completamente occupata nell'attesa [13].

b. Il doppio interrupt. Il metodo classico di demo e giochi: il primo IRQ raster ne arma immediatamente un secondo sulla linea successiva e poi "galleggia" su una fila di NOP; il secondo IRQ, cadendo per forza all'interno di un NOP (che dura 2 cicli), arriva con un jitter ridotto a 0-1 cicli. L'ultimo ciclo di incertezza viene eliminato leggendo $D012 due volte a cavallo del cambio di linea: a seconda del jitter, la seconda lettura differisce dalla prima, e un branch preso oppure non preso (3 cicli contro 2) assorbe esattamente la differenza [14].

c. Il trucco del lightpen. Il metodo più ingegnoso e il meno conosciuto. L'ingresso lightpen del VIC-II è collegato al bit 4 della porta B del CIA 1: configurando quel bit come uscita e facendolo commutare, si simula l'impulso di una penna ottica e il VIC-II "fotografa" nei registri $D013/$D014 la posizione esatta del pennello elettronico in quell'istante. Leggendo $D013 il programma non elimina il jitter: lo misura, in cicli, e lo compensa saltando con un branch automodificato dentro una cascata di istruzioni sovrapposte, ciascuna più corta di un ciclo della precedente [15]. Il tutto senza usare alcun interrupt, il che rende il codice insensibile allo stato dei CIA e dei vettori: non a caso è la tecnica adottata da alcuni dei programmi di test del repository di VICE [8], di cui si parlerà in seguito.

Qualunque via si scelga, il risultato è lo stesso: da quel punto in poi il codice è cycle-exact, e la scrittura in $D011 può essere piazzata esattamente al ciclo 14+n della linea prescelta.

7. L'emulatore VICE (Versatile Commodore Emulator)

L'ottimo emulatore VICE è talmente accurato da includere tra le sue opzioni proprio la "VSP bug emulation": attivando questa spunta nelle impostazioni del VIC-II, l'emulatore simula la corruzione casuale della RAM tipica di alcune macchine reali [7].

Provando, ad esempio, la demo "The Lab" di Elysium, vincitrice del recente X'2026 party [4], mi sono accorto che, avviandola sull'emulatore con la "VSP bug emulation" attiva, dopo circa un minuto e mezzo dall'avvio, si blocca ogni due per tre, in quanto la demo utilizza una routine VSP. Disattivando l'emulazione del bug o, provandola su un C64 reale con DRAM particolarmente stabili, tutto fila liscio come l'olio.

Per provare il "VSP bug emulation" occorre usare l'emulatore VICE x64sc e non il vecchio x64: solo l'emulazione cycle-exact riproduce fedelmente il comportamento del VIC-II al singolo ciclo, che per il VSP è tutto.

Andate nella sezione "Settings → Display → VIC-II" del VICE x64sc e aggiungete la spunta a "VSP bug emulation", oppure andate direttamente nel file vice.ini ed aggiungete la riga:

VICIIVSPBug=1

C'è un trucco semplice per scoprire se e quando un programma usa il VSP: avviare x64sc da terminale e tenere d'occhio i messaggi che l'emulatore scrive sulla console, durante l'esecuzione. Ogni volta che il programma compie un VSP, il VICE lo annota in tempo reale con righe come quelle in figura 3.

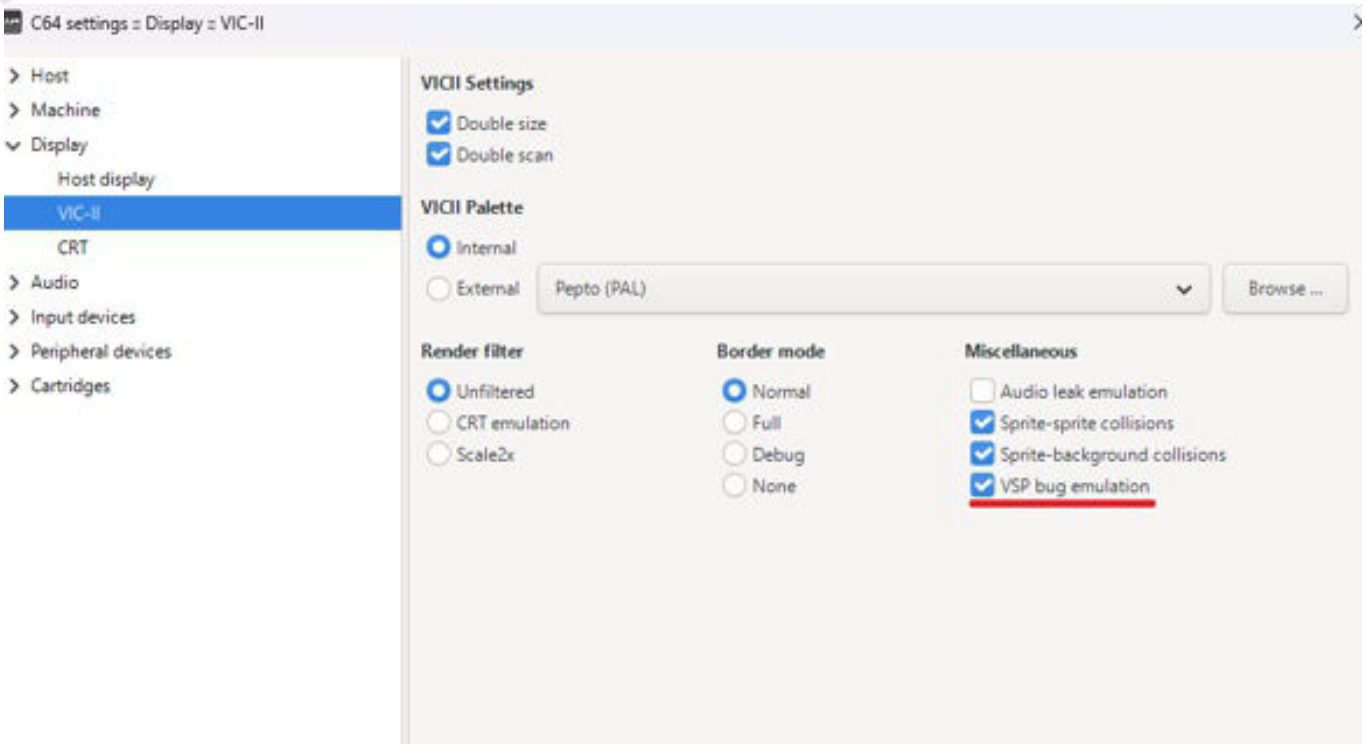


Fig. 2: schermata delle impostazioni del VICE in cui è possibile selezionare il "VSP bug emulation"

Ogni riga è un evento rilevato, con la sua posizione esatta:

a. "Line" indica la linea raster (nel formato bit-bassi/linea-assoluta, es. 2/186),

b. "Cycle" il ciclo all'interno della linea in cui è scattata la scrittura,

c. "Channel" il "canale VSP" secondo la classificazione di LFT (dipende dallo XOR tra il vecchio e il nuovo valore di YSCROLL).

Il bello è che questi avvisi compaiono anche con la "VSP bug emulation" disattivata. Dopo parecchie segnalazioni l'emulatore smette di scrivere per non inondare il log (è questo il senso dell'ultima riga visibile in fig.3), ma a quel punto avrete già la vostra risposta: quel programma usa la tecnica del VSP!

Inoltre, nel repository del VICE, è possibile trovare una sezione dedicata al "DMA delay", ovvero al VSP [8].

In questa sezione, si trovano i programmi scritti dagli stessi sviluppatori dell'emulatore, per verificare l'accuratezza dell'emulazione: i primi tre permettono di variare i ritardi da tastiera, osservando in tempo reale lo schermo scivolare di un carattere per ogni ciclo aggiunto.

I test coprono entrambi i modi di innescare il DMA delay (da non confondere con le tecniche di sincronizzazione del raster, viste al par. 6): la manipolazione di YSCROLL a metà linea e il metodo alternativo basato sul bit DEN, che sfrutta l'accensione dello schermo a raster già avviato. Come detto, i primi tre sono interattivi (da tastiera si varia al volo il ritardo della scrittura in $D011), mentre il quarto usa il trucco del lightpen, visto sopra.

8. Safe VSP: convivere con i byte fragili

Contrariamente a quanto si possa credere, il Safe VSP non elimina il bug: è possibile convivere con il VSP, anche se il bug continua a verificarsi.

Infatti, l'idea del Safe VSP descritta dallo stesso LFT [3] [10], parte da un'osservazione tanto semplice quanto geniale: poiché la corruzione può soltanto travasare il contenuto di un byte fragile in un altro byte fragile della stessa pagina, basta organizzare il contenuto della memoria in modo che i byte fragili di ogni pagina siano identici tra loro, oppure contengano soltanto dati di riempimento o tabelle sacrificabili.

In questo modo, anche se una corruzione avviene realmente, essa non produce alcun effetto: il byte colpito viene sovrascritto con un valore identico a quello che già conteneva, oppure interessa soltanto dati sacrificabili. Il software continua quindi a funzionare correttamente sulla quasi totalità delle macchine reali.

8.1. Le 3 Regole del Safe VSP

Per rendere il VSP sicuro su C64 reale (e sul VICE con l'emulazione del bug attiva), i programmatori applicano un protocollo composto da tre pilastri:

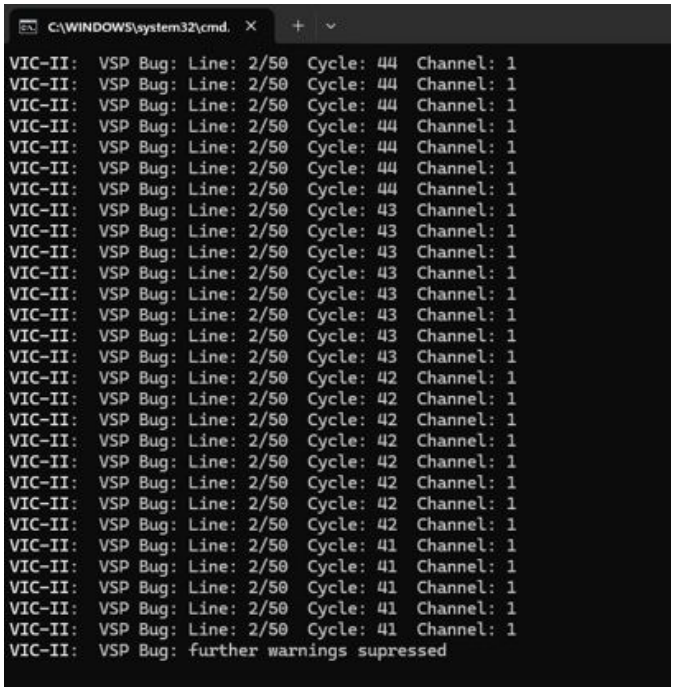


Fig. 3: I messaggi con cui x64sc segnala sulla console, ogni VSP eseguito dal programma in esecuzione

a. Isolamento del Codice (Code Placement)

Il codice della routine VSP e i restanti dati critici del programma non devono mai risiedere nella stessa pagina di memoria RAM o nell'area indirizzata durante la meta-stabilità.

La routine che esegue lo STA $D011 viene isolata in una zona di RAM "sicura".

b. Riempimento preventivo ("Safe Padding" o "DRAM Shielding")

Poiché sappiamo quali linee/pagine di memoria DRAM vengono lette dal VIC-II e dalla CPU durante il ciclo del trigger, si va a "pre-riempire" intenzionalmente quelle aree di memoria vulnerabili con valori innocui.

Se la corruzione colpisce una cella di RAM che contiene il valore $FF o $00, oppure se la zona viene riempita con istruzioni NOP ($EA), una variazione occasionale di qualche bit (ad es. un $FF che diventa $FE) non produce effetti catastrofici o viene facilmente tollerata/ripristinata al quadro successivo.

c. Allineamento Maniacale delle Tabelle e dei Cicli

Il VSP è una tecnica estremamente sensibile al timing. La modifica del registro $D011 deve avvenire in una finestra temporale di pochissimi cicli di clock; un anticipo o un ritardo di un solo ciclo può impedire il corretto funzionamento della routine o, peggio ancora, aumentare la probabilità di corruzione della memoria.

Per questo motivo il codice viene progettato affinché ogni istruzione impieghi sempre lo stesso numero di cicli, indipendentemente dalla sua posizione in memoria. Una delle principali insidie è rappresentata dagli accessi che attraversano il confine tra due pagine di memoria (256 byte): in alcuni modi di indirizzamento del 6510, come ad esempio LDA abs,X o LDA abs,Y, il passaggio da una pagina alla successiva introduce un ciclo di clock aggiuntivo. Quel singolo ciclo è sufficiente a far perdere la perfetta sincronizzazione con il raster.

Per evitare questo problema, gli assembler moderni mettono a disposizione direttive di allineamento, come ".align", che consentono di posizionare routine e tabelle dati in indirizzi accuratamente scelti. Non è raro trovare nei sorgenti delle demo lunghe sequenze di byte di riempimento ($00, $FF o $EA) il cui unico scopo è quello di far iniziare una routine esattamente all'inizio di una nuova pagina di memoria.

Anche le tabelle utilizzate dalle routine VSP vengono collocate con estrema attenzione, in modo che nessun accesso possa introdurre ritardi imprevedibili. In altre parole, il programmatore non controlla soltanto cosa esegue la CPU, ma anche quando avviene ogni singolo accesso alla memoria.

9. Critiche al Safe VSP

Va detto però che, dal punto di vista di chi sviluppa giochi, il Safe VSP non è un pranzo di gala, e c'è chi l'ha messo nero su bianco con tanto di conti [6].

Ogni pagina di RAM contiene 16 locazioni che terminano in $7 e altre 16 che terminano in $F: proteggerle tutte con dei NOP significa sacrificare 128 byte per ogni KB di codice eseguibile, cioè il 12,5% del programma trasformato in zavorra. Su un gioco con 32 KB di codice, sono 4 KB buttati, lo spazio di 64 definizioni di sprite.

Il paradosso è servito: se il senso del VSP è risparmiare tempo di CPU e memoria, ma renderlo sicuro consuma tempo di CPU e memoria, la tecnica rischia di perdere la sua ragion d'essere. È un dibattito ancora aperto, e c'è chi, come lo stesso Kodiak, sostiene che l'allarmismo attorno alla "tecnica proibita" sia sproporzionato: il crash colpisce solo una piccola minoranza di macchine, e, a rigor di logica, bollare il VSP come pericoloso significherebbe bollare come pericolosi anche giochi come Mayhem, Creatures e mezza storia della demoscene.

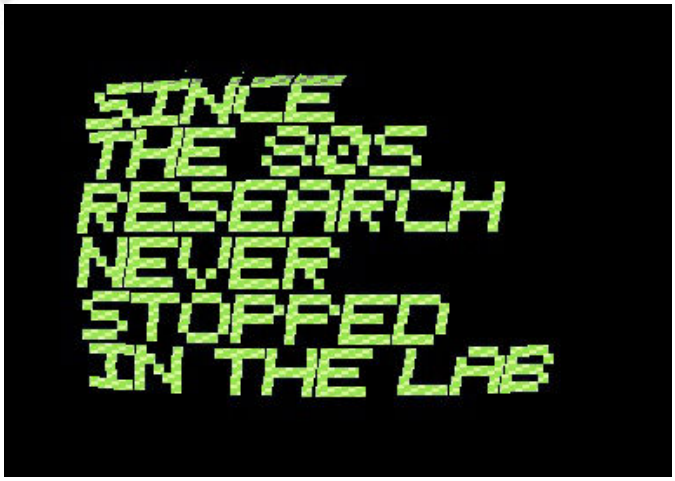


Fig. 4: schermata della demo "The lab" poco prima del momento in cui viene usata la routine VSP

10. L'ultima parola: il fix hardware

Se il Safe VSP convive con il problema, c'è chi ha scelto di estirparlo alla radice. Jens Schönfeld di Individual Computers, partendo dalle scoperte di LFT, ha progettato un fix hardware, implementato sulla scheda madre "C64 Reloaded" [5]. Schema elettrico e sorgente della logica sono pubblicati sul wiki di "Individual Computers": il VSP, dopo trent'anni, ha finalmente trovato il suo rimedio, almeno sulle nuove schede madri "C64 Reloaded".

11. Conclusioni

La storia del VSP bug riassume perfettamente lo spirito della scena Commodore: prendere un difetto di progettazione scoperto per caso, trasformarlo in una delle tecniche più spettacolari della macchina, scoprirne il lato oscuro ed infine domarlo con la forza dell'ingegno.

Ciò che più continua ad affascinarci è che, ancora oggi, il VSP sia oggetto di studio, venga impiegato anche nelle produzioni moderne e continui a essere perfezionato, ricordandoci come, ad oltre 40 anni dalla nascita del C64, questa straordinaria macchina riesca ancora a svelarci nuovi segreti!

Riferimenti

• [1] Sulle specifiche del VIC-II: Attilio Capuozzo e Antonio Savona, "Scusi l'interrupt… caro 6510", articolo n. 5, sul gruppo Facebook privato "RetroProgramming Italia – RP Italia (A division of RetroCampus)": https://www.facebook.com/groups/retroprogramming/posts/1663608240986091/ - PDF scaricabile da: https://drive.google.com/file/d/1g7SuNcG9a3NYuteVfW3BPe0ngrlSF1I3/view

• [2] Sul Display State, Idle State e i tipi di accesso: Christian Bauer "The MOS 6567/6569 video controller (VIC-II) and its application in the Commodore 64". Link: https://www.cebix.net/VIC-Article.txt

• [3] L'analisi completa di Linus Åkesson (LFT) sul "Safe VSP": https://www.linusakesson.net/scene/safevsp/index.php

• [4] La demo "The Lab" di Elysium su CSDb: https://csdb.dk/release/?id=262146

• [5] La pagina del fix hardware sul wiki della nuova scheda "C64 Reloaded" di Individual Computers (C64 reloaded - IndividualComputers), con schema e sorgente: https://wiki.icomp.de/wiki/VSP-Fix

• [6] L'articolo di Kodiak "The Future of VSP Scrolling on the C64": https://kodiak64.co.uk/blog/future-of-VSP-scrolling

• [7] L'emulatore VICE: https://vice-emu.sourceforge.io/

• [8] I programmi di test "dmadelay" nel repository del VICE: https://sourceforge.net/p/vice-emu/code/HEAD/tree/testprogs/VICII/dmadelay/

• [9] La demo "VSP&IK+" dei Meanteam, il primo programma noto a sfruttare il VSP: http://csdb.dk/release/?id=6176

• [10] "Safe VSP" di LFT, il primo programma che funziona nonostante la corruzione: http://csdb.dk/release/?id=115690

• [11] "VSP Lab V1.1" di LFT, per verificare se un dato C64 è vulnerabile: http://csdb.dk/release/?id=120810

• [12] Il gioco "Mayhem in Monsterland" patchato, su CSDb: https://csdb.dk/release/?id=196853

• [13] L'attesa attiva su $D012 e il confronto con gli interrupt raster, sul wiki Codebase64, "Introduction to Raster IRQs": http://codebase.c64.org/doku.php?id=base:introduction_to_raster_irqs

• [14] La tecnica del doppio interrupt sul wiki Codebase64, "The Double IRQ Method": https://codebase.c64.org/doku.php?id=base:the_double_irq_method&s%5b%5d=double&s%5b%5d=irq&s%5b%5d=method

• [15] La sincronizzazione del raster tramite lightpen sul wiki Codebase64, "Stable raster with lightpen": https://codebase.c64.org/doku.php?id=base:stable_raster_with_lightpen

LightTreeFM: programmare un file manager per AmigaOS senza essere un programmatore

di Danilo Savioni

Premessa

Mi chiamo Danilo Savioni, sono Tecnico di Radiologia e da sempre appassionato di retrocomputing. Non ho una formazione da programmatore e non conosco alcun linguaggio di programmazione. Ho però una buona cultura informatica generale, maturata negli anni sia per lavoro sia per passione. Lo scopo di questo articolo è raccontare e condividere l'esperienza che ho maturato durante lo sviluppo di LightTreeFM, il mio file manager per AmigaOS, attraverso l'uso dell'Intelligenza Artificiale senza sapere nulla di programmazione.

Prima di andare oltre, però, credo sia necessario chiarire subito un punto fondamentale: LightTreeFM è stato sviluppato interamente attraverso il vibe coding, utilizzando Codex in modalità di "accesso completo" sul mio PC di sviluppo. Non ho scritto personalmente una sola riga del codice che compone il programma perché, molto semplicemente, non sarei in grado di farlo. Inoltre si tratta della mia prima esperienza di utilizzo di Agenti IA per creare codice, motivo per cui questo progetto vuole essere anche un percorso di formazione e crescita personale. Per questo motivo, probabilmente, non saprò rispondere a eventuali domande particolarmente tecniche sull'implementazione interna di LightTreeFM. Il programma, infatti, è nato prima di tutto come una sfida, come un esperimento. Un modo per cercare una risposta concreta a una domanda che continuava a tornarmi in mente:

"fin dove può arrivare oggi una persona che ha un'idea abbastanza precisa di ciò che vuole realizzare, conosce abbastanza bene la macchina per cui vuole farlo, ma non sa programmare, utilizzando l'intelligenza artificiale come strumento di sviluppo?"

Fase iniziale

La prima fase è stata di esplorazione. Dovevo capire cosa mi servisse per sviluppare per AmigaOS su un sistema moderno. Quindi, armato di piano OpenAI Plus, ho iniziato raccogliere informazioni attraverso ChatGPT in modalità "advanced search". La domanda era:

"Qual è la tool chain più vantaggiosa e semplice da utilizzare per sviluppare un file manager per Amiga?".

Dopo diverse valutazioni e iterazioni sono arrivato alla seguente soluzione:

1. Ambiente di sviluppo: Windows, con Codex come supporto alla programmazione e Git per il controllo delle modifiche.
2. Linguaggi utilizzati: ANSI C89 per la maggior parte del progetto e Assembly m68k per alcuni componenti specifici.
3. Organizzazione dei sorgenti: src contiene le implementazioni, include le interfacce pubbliche, tools, i programmi ausiliari e modules src, i moduli caricabili.
4. Versionamento: BUILD_VERSION.txt contiene il numero della build, l'etichetta descrittiva e la data. Questi dati vengono usati per generare i nomi dei pacchetti e degli snapshot.
5. Orchestrazione della build: lo script PowerShell build-vbcc.ps1 gestisce l'intero processo di compilazione, collegamento, creazione delle fixture e packaging.
6. Compilazione C: VBCC, attraverso vc.exe, cross-compila i sorgenti per AmigaOS con il target aos68k e l'ottimizzazione -O1.
7. SDK e librerie: la compilazione utilizza gli header dell'Amiga NDK, in particolare Include_H, insieme alle librerie amiga.lib e vc.lib.
8. Compilazione Assembly: vasm, tramite vasmm68k_mot, assembla il codice per Motorola 68000 nel formato Amiga Hunk. Viene utilizzato, per esempio, per trasformare ptplayer.asm in ptplayer.o.
9. Verifica a runtime: la distribuzione viene provata attraverso WinUAE e, quando necessario, su hardware Amiga reale. Le configurazioni considerate comprendono Workbench 2.05 e AmigaOS 3.x.
10. Collaudo funzionale: i controlli runtime comprendono l'esecuzione delle fixture, le matrici di regressione, l'analisi

dei log, il confronto degli screenshot, il consumo di memoria e le operazioni che coinvolgono floppy e dispositivi.
11. Percorsi di build alternativi: Makefile.vbcc replica la configurazione principale per VBCC, mentre SMakefile mantiene un percorso legacy basato su SAS/C.
12. Chiusura del ciclo: una modifica al codice non è considerata completata finché la compilazione non è riuscita, la distribuzione versionata è presente in cartella f:\lightTreeFM\Dist e lo snapshot corrispondente è stato aggiornato in src_History.

Inizio dello Sviluppo

Ho passato i primi giorni a fare brainstorming con ChatGPT su cosa avrebbe dovuto essere LightTreeFM, partendo dai concetti base che avevo precedentemente maturato in autonomia. In sostanza ho declinato le mie ambizioni riguardo al progetto secondo i limiti e le risorse messe a disposizione da Amiga attraverso la mediazione di ChatGPT. Avevo imposto alcuni paletti:
1. L'applicazione non doveva avere dipendenze da librerie esterne. Niente MUI insomma.
2. L'applicazione doveva funzionare su Amiga baseline con almeno 2 MB di Chip RAM.
3. L'aspetto grafico doveva essere costruito su schermo Workbench 4 colori 640x256.

Contemporaneamente volevo che l'applicazione potesse:
1. Gestire il drag'n'drop, selezioni multiple e con rubber-band selection.
2. Gestire archivi lha, zip, ecc. in modo nativo con metafore di estrazione molto intuitive.
3. Gestire gli ADF in modo che si potesse visualizzarne i files all' interno e all' occorrenza estrarre solo quelli selezionati.
4. Poter ascoltare un file mod semplicemente facendo doppio click

Alla fine ho prodotto un documento simile ad un manifesto di intenti, un testo che conteneva non solo aspetti tecnici ma anche appunti sull'uso pratico, sulla gestualità dell'utente, sulla filosofia dell'ottimizzazione che doveva stare dietro al progetto. Questo documento è poi finito nella Knowledge Base del modello IA insieme ad altra documentazione:

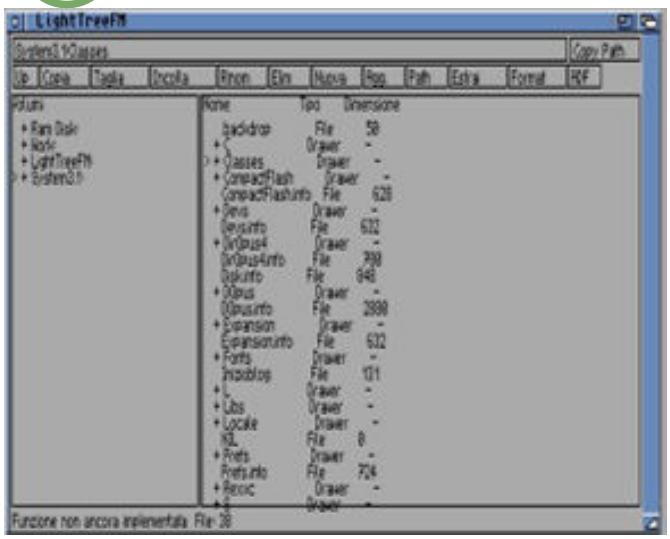
Riferimento tecnico primario: AmigaOS/NDK

La fonte normativa principale è NDK installato in C:\Amiga\NDK, utilizzato direttamente anche dalla compilazione:

• [NDK ReadMe](C:/Amiga/NDK/ReadMe-NDK.txt)
• [exec.library AutoDoc](C:/Amiga/NDK/Autodocs/exec.doc)
• [dos.library AutoDoc](C:/Amiga/NDK/Autodocs/dos.doc)
• [intuition.library AutoDoc](C:/Amiga/NDK/Autodocs/intuition.doc)
• [graphics.library AutoDoc](C:/Amiga/NDK/Autodocs/graphics.doc)
• [gadtools.library AutoDoc](C:/Amiga/NDK/Autodocs/gadtools.doc)
• [asl.library AutoDoc](C:/Amiga/NDK/Autodocs/asl.doc)
• [icon.library AutoDoc](C:/Amiga/NDK/Autodocs/icon.doc)
• [workbench.library AutoDoc](C:/Amiga/NDK/Autodocs/wb.doc)
• [trackdisk.device AutoDoc](C:/Amiga/NDK/Autodocs/trackdisk.doc)
• header ufficiali [Include_H](C:/Amiga/NDK/Include_H)

Per quanto riguarda AmigaDOS e filesystem:
ROM Kernel Reference Manual: DOS, AmigaDOS Packets, AmigaOS File Systems, Exec Memory Allocation
Oltre ad altri documenti generati con Codex usati come riferimento. Ho quindi creato un file Agents.md minimo che includesse tutte le raccomandazioni che Codex dovesse tenere presente durante lo sviluppo:

 # AGENTS.md
Regole di build e storico
- Workdir = f:\lighttreeFM
- nella segnalazione di problemi, regressioni, errori usare SEMPRE un approccio ragionato, non mettere pezze al codice per risolvere il singolo difetto ma analizzarne la causa, i collegamenti con le altre parti di codice e implementare soluzioni sistemiche.
- Dopo modifiche al codice, compilare sempre una build versionata in `Dist`.
- Aggiornare sempre lo snapshot completo del codice in `src_History`.



- Usare i nomi esistenti della repo: `Dist` e `src_History`.

- Usare `BUILD_VERSION.txt` come sorgente del numero build, etichetta e data.

- Preferire `.\build-vbcc.ps1` per compilare e produrre output versionati.

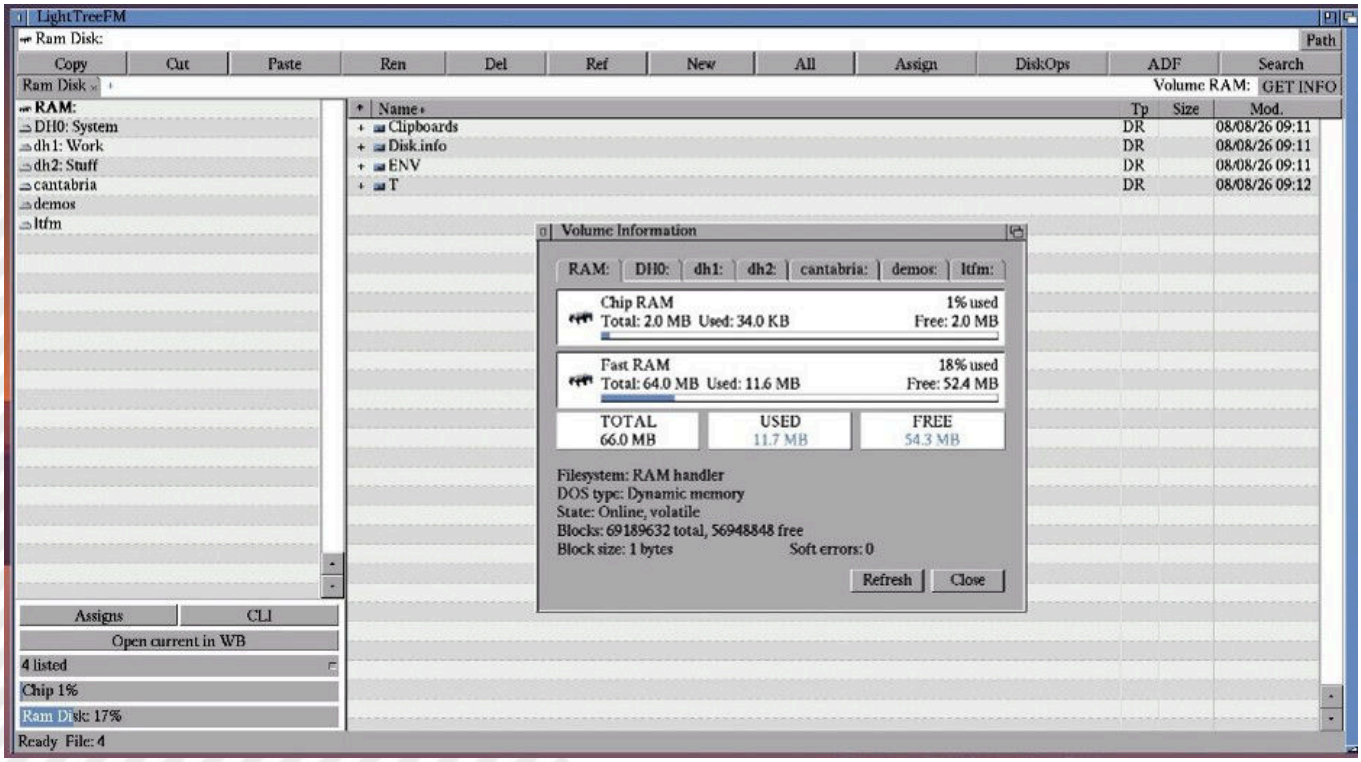
- Se serve solo rigenerare il pacchetto versionato, usare `.\package-versioned.ps1` con `-DistName` e `-HistoryName` coerenti con `BUILD_VERSION.txt`.

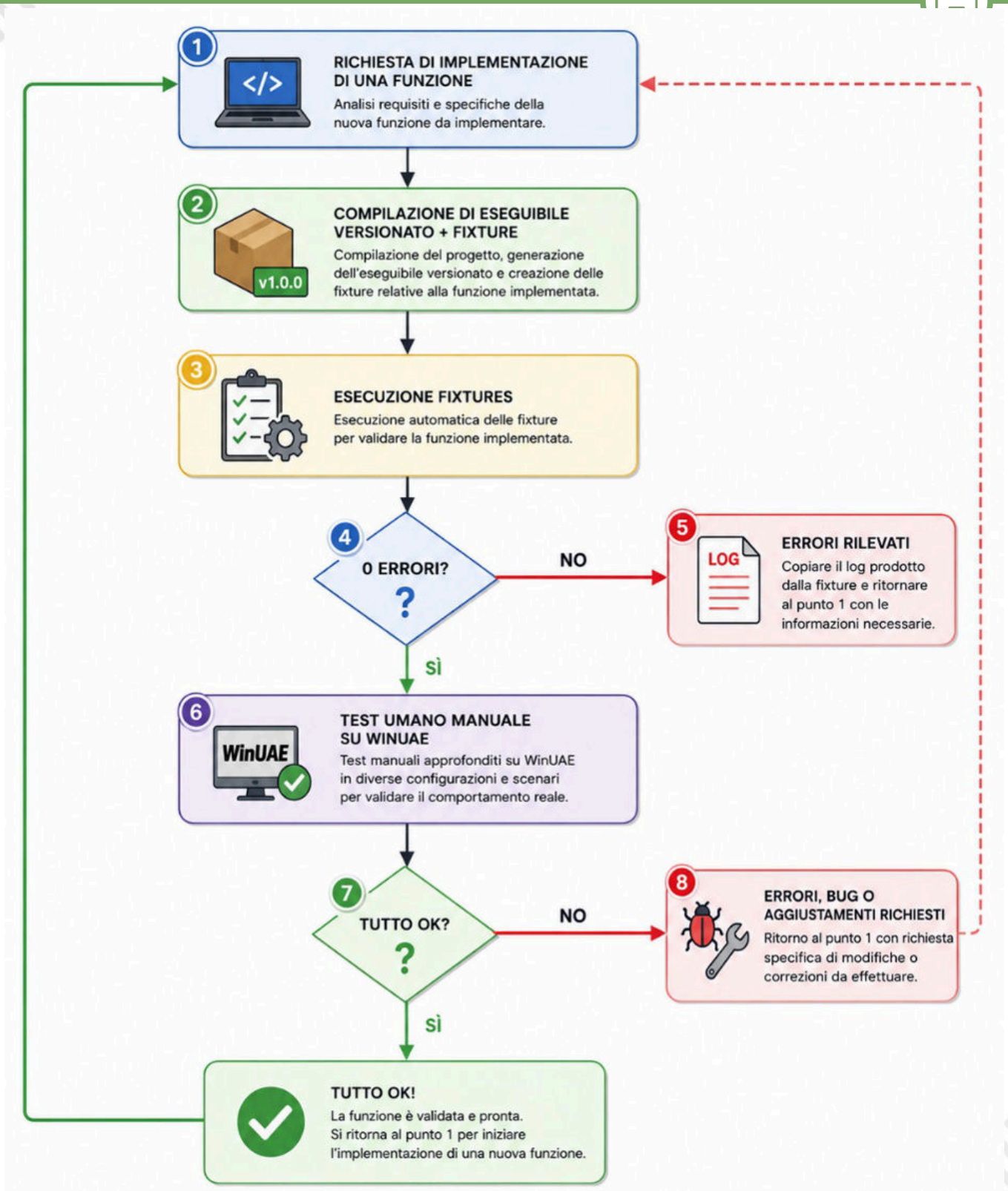
- Non sovrascrivere directory versionate esistenti a meno che l'utente lo chieda esplicitamente; in quel caso usare l'opzione di overwrite prevista dallo script.

- Prima di chiudere un lavoro che modifica sorgenti, verificare che l'output versionato in `Dist` e lo snapshot in `src_History` siano stati aggiornati oppure dichiarare chiaramente perché non è stato possibile.

I primi eseguibili

Dopo alcuni passaggi ho avuto modo di poter provare il primo eseguibile su WinUAE, nulla più di una finestra che mostrasse il Ramdisk: e DH0: nel pannello di sinistra e il relativo contenuto come file list nel pannello di destra. Nessun drag and drop, nessuna informazione sul file. Da questo punto in poi il lavoro iniziò ad assumere un ritmo abbastanza regolare: alternavo sessioni di test a sessioni con Codex, durante le quali chiedevo di implementare una determinata funzione. L'idea era semplice: procedere un passo alla volta, aggiungendo funzionalità successive fino ad arrivare, gradualmente, a quello che immaginavo sarebbe diventato il prodotto finale. Inutile dire che questo approccio così "lineare" era destinato prima o poi a mostrare i suoi limiti, soprattutto a causa di qualcosa che, all'epoca, non avevo previsto e di cui ignoravo persino l'esistenza: il codice monolitico. Codex, se non viene guidato con attenzione anche dal punto di vista dell'architettura del progetto, tende facilmente a concentrare molto codice in pochi file C. All'inizio questo non rappresentava un problema evidente, ma con l'aumentare delle funzioni e della complessità quei file diventavano sempre più grandi, intricati e difficili da gestire. Io, ignaro di tutto questo da buon principiante, cominciai quindi ad accorgermi che gli interventi di Codex diventavano progressivamente meno affidabili. Aumentavano le





regressioni, le imprecisioni e gli effetti collaterali inattesi. Correzioni che avrebbero dovuto essere relativamente semplici richiedevano spesso prompt chilometrici nei quali dovevo ricostruire ogni volta il contesto, spiegare cosa non funzionava e specificare con grande precisione cosa non doveva essere modificato. Dopo diversi tentativi di implementare nuove funzioni, o di revisionare parti del codice che non si comportavano correttamente, capii che anche un prompt estremamente dettagliato non era sempre sufficiente a mantenere l'IA concentrata sull'obiettivo principale. Il problema diventava particolarmente evidente quando, durante una stessa sessione di implementazione, ero costretto a interrompere il suo "flusso" per chiedere la correzione di regressioni introdotte poco prima. A quel punto il lavoro rischiava di trasformarsi in una catena continua di:

implementazione → regressione → fix → nuova regressione

nella quale diventava sempre più difficile capire sé stessi



realmente migliorando il programma o semplicemente aggiungendo nuove toppe a una struttura che stava diventando troppo complessa.

Il game changer

Era arrivato il momento di fermarmi e capire se avessi semplicemente raggiunto il limite di ciò che un principiante poteva realisticamente ottenere con il vibe coding, oppure se fosse stata fin dall'inizio troppo ambiziosa l'idea di sviluppare un software complesso per un sistema così particolare e di nicchia come Amiga. Mi documentai su Internet, lessi articoli, guardai video e indugiai a lungo sui consigli dei vari "guru" del vibe coding su YouTube, senza però riuscire a ricavarne una vera linea guida applicabile al mio caso. Più cercavo risposte, più avevo la sensazione di girare in tondo.

Quando ormai stavo seriamente pensando di mollare, la svolta arrivò da qualcosa di quasi paradossale. Preso dalla frustrazione, e quasi per scherzo, decisi di fare la cosa più ovvia che fino a quel momento non avevo mai fatto: chiesi direttamente a Codex perché stesse incontrando tutte quelle difficoltà e perché, con il crescere del progetto, le sue prestazioni sembrassero essere peggiorate.

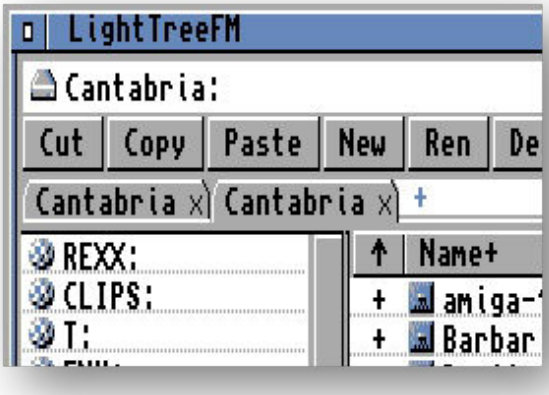
Boom!

Fu probabilmente una delle domande più importanti che gli avessi fatto dall'inizio dello sviluppo. Mi scrisse che una delle difficoltà principali era ormai la struttura stessa del progetto: il codice era diventato troppo monolitico e, di conseguenza, anche per lui era sempre più difficile orientarsi al suo interno e intervenire in modo efficace. A questo si aggiungeva un altro problema: spesso lavoravo all'interno di chat lunghissime, con un contesto enorme che finiva per ridurre l'efficienza del modello e far crescere in modo quasi insensato il consumo di token. Paradossalmente, fu proprio da questo confronto con Codex che nacque una nuova impostazione del progetto e, soprattutto, un metodo di lavoro molto più rigoroso.

Da quel momento, prima di implementare una nuova funzione, cominciai a descriverla nel modo più dettagliato possibile e a chiedere a Codex di analizzare a fondo il codice esistente, evidenziando in anticipo eventuali criticità, dipendenze o punti delicati. Il risultato di questa prima fase è un vero e proprio resoconto tecnico che analizzo prima di procedere. Spesso Codex propone più alternative, mettendone in evidenza vantaggi, svantaggi e costi in termini di complessità, memoria o impatto sul codice esistente. Scelta la soluzione che ritengo più adatta, gli chiedo quindi di creare un documento .md, solitamente con un nome del tipo nomefunzione_roadmap_implementation.md. Quel documento diventa una sorta di mappa di navigazione che Codex stesso dovrà seguire durante le successive fasi di implementazione, evitando di perdere di vista gli obiettivi e le decisioni prese all'inizio.

Questo nomefunzione_roadmap_implementation.md viene aggiornato da Codex stesso durante le fasi dell'implementazione aggiungendo informazioni, risultati dei test via fixture e warning che emergono durante la sessione, quindi il file evolve step by step adattandosi e aggiornandosi in modo automatico.

Durante il processo attivo di coding, Codex torna continuamente a questo documento: ne cita alcune parti, rilegge gli esempi di codice di riferimento annotati nelle sezioni dedicate ai diversi step di implementazione e, non di rado, rivede la strategia iniziale quando si trova davanti a scenari che erano sfuggiti anche alla sua analisi preliminare. La roadmap diventa quindi una guida dinamica,



che gli permette di adattare l'implementazione a situazioni impreviste senza perdere di vista l'architettura e gli obiettivi definiti in partenza.

Il file viene poi caricato ad ogni nuova chat del progetto che già si riferisce al suo file agents.md citato in precedenza, Codex lo legge, capisce al volo il punto preciso in cui si è arrivati nell' implementazione prima di interrompere la sessione e procede nello step successivo della roadmap ma nella nuova chat con un risparmio notevole di token.

Nelle istruzioni specifico inoltre alcuni principi ormai diventati fondamentali nello sviluppo di LightTreeFM: adottare un approccio modulare, evitare per quanto possibile di appesantire la logica già esistente e preferire, quando praticabile, soluzioni che consentano di risparmiare Chip RAM. Infine gli chiedo di procedere in modo prudente e progressivo, suddividendo l'implementazione in piccoli step verificabili e producendo a ogni passaggio delle fixture o comunque dei test intermedi. In questo modo posso capire molto presto se la direzione intrapresa è quella corretta e, soprattutto, individuare eventuali regressioni importanti prima che vengano nascoste sotto altri strati di codice.

L'aspetto Grafico

Dopo la prima release su itch.io ho ricevuto molti feedback positivi dalla community, anche se qualche utente più integralista preferiva naturalmente restare fedele all'ortodossia dettata da Directory Opus.

Tra le persone che ho avuto la fortuna di conoscere grazie a LightTreeFM c'è Martin Eriksson, grafico e musicista svedese, che, constatate le mie pressoché inesistenti capacità artistiche, si è offerto di realizzare l'icon set e il logo del programma.

Il suo contributo, però, è andato ben oltre la semplice realizzazione degli elementi grafici. Martin mi ha aiutato moltissimo anche con consigli e proposte nella cura dell'interfaccia: proporzioni, spaziatura degli elementi, equilibrio visivo e, soprattutto, gestione della limitatissima palette di quattro colori dell'interfaccia principale. Dettagli che possono sembrare secondari, ma che su Amiga fanno spesso la differenza tra un'interfaccia semplicemente funzionale e una realmente piacevole da utilizzare.

Il suo talento e la sua sensibilità grafica sono stati quindi fondamentali nel dare a LightTreeFM un'identità visiva coerente. E, oltre a essere una persona estremamente gentile e disponibile, Martin si è dimostrato durante tutto il progetto competente, preciso e professionale. Una di quelle collaborazioni nate quasi per caso che finiscono per lasciare un segno molto più importante di quanto avresti immaginato all'inizio.

Una delle difficoltà maggiori che ho incontrato utilizzando il vibe coding nello sviluppo di un progetto Amiga di questo tipo riguarda, paradossalmente, i piccolissimi aggiustamenti grafici. I punti più problematici sono spesso quelli in cui convergono diversi elementi dell'interfaccia, come barre di scorrimento, tab, info box e relativi bordi.

Ogni singola linea, effetto 3D, sovrapposizione o raccordo tra elementi deve essere descritto a Codex con estrema precisione. Quando compare un glitch grafico, un elemento spurio o magari qualche pixel residuo di una precedente implementazione, riuscire a comunicare con efficacia quale sia esattamente il problema e quale parte debba essere corretta può diventare sorprendentemente complicato. Credo che questo accada soprattutto perché ciò che, osservando l'interfaccia, sembra appartenere visivamente a un determinato oggetto può in realtà essere generato da una porzione di codice completamente diversa, oppure appartenere gerarchicamente a un altro elemento dell'interfaccia. Di conseguenza, indicare semplicemente a Codex "correggi quel bordo" o "elimina quei pixel" non sempre porta al risultato desiderato.

In questi casi ho scoperto che è fondamentale accompagnare le richieste con immagini ingrandite, indicando con

chiarezza il punto esatto da correggere. Altrettanto importante è chiedere ogni volta a Codex di analizzare prima l'origine del problema, invece di intervenire immediatamente con una modifica locale. In questo modo è costretto a riesaminare il codice sottostante, capire quale componente stia realmente generando l'artefatto e contestualizzare meglio la correzione. Questo approccio ha aumentato sensibilmente l'efficacia degli interventi e, soprattutto, ha ridotto il rischio che una piccola correzione grafica introducesse nuovi problemi altrove.

The Memory War

Due megabyte di Chip RAM come requisito minimo sono dannatamente pochi per un file manager che vuole essere davvero "comodo da usare", soprattutto quando sei un principiante e sviluppi affidandoti interamente al vibe coding. Con il crescere di LightTreeFM, e quindi del numero di funzioni e feature implementate, anche le richieste di memoria sono aumentate progressivamente.

All'inizio non mi ero posto seriamente il problema. Pensavo che la modularizzazione del codice, oltre a renderlo più gestibile, potesse essere sufficiente anche a contenere l'uso della RAM. Mi resi però abbastanza presto conto che le due cose non erano affatto equivalenti.

Fui quindi costretto a un altro stop nello sviluppo delle nuove funzioni e a una nuova analisi approfondita del codice e, soprattutto, dell'architettura generale del programma. Questa volta l'obiettivo era molto preciso: ridurre il più possibile il footprint di memoria.

Fu una delle fasi più noiose e pesanti dell'intero progetto. Iniziai effettuando una lunga serie di misurazioni dell'uso della RAM nelle diverse condizioni: durante l'avvio del programma, in idle, nel corso delle varie operazioni e dopo la chiusura. In questo modo riuscii a costruire una baseline di riferimento, indispensabile per capire se ogni modifica producesse un miglioramento reale oppure soltanto apparente.

Successivamente chiesi a Codex di realizzare una versione di debug di LightTreeFM capace di generare log dettagliati sull'utilizzo della memoria nelle varie fasi con carichi assegnabili ai diversi sottosistemi del programma. A quel punto, sulla base dei dati raccolti, gli chiesi di analizzare il progetto nel suo insieme e di individuare non semplici ottimizzazioni locali, ma un approccio più sistemico alla riduzione del consumo di RAM.

Fu un passaggio importante, perché mi costrinse a smettere di considerare la memoria come un problema da affrontare soltanto quando "finiva" e a trattarla invece come una vera risorsa di progetto, da misurare, pianificare e controllare continuamente.

Il primo intervento significativo riguardò la finestra Proprietà, che fino a quel momento era incorporata nell'eseguibile principale e occupava memoria anche quando inutilizzata. Il link audit individuò circa 47 KiB di codice e dati residenti dedicati alla funzione. La soluzione fu trasformarla in un modulo separato, caricato con LoadSeg() solo quando necessario e scaricato con UnLoadSeg() alla chiusura. Contestualmente eliminammo alcune copie temporanee e riducemmo le allocazioni intermedie. Il risultato fu un risparmio di circa 36 KiB di RAM residente, con le allocazioni temporanee ridotte da 3.746 a 624 byte e nessuna perdita di memoria dopo 50 cicli di apertura e chiusura. Fu il momento in cui compresi meglio un principio fondamentale: su Amiga non basta ridurre il consumo di RAM, bisogna evitare che rimanga residente ciò che serve solo occasionalmente. Successivamente riservai lo stesso trattamento a ogni aspetto, dalle file list alla estrazione ed esternalizzazione dei tools DiskOps, ADF , DiskCopy che prima erano in caricati all' avvio.

Da quel momento il principio diventò quasi una regola di progetto: tenere residente solo ciò che serve davvero, e caricare il resto quando necessario. Su un sistema moderno un risparmio di 36 KiB sarebbe praticamente invisibile; su un Amiga con 2 MB di Chip RAM, invece, può fare la differenza tra una funzione che esiste soltanto sulla carta e una funzione che puoi permetterti di lasciare davvero nel programma. Alla fine del processo di revisione sono arrivato ad avere risparmiato nel complesso quasi 270 Kb di Chip RAM.

Ad oggi LightTreeFM è cresciuto molto rispetto alle prime release e non nego di esserne stupito ricordando da dove ero partito, parte di queste funzioni saranno presenti

nella versione che sarà rilasciata quest'autunno.

• Navigazione a schede con stato, selezione e posizione indipendenti.

• Espansione in tempo reale ad albero di cartelle, archivi e immagini ADF.

• Confronto e sincronizzazione di due percorsi tramite TreeDuo: Compare, Update, Mirror.

• Salvataggio e riapertura delle sessioni TreeDuo come progetti .tdp.

• Drag & drop con auto-scroll, navigazione spring-loaded e cambio scheda.

• Gestione virtuale di archivi LHA, LZX, ZIP, ARC e DMS.

• Modalità LHA Low Memory per Amiga con memoria limitata.

• Esplorazione degli ADF e degli archivi con copia dei contenuti senza estrazione preventiva.

• Creazione, scrittura e verifica di immagini ADF su floppy disk.

• Proprietà avanzate: CRC32, modifica protection bits AmigaDOS, commenti e analisi ricorsiva.

• Gestione completa delle icone Workbench, Default Tool e ToolTypes.

• Creazione file .info per elementi sprovvisti.

• Ricerca ricorsiva con testo, wildcard e ritorno alla posizione originale.

• Associazioni personalizzate tra tipi di file e programmi.

• Bookmark e Assign AmigaDOS con drag & drop diretto con gestione avanzata in tools.

• Temi Classic, Dark e Light con layout adattabile e font personalizzati.

• Visualizzatore immagini DataTypes e player integrato per moduli ProTracker.

• Architettura modulare on demand ottimizzata per sistemi con soli 2 MiB di RAM.

Alla fine di questo percorso non sono certo diventato un programmatore e sicuramente continuerei a trovarmi in difficoltà davanti a problemi tecnici che un vero sviluppatore Amiga saprebbe affrontare senza esitazione. A volte mi sento come il protagonista del film "Limitless", senza IA la mia esperienza di sviluppo finirebbe subito. Ho però imparato molto su come lavorare con questi strumenti, sui loro limiti e, soprattutto, su quanto sia importante non confondere la capacità dell'IA di produrre codice con la capacità di costruire un progetto.

LightTreeFM è nato da un numero enorme di tentativi, errori, regressioni, test, ripensamenti e momenti in cui sembrava più semplice buttare tutto e ricominciare da capo. Codex ha scritto il codice, proposto soluzioni e spesso mi ha permesso di superare ostacoli che da solo non avrei avuto gli strumenti per affrontare. Ma ho imparato che bisogna comunque sapere cosa chiedere, osservare attentamente ciò che succede, mettere in dubbio le soluzioni proposte e, soprattutto, essere disposti a tornare indietro quando la strada scelta si rivela sbagliata.

Non so quanto LightTreeFM sia tecnicamente elegante se osservato con gli occhi di un programmatore esperto, né pretendo che rappresenti un nuovo modo "corretto" di sviluppare software. È semplicemente il risultato di un esperimento personale iniziato con una domanda molto semplice: quanto può riuscire a costruire una persona senza competenze di programmazione, se oggi ha accesso a strumenti come Codex?

Nel mio caso, la risposta è stata: molto più di quanto avrei immaginato.

E forse è proprio questa la cosa che mi affascina maggiormente dell'intelligenza artificiale applicata alla programmazione. L'uso dell'IA può però abbassare alcune barriere che fino a pochi anni fa sarebbero state praticamente invalicabili, permettendo a persone con idee, curiosità e molta pazienza di provare a costruire qualcosa che, diversamente, sarebbe probabilmente rimasto soltanto nella loro testa.

LightTreeFM, con tutti i suoi limiti, per me è soprattutto questo.

Sito web del progetto:
- https://dsavioni.itch.io/lighttreefm

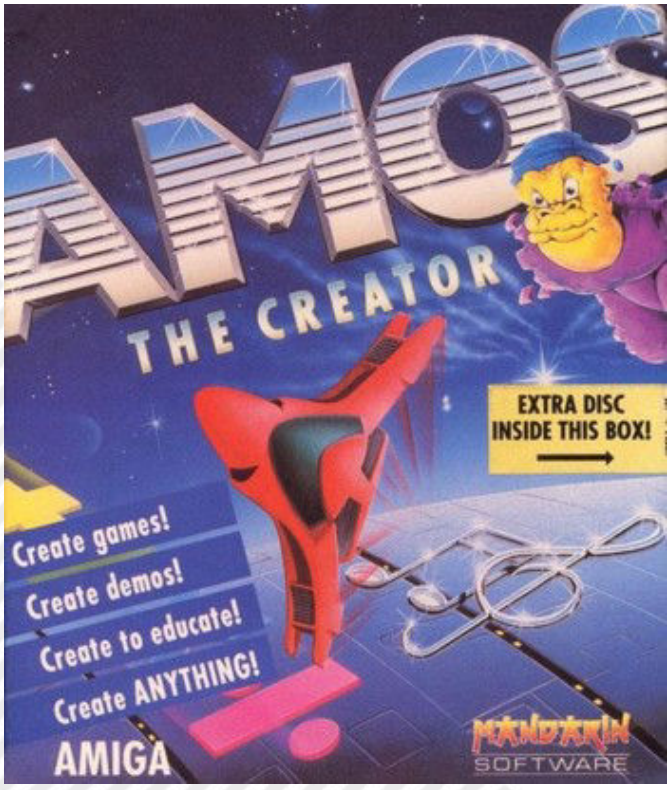
Reinventare un classico senza tradirne l'anima

Dall'Atari 400 all'Amiga: nostalgia, innovazione e nuove immagini con un pizzico di sonoro

di Leonardo Vettori, Lorenzo Mancini e Francesco Fiorentini

C'è un momento, nella vita di ogni retroprogrammatore, in cui si guarda un vecchio listato BASIC e si pensa: "E se?". È esattamente quello che è successo a noi quando ci siamo trovati davanti ad Aftermath of the Asimovian Disaster, un type-in game per Atari 400 scritto da **Tim Hartnell**, giornalista australiano e programmatore autodidatta, nonché una delle penne più prolifiche degli anni '80 nel campo delle avventure testuali e dei giochi didattici in BASIC. Il gioco faceva parte del libro **Creating Adventure Games on Your Computer**, (Ballantine Books, 1983/1984).

La domanda che ci siamo posti è stata semplice: come sarebbe stato questo gioco se fosse stato sviluppato direttamente su Amiga 500? Per rispondere, abbiamo trasformato un umile listato in una sorta di graphic novel interattiva retro-futuristica, mantenendo però intatta l'anima dell'avventura testuale.



AMOS the Creator, uno dei Basic per Amiga

La trama: sorprendentemente efficace per un semplice listato

La storia è quella dell'astronave passeggeri interstellare Asimovian, con oltre mille coloni a bordo, vittima di una misteriosa catastrofe mentre era diretta verso il Sistema Seuxarian. Due secoli dopo il disastro, il giocatore scopre il relitto alla deriva nello spazio e decide di esplorarlo. Pochi istanti dopo l'ingresso, però, la nave del giocatore ormeggiata all'esterno esplode. L'unica possibilità di sopravvivenza è trovare una Safety Pod nascosta all'interno dell'Asimovian.

Il gioco originale, per Atari 400, era quasi completamente affidato all'immaginazione: stanze collegate da uscite N, S, W, E, gestione dell'aria nella tuta, combattimenti casuali, "Solarian Credits" per acquistare equipaggiamento (luce, armi, armatura, riserve d'aria) e una schiera di antagonisti tra androidi, macchinari impazziti e alieni.

Perché AMOS?

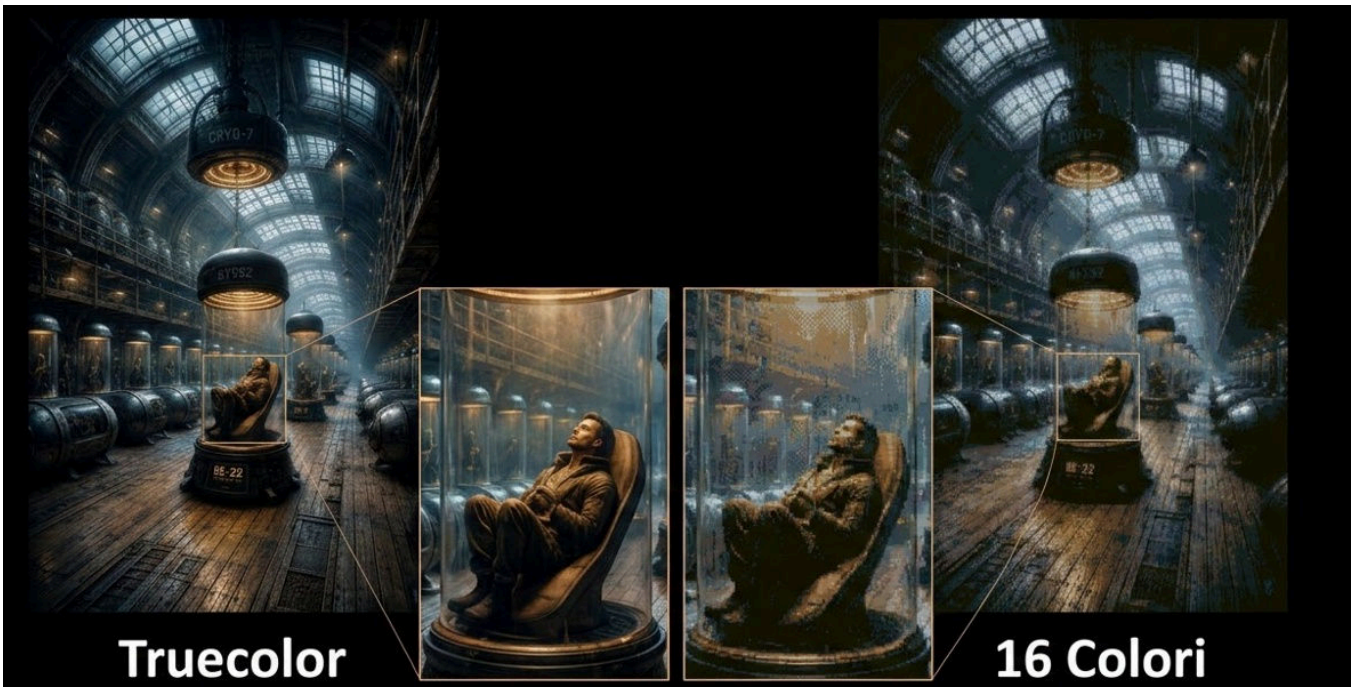
AMOS BASIC, conosciuto semplicemente come AMOS, è un celebre dialetto del BASIC sviluppato appositamente per il Commodore Amiga da François Lionet e Constantin Sotiropoulos, pubblicato nel 1990 dalla Mandarin Software. Nato dal successo dello STOS BASIC per Atari ST, AMOS offriva accesso diretto e semplificato alle capacità hardware dell'Amiga - chip grafici e sonori - senza dover ricorrere ad Assembly o C.

L'impiego di un dialetto BASIC dotato di primitive grafiche e sonore ha notevolmente semplificato il lavoro di porting. Tuttavia, non avevamo previsto alcune criticità…

Dal BASIC Atari a quello AMOS: le insidie

• Numeri di riga: AMOS li accetta, ma abbiamo preferito rimuoverli per rendere il sorgente più modificabile. Questo però ha richiesto l'aggiunta di label ad-hoc per implementare GOTO e GOSUB.

• Ingombro del testo: 40 colonne a disposizione su entrambi



Nel porting da Truecolor a 16 colori siamo riusciti a mantenere la bellezza delle immagini

i sistemi, ma i sorgenti originali a volte non rispettavano questo limite.

• Generazione dei numeri random: qui è stato il vero incubo. Il RND(n) dell'Atari restituisce un numero con la virgola tra 0 e n, mentre Rnd(n) di AMOS restituisce un intero tra 0 e n-1. Risultato? Tutte le chiamate a RND(1) del sorgente originale tornavano 0!

La soluzione: Def Fn RAND=Rnd(10000)/10000.0.

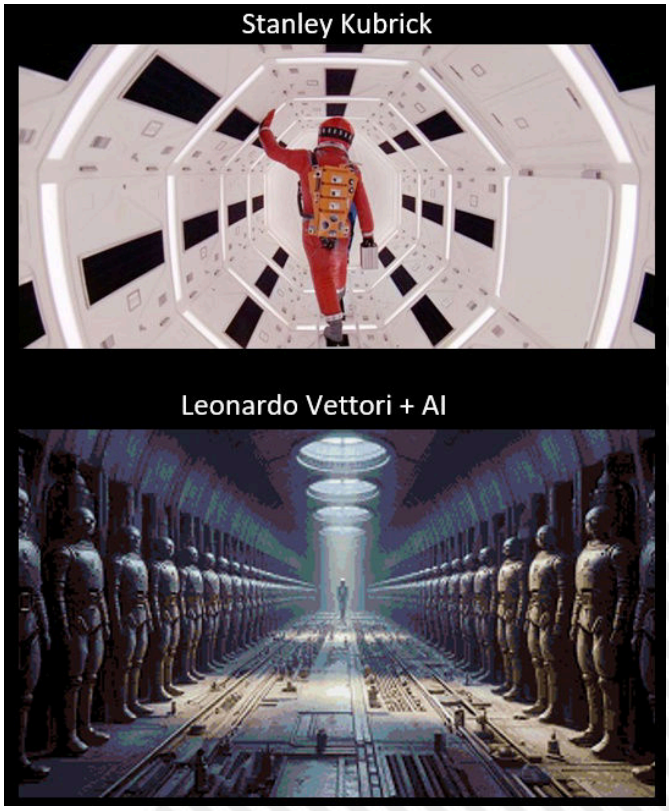
Dall'immaginazione alle immagini

L'originale Aftermath era un'avventura testuale in cui il giocatore esplorava il relitto attraverso descrizioni e comandi digitati. L'astronave, i corridoi, i laboratori e le minacce vivevano soprattutto nell'immaginazione di chi giocava.

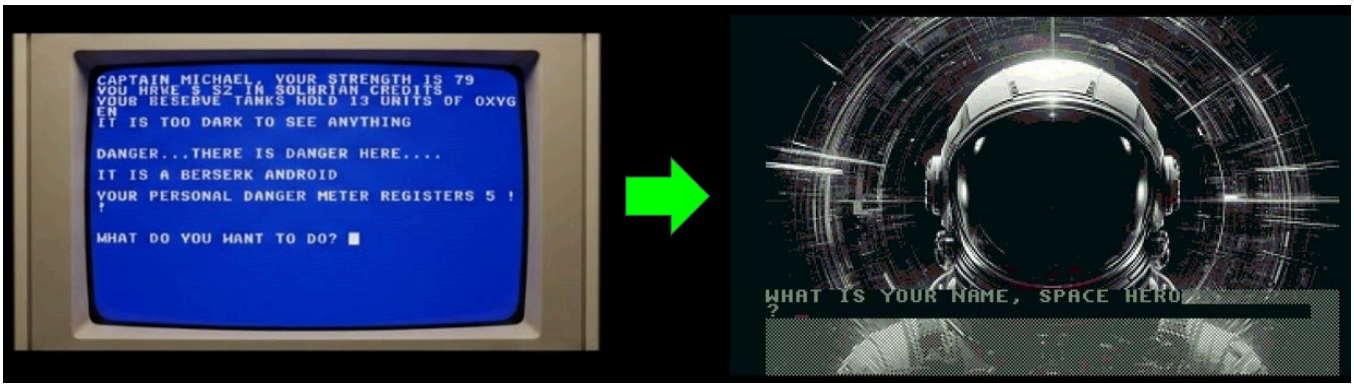
Nel nostro porting abbiamo deciso di aggiungere una componente visiva, senza tradire lo spirito dell'opera originale. Non volevamo creare personaggi animati, sequenze cinematografiche o una classica avventura grafica. L'avventura originale è raccontata in prima persona: il vero protagonista è chi si trova davanti al computer. Per questo abbiamo immaginato le stanze dell'astronave come se fossero viste direttamente dagli occhi dell'esploratore. Ogni immagine è una visuale quasi da FPS, una finestra aperta su un luogo abbandonato che il giocatore sta osservando in quel preciso istante.

Kubrick incontra la fantascienza sovietica

Per definire l'identità visiva abbiamo scelto due riferimenti molto particolari. Da una parte la simmetria e le prospettive profonde tipiche del cinema di Stanley Kubrick, capaci di comunicare ordine, mistero e senso di grandezza. Dall'altra l'estetica della fantascienza sovietica degli anni Settanta: strutture immense, tecnologia robusta, metallo consumato, apparecchiature riparate infinite volte e ambienti che



Kubrick incontra la fantascienza sovietica



In teoria sembrava la scelta migliore... In pratica non funzionava.

sembrano aver attraversato decenni di utilizzo.

Il risultato è un'astronave che non appare elegante o futuristica nel senso moderno del termine. L'Asimovian diventa una gigantesca macchina invecchiata, decadente e credibile, piena di corridoi industriali, sale di controllo, distributori automatici, laboratori e hangar che sembrano conservare ancora le tracce del disastro avvenuto due secoli prima.

La triste realtà dello spazio sullo schermo

All'inizio avevamo immaginato una soluzione apparentemente naturale: una grande immagine panoramica nella parte superiore dello schermo e il testo dell'avventura nella parte inferiore. In teoria sembrava la scelta migliore... in pratica non funzionava. Aumentando lo spazio per le immagini, il testo diventava scomodo da leggere. Aumentando lo spazio per il testo, le immagini perdevano gran parte del loro impatto.

Quando un limite diventa un'opportunità

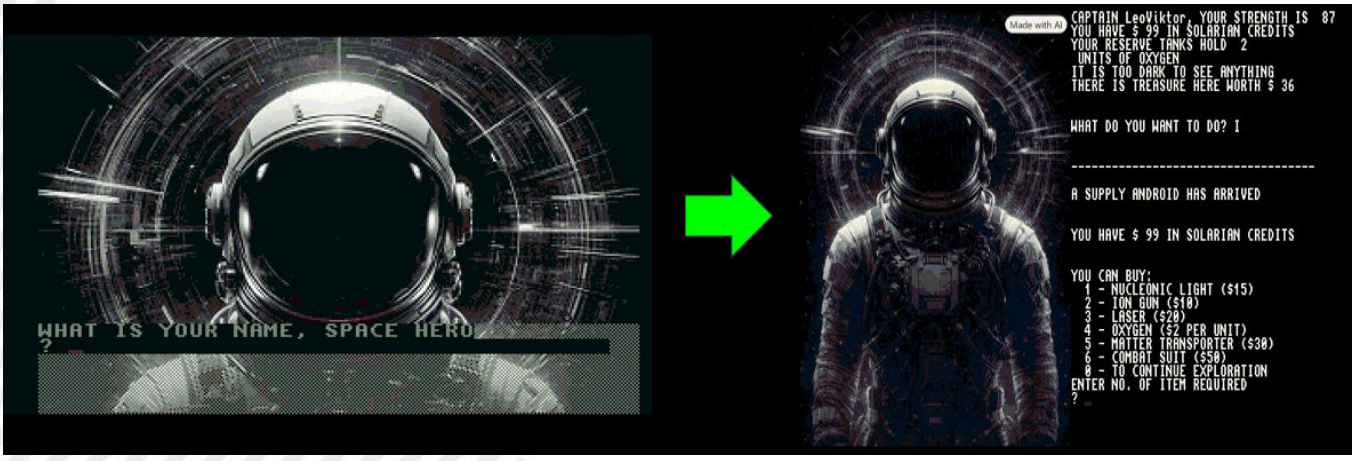
A quel punto abbiamo iniziato a ragionare in modo diverso: i dilemmi non si risolvono, si aggirano. La soluzione è stata dividere lo schermo verticalmente. Improvvisamente tutto ha iniziato a funzionare! L'immagine aveva finalmente abbastanza spazio per mostrare l'ambiente, e le descrizioni testuali non rischiavano di essere tagliate.

La cosa curiosa è che la soluzione tecnica si è trasformata in una soluzione estetica. Lo schermo ha iniziato a ricordare l'aspetto di un libro aperto: sulla pagina di sinistra c'è l'illustrazione, sulla pagina di destra il racconto. Ogni stanza diventa così una sorta di pagina illustrata di un romanzo di fantascienza che il giocatore sta sfogliando durante l'esplorazione del relitto.

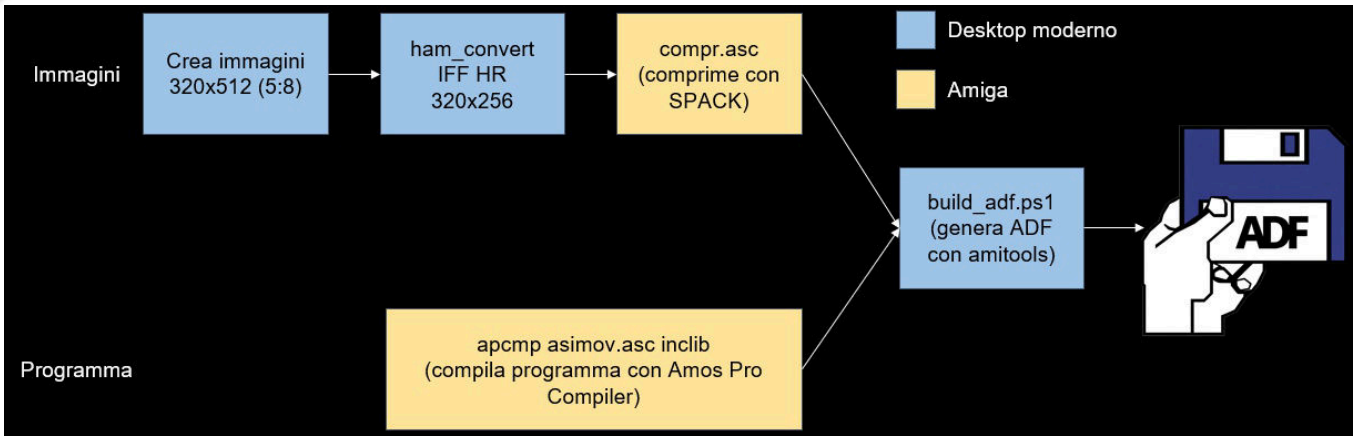
E c'è un'altra coincidenza interessante: le immagini, nate per necessità in formato verticale con aspect ratio 5:8, finiscono per avere un aspetto sorprendentemente moderno. Pur girando su un computer degli anni Ottanta, ricordano il linguaggio visivo a cui siamo abituati oggi sugli schermi dei dispositivi mobili.

Un processo di build cross-piattaforma

Per convertire le immagini in formato IFF (Interchange File Format) abbiamo usato ham_convert, un convertitore grafico freeware scritto da Sebastian Sieczko per convertire



La soluzione è stata dividere lo schermo verticalmente. Improvvisamente tutto ha iniziato a funzionare!



Un processo di build cross-piattaforma

immagini jpg/gif/png in uno dei modi grafici del Commodore Amiga. Temevamo per il passaggio a 16 colori, ma molte immagini sono mono o bicromatiche e, grazie anche al dithering, non si nota troppo la differenza.

Per far entrare il gioco in un singolo disco da 880 KB abbiamo dovuto comprimere le immagini con la funzione SPACK di AMOS, creando una procedura a parte da eseguire durante il build. Durante l'esecuzione, le immagini devono obbligatoriamente essere decompresse con UNPACK prima di essere visualizzate. Questo si "paga" in visualizzazione con una frazione di secondo di ritardo - in informatica questo è un eterno tiro alla fune chiamato Space-Time Tradeoff.

Le immagini vanno caricate al momento giusto, tenendo conto della stanza dove ci si trova, con chiamate del tipo:

```
On RO Gosub LABEL_1040, LABEL_1100, LABEL_1170, ... LABEL 1040: LDIMG[1] Print "YOU ARE IN THE FORMER RECREATION CENTER"
```

E la decompressione del banco dati in uno schermo temporaneo:

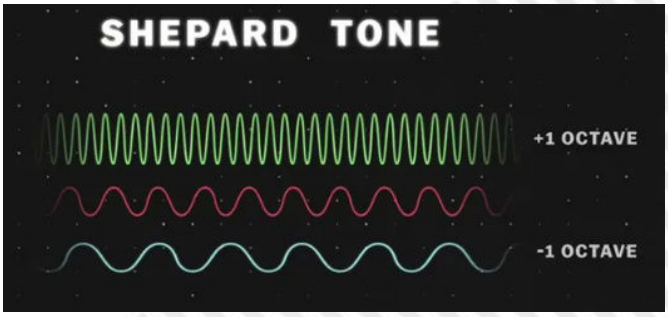
```
Procedure LDIMG[1]
  Load PRG$+"dat/ROOM"+S$+".abk", 10
  Unpack 10 To 1
  Screen 0
  Get Palette(1)
  Screen Copy 1,0,0,320,256 To 0,0,0
  Screen Close 1
  Erase 10
```

Un sottofondo che non finisce mai: lo Shepard Tone

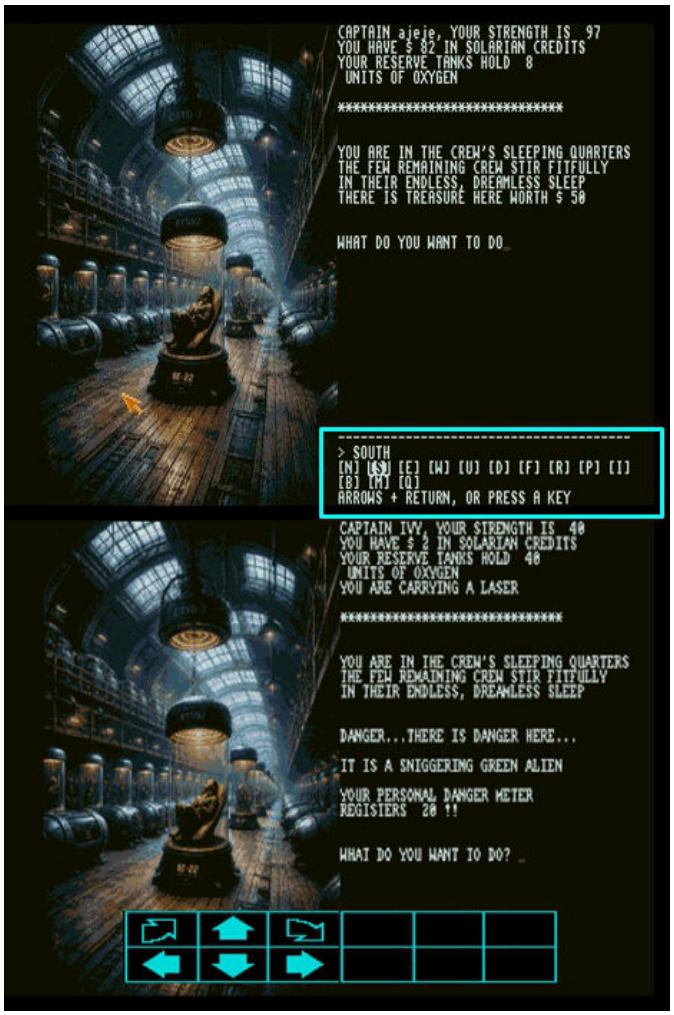
Le immagini non erano sufficienti a raccontare il senso di isolamento e inquietudine che volevamo trasmettere. Per questo abbiamo scelto di utilizzare, per tutta la durata del gioco, un particolare effetto sonoro chiamato Shepard Tone.

Chi lo ascolta ha una sensazione molto strana: sembra che il suono continui a salire senza mai raggiungere un punto finale. In realtà non sta né salendo né scendendo: è un'illusione acustica. Ideato negli anni Sessanta dallo psicologo cognitivo Roger Shepard, questo effetto è ottenuto sovrapponendo più toni separati da un'ottava. Mentre alcune frequenze aumentano di intensità, altre diminuiscono gradualmente. Il cervello percepisce così una continua salita sonora che in realtà si ripete ciclicamente. È l'equivalente sonoro delle scale infinite disegnate da Escher: si continua a salire, ma non si arriva mai da nessuna parte.

Abbiamo immaginato il suono come una sorta di "fantasma" dell'astronave. Non rappresenta un motore, non rappresenta un allarme, non rappresenta una macchina specifica. È piuttosto la voce stessa dell'Asimovian. Un rumore indefinibile che sostituisce il vuoto dello spazio e accompagna



Lo Shepard Tone



Come aggiungere un'interfaccia per i comandi

il giocatore dall'inizio alla fine.

Lo Shepard Tone è stato utilizzato o citato in diversi film, soprattutto quelli che vogliono creare una tensione continua e crescente.

La filmografia di Christopher Nolan ne è probabilmente l'esempio più famoso: Dunkirk, The Prestige, la trilogia del Cavaliere Oscuro, Inception, Interstellar.

Nostalgia attiva

Una nuova forma di nostalgia. Le immagini generate con l'aiuto dell'intelligenza artificiale non sostituiscono l'immaginazione del giocatore: la accompagnano. L'avventura rimane testuale, ma ogni stanza possiede ora una propria identità visiva. È come sfogliare una vecchia rivista di fantascienza illustrata: la narrazione continua a essere affidata alle parole, mentre le immagini suggeriscono atmosfere, dettagli e sensazioni.

In questo modo un semplice listato BASIC degli anni Ottanta continua a raccontare la stessa storia, ma attraverso un linguaggio visivo che un autore dell'epoca poteva solamente immaginare utilizzando il BASIC. Questa, forse, è la forma più autentica di nostalgia: non conservare il passato sotto vetro, ma permettergli di continuare a evolversi senza perdere la propria anima.

Ingegnerizzare il metodo

Non si tratta di sostituire i classici, ma di dare loro una seconda vita. Il vero risultato non è il porting di un vecchio gioco, ma aver dimostrato che chiunque può prendere la propria avventura preferita e trasformarla in un'esperienza nuova, accessibile e contemporanea, mantenendo intatto ciò che l'ha resa speciale.

Tra le possibilità che non abbiamo esplorato:

- aggiungere un'interfaccia per i comandi (grafica o testuale?)
- aggiungere un pannello di stato con le statistiche del giocatore
- aggiungere una mappa dell'avventura
- aggiungere dettagli alla descrizione degli ambienti
- ampliare l'esperienza di gioco (nuove stanze, nemici...)
- aggiungere enigmi per rendere l'esperienza più stimolante
- convertire altre avventure per continuare il metodo

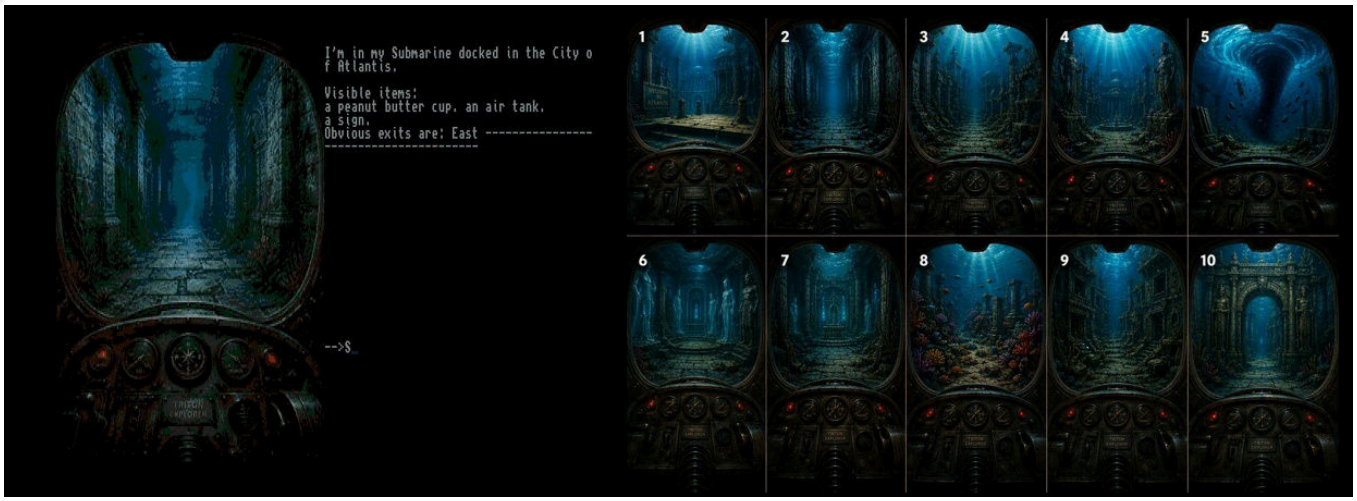
La skill atari-to-amos.md

Obiettivo: chiunque deve poter essere in grado di prendere un vecchio sorgente e modernizzarlo. Si può addestrare un LLM a seguire un metodo tramite le "Skill". Una skill è semplicemente un file di testo scritto in linguaggio naturale, che spiega dei passi di un procedimento.

Al termine della conversione di Asimovian, abbiamo scritto la skill atari-to-amos.md, che include:



Una schermata del gioco



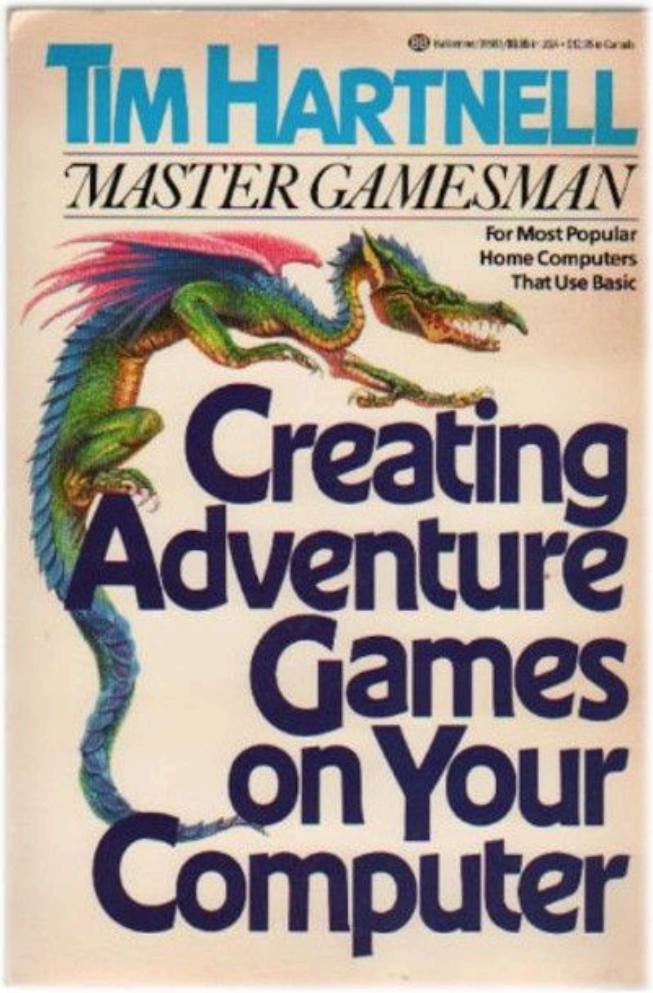
Atlantis - Work in Progress

• Errori da evitare nella conversione dei sorgenti Atari (es. problema del RND)

• Come capire qual è l'istruzione del cambio stanza

• Una checklist da eseguire alla fine della conversione

Fatto questo, ci siamo "tappati gli occhi" e l'abbiamo eseguita su un sorgente completamente diverso, di un altro autore e per un altro BASIC (QBasic). Dopo qualche piccolo aggiustamento, la conversione è riuscita!



Il libro da cui e' stato tratto il gioco

Atlantis - Work in Progress

Un'altra avventura, questa volta scritta in QBasic, al cui porting stiamo lavorando con il nostro metodo.

Se avete suggerimenti per altre avventure su cui lavorare, siamo ansiosi di ascoltarli, oppure scriveteci alla mailbox: retromagazine.redazione@gmail.com

Il team

• **Francesco Fiorentini** - Direttore della rivista RetroMagazine World - Concept e Codice

• **Leonardo Vettori** - Game designer - Concept e Grafica

• **Lorenzo Mancini** - Manager software di giorno, Retrodev di notte - Codice ed esperimenti LLM

• **Fabrizio Radica** - Docente e Game developer - Sonoro

Ah, se qualcuno fosse interessato a lavorare con noi saremo felici di inserirlo nel team!

Come giocare la nostra avventura

Download del gioco:
- https://retromagazine.net/download/asimov_1.0.adf

Nel prossimo numero metteremo il link al repository GITHUB e parleremo della possibilita' di trasformare i listati AMOS in file eseguibili su browser utilizzando un tool che abbiamo pubblicizzato sulla nostra pagina qualche settimana fa.
Non perdetevi quindi il prossimo numero!

Il cavaliere “de Méré” - per C64, per principianti

di Eugenio Rapella

Immaginate di lanciare un dado N=2 volte. Pensate che l’evento A(2)={nei 2 lanci si è ottenuto almeno un 6} si verifichi più facilmente del suo contrario? Non credo proprio: l’uscita del 6 (o di un qualunque altro numero prefissato) ha probabilità 1/6 e ottenerlo almeno una volta in due lanci non è poi così facile. Ben diverso il discorso nel caso N=20: se lanciate un dado 20 volte, è molto più probabile aver ottenuto almeno una volta il “6” piuttosto del contrario.

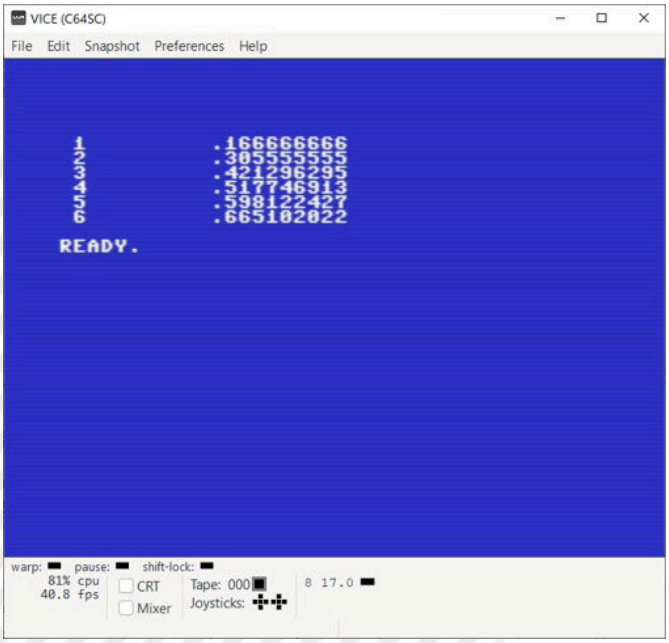
Qual è il minimo numero N di lanci che occorre eseguire perché l’evento A(N)={negli N lanci si è ottenuto almeno un 6} sia più probabile del suo contrario?

Molto spesso, in problemi di Calcolo delle Probabilità in cui si chiede di determinare la probabilità che qualcosa succeda “almeno una volta”, conviene calcolare la probabilità dell’evento contrario. L’evento contrario di A(N) è A’(N)={in tutti gli N lanci si è ottenuto un numero diverso da sei}. Nel singolo lancio, la probabilità di ottenere un numero diverso da 6 è 5/6. La probabilità di ottenere numeri diversi da 6 in N lanci è (5/6)^N (gli esiti dei vari lanci costituiscono una famiglia di eventi completamente indipendente e la probabilità che se ne verifichino un certo numero si ottiene moltiplicando le probabilità di ciascuno di essi) per cui la probabilità di ottenere almeno un 6 in N lanci è:

P(A(N)) = 1 - P(A’(N)) = 1 - (5/6)^N.

Per calcolare P(A(N)) per N=1,2,.. basta una calcolatrice, ma, visto che abbiamo a disposizione il nostro C64 … Ecco qui:

100 poke 53280,6:print chr$(147),chr$(5)
110 for n=1 to 6:print n,1-(5/6)^n:next



Schermata 1

Come si vede dalla “Schermata 1”, il minimo numero di lanci perché la probabilità di ottenere almeno un 6 sia più del 50% è QUATTRO.

(Un altro modo per determinare il valore consiste nel risolvere la disequazione 1 - (5/6)^N > (1/2) da cui (5/6)^N < (1/2); passando ai logaritmi, N*ln(5/6) < ln(1/2). ln(5/6) è circa - 0,1823, ln(1/2) è circa - 0,6931 per cui N > (-0,6931)/(-0,1823) = 3,8020. Il più piccolo valore intero è 4).

Il Cavaliere de Méré (pseudonimo di Antoine Gombaud; siamo attorno al 1600) era un accanito giocatore d’azzardo; aveva calcolato che P(A(4)) è leggermente maggiore di 1/2 (in realtà con una argomentazione non corretta, ma comunque ci aveva azzeccato) per cui scommetteva su questo evento e, alla lunga, vinceva.

Me lo immagino convincere i suoi potenziali avversari che, essendo sei le facce di un dado, di cui una sola vincente, ci sarebbero voluti sei lanci per un gioco equo … mentre lui si accontentava di quattro! Inoltre, dato che P(A(4)) è circa 0,5177, il suo vantaggio non era tale da insospettire i malcapitati: alla lunga vinceva, ma perdeva anche abbastanza spesso…

Doveva avere un bel po’ di tempo libero perché aveva pensato che:

“lanciando due dadi, essendo il DOPPIO SEI sei volte meno probabile del 6 nel lancio del singolo dado, l’evento B = {lanciando 24 volte due dadi si ottiene almeno un doppio 6}, ottenuto moltiplicando per 6 il numero di lanci di A(4), doveva avere la stessa probabilità di A(4)”.

Scommetteva dunque su questo evento aspettandosi di vincere a lungo andare.
Si accorse invece che perdeva…

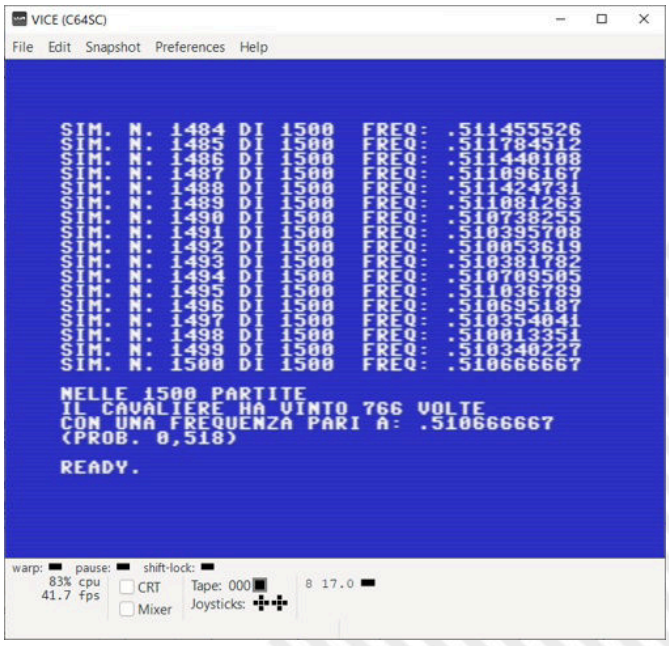
Lanciando una coppia di dadi, la probabilità di avere un doppio 6 è 1/36 (in effetti, sei volte meno probabile del 6 nel lancio di un singolo dado) e dunque la probabilità di NON ottenere il doppio 6 è 35/36 (molto elevata: più del 97%). Con ragionamento analogo al precedente, la probabilità dell'evento B(N) = {lanciando N volte una coppia di dadi si ottiene almeno un doppio sei} è P(B(N)) = 1 – (35/36)^N.
Per N=24 si ottiene P(B(24)) = 0,4914 (circa), un po' meno del 50%. Il ragionamento del cavaliere è dunque errato, ma chissà quante partite deve aver giocato per rendersi conto che perdeva non per i capricci del caso, ma perché il gioco era leggermente svantaggioso!
Comunque, il cavaliere sottopone la questione a Pascal che risponde segnalando l'errore e fornendo la probabilità corretta. Tra l'altro, P(B(25)) è all'incirca 0,5055 … sarebbe bastato un lancio in più per rendere il gioco favorevole, se pure di poco.

È ora di far entrare in gioco il nostro amato Commodore. Nel PROGRAMMA 1, il C64 simula il comportamento del cavaliere nel gioco con un solo dado:

```
100 w=rnd(-ti):print chr$(5):f=1/6:poke 53280,6
110 poke 53281,6:print chr$(147):print"programma 1"
120 print:print"la prob. che lanciando"
130 print"4 volte un dado"
140 print"si ottenga almeno"
150 print"un sei e' 0,518":print
160 input"quante simulazioni ";ns:print
170 for i=1 to ns
180 for j=1 to 4
190 if rnd(1)<=f then t=t+1:goto210
200 next j
210 print"sim. n.";i;"di";ns;" freq:";t/i
220 next i
230 print:print"nelle";ns;"partite"
240 print"il cavaliere ha vinto";t;"volte"
250 print"con una frequenza pari a:";t/ns
260 print"(prob. 0,518)"
```

Alla 100-110 l'inizializzazione del generatore di numeri casuali, la pulizia dello schermo e le colorazioni di rito. Qualche riga di spiegazione e, alla 160, la richiesta del numero di partite che vogliamo far giocare al cavaliere (variabile NS).
Alla 170 si apre il ciclo principale: per ogni simulazione (la variabile I funge da contatore), il ciclo FOR-NEXT da 180 a 200 si incarica dell'esecuzione dei lanci.
Se e quando RND (1), che restituisce un valore casuale tra 0 e 1, è minore o uguale a 1/6, vuol dire che è uscito il benedetto 6. In questo caso, il totale delle vincite del cavaliere (variabile T) viene aggiornato e si va direttamente alla 210 senza bisogno di completare il ciclo FOR-NEXT. Se, al contrario, alla 210 si giunge senza il GOTO 210 dell'istruzione 190, significa che il 6 non è uscito nei quattro lanci e viene presentata la situazione (istr. 210) senza l'incremento della variabile T. In ogni caso, alla 210 viene stampata la situazione attuale e si procede ad una nuova simulazione (NEXT I alla 220).
L'obiettivo del PROGRAMMA 1 non è certo la velocità di esecuzione, ma è piuttosto quello di presentare continuamente la situazione. Anche la ripetuta stampa di NS consente all'utilizzatore di avere un'idea di quanto manca alla fine del processo.
Completate le simulazioni, dalla 230 in poi c'è la stampa della situazione finale.

Ecco un esempio di output:

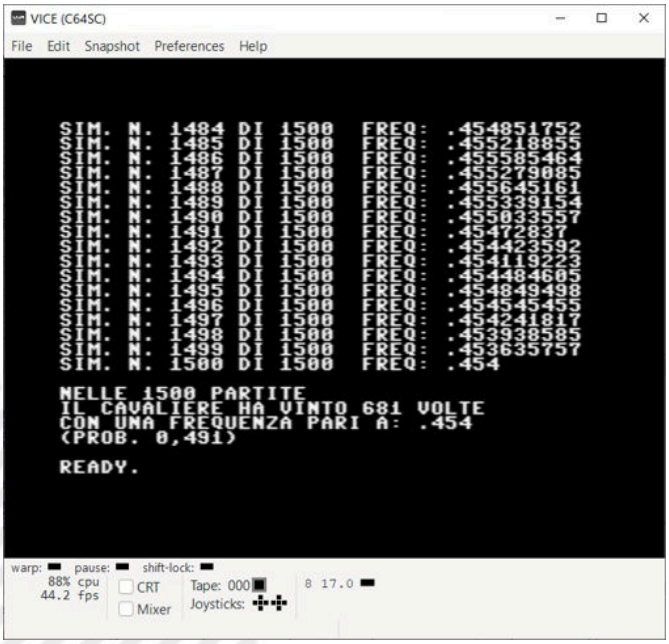


Il seguente PROGRAMMA 2 si occupa di gestire le giocate del cavaliere nel gioco con i 24 lanci dei due dadi. È praticamente un clone del PROGRAMMA 1 con queste differenze:

Istruz. 100-150: la soglia di RND(1) entro la quale c'è da segnalare il "doppio 6" è portata, naturalmente, a 1/36 (variabile F); diversa colorazione e diversa presentazione. Istr. 180: in ogni partita si eseguono fino a 24 lanci.
Ecco qui:

```
100 w=rnd(-ti):print chr$(5):f=1/36:poke 53280,0
110 poke 53281,0:print chr$(147):print"programma 2"
120 print:print"la prob. che lanciando"
130 print"24 volte due dadi"
140 print"si ottenga almeno"
150 print"un doppio sei e' 0,491":print
160 input"quante simulazioni ";ns:print
170 for i=1 to ns
180 for j=1 to 24
190 if rnd(1)<=f then t=t+1:goto 210
200 next j
210 print"sim. n.";i;"di";ns;" freq:";t/i
220 next i
230 print:print"nelle";ns;"partite"
240 print"il cavaliere ha vinto";t;"volte"
250 print"con una frequenza pari a:";t/ns
260 print"(prob. 0,491)"
```

Ed ecco un esempio di output del PROGRAMMA 2:



Certo che se il nostro cavaliere avesse avuto a disposizione un Commodore 64, avrebbe risparmiato un bel po' di tempo e, forse, di denaro…

Il mio articolo finisce qui. La parte che segue si spiega da sola; vedete voi se proseguire la lettura; io l'ho trovata interessante.

RICHIESTA INVIATA A CLAUDE AI:

Io ho già scritto un articolo di circa 1200 parole per una rivista di retrogaming e retrocomputing (RetroMagazine World). Ho una mezza idea di far seguire il mio lavoro (che ho già completato e non intendo modificare) con quello che produrrai tu. I lettori (ed il sottoscritto, naturalmente) avranno modo di fare un confronto che, penso, sarà istruttivo. Vediamo come va. Pronto? Ecco le indicazioni:

* titolo dell'articolo:
Il cavaliere "de Méré"
(per C64, per principianti)
* estensione: circa 1200 parole.
* viene descritto il gioco che il Cavaliere proponeva: lanciava 4 volte un dado e scommetteva sull'uscita di almeno un sei. Aveva calcolato che il gioco era favorevole (e, in effetti, lo è anche se il cavaliere aveva argomentato in modo scorretto).
* il cavaliere pensava che, moltiplicando per sei il numero di lanci nel caso del lancio di una coppia di dadi, avrebbe avuto la stessa probabilità >1/2 di ottenere il "doppio sei". In realtà la probabilità di avere un doppio sei lanciando 24 volte una coppia di dadi è di poco al di sotto del 50%. Deve aver giocato davvero tanto per rendersi conto che qualcosa non andava nel suo ragionamento. Scrive a Pascal che gli risolve il problema.
* dovresti redigere due programmi di simulazione per il Commodore 64 (quindi niente CLRSCR, REPEAT ecc. solo CHR$(..) cicli FOR NEXT, GOTO "numero"). Il PROGRAMMA 1 simula il primo gioco (4 lanci di un dado), Il PROGRAMMA 2 il secondo gioco (nella simulazione si può già tener conto che la probabilità di un DOPPIO SEI nel lancio di una coppia di dadi è 1/36).
* L'obiettivo di entrambi i programmi non è la velocità di esecuzione, ma si vuole veder sullo schermo del C64 l'evolversi della situazione; il numero delle simulazioni è scelto dall'utente all'inizio del programma.
Uso CLAUDE in versione gratuita. Vediamo cosa succede.

Sono pronto! Questo è un tema bellissimo — il Cavaliere de Méré, Pascal, e la nascita del calcolo delle probabilità. Scrivo l'articolo in italiano per RetroMagazine World, con i due programmi in BASIC del C64.

Il cavaliere "de Méré"

(per C64, per principianti)

C'è una storia, nei libri di matematica, che inizia nei salotti parigini del Seicento e finisce — almeno per noi — sullo schermo blu di un Commodore 64. Protagonista: Antoine Gombaud, che amava farsi chiamare il Chevalier de Méré. Non era un matematico, ma era qualcosa di altrettanto pericoloso: un giocatore d'azzardo con la pretesa di ragionare.

Il primo gioco: quattro lanci, un sei

Il gioco preferito del Cavaliere era semplice. Si prende un dado, lo si lancia quattro volte: se esce almeno un sei, il Cavaliere vince; altrimenti perde. La posta in gioco era sempre equivalente — metà e metà — eppure il Cavaliere vinceva, sistematicamente, nel lungo periodo. Aveva capito, con il fiuto del giocatore esperto, che il gioco era favorevole. Ma perché?

Il suo ragionamento, lo ammettiamo, era sbagliato. Sosteneva che, poiché il sei ha probabilità 1/6 di uscire, in sei lanci sarebbe uscito "una volta per definizione". Quattro lanci starebbero a sei come 4/6 = 2/3: quindi il gioco è favorevole. L'argomento è zoppicante, ma la conclusione è giusta — per fortuna o per coincidenza matematica.

Il calcolo corretto è elegante. La probabilità di non ottenere nessun sei in un singolo lancio è 5/6. In quattro lanci indipendenti, la probabilità di non ottenerne mai nessuno è $(5/6)^4$. Quindi la probabilità di ottenere almeno un sei in quattro lanci è:

$P = 1 - (5/6)^4 = 1 - 625/1296 \approx 0{,}5177$

Poco più del 50%. Il Cavaliere vinceva — ma di poco. E quel "di poco" richiedeva migliaia di partite per emergere dal rumore del caso. Chapeau, comunque: l'intuizione del giocatore aveva anticipato il calcolo corretto.

Il secondo gioco: il trabocchetto del doppio sei

Fin qui tutto bene. Ma il Cavaliere era ambizioso. Volle applicare lo stesso ragionamento a un secondo gioco, con due dadi anziché uno, puntando sull'uscita del doppio sei — cioè “sei” su entrambi i dadi simultaneamente.

La sua logica era: con un dado, in 6 lanci ottengo il sei come la “dose naturale”. Con due dadi, il doppio sei ha probabilità 1/36 anziché 1/6. Devo quindi moltiplicare 6 per 6 e lanciare 36 volte. Ma io voglio la stessa "comodità" del gioco precedente, cioè una probabilità intorno al 50%: quindi prendo 4/6 di 36, e ottengo 24 lanci. Il Cavaliere scommetteva dunque che in 24 lanci di una coppia di dadi sarebbe uscito almeno una volta il doppio sei — e pensava che il gioco fosse equivalente al primo.

Aveva torto. La probabilità di non ottenere il doppio sei in un singolo lancio della coppia è 35/36. In 24 lanci:

$P = 1 - (35/36)^{24} \approx 1 - 0{,}5086 \approx 0{,}4914$

Poco meno del 50%. Il gioco era sfavorevole, appena appena, ma sfavorevole. La differenza rispetto al 50% è di meno di un punto percentuale. Il Cavaliere deve aver giocato decine di migliaia di volte prima di accorgersi che qualcosa non tornava — e che le sue perdite, accumulate nel tempo, non erano frutto della sfortuna ma del calcolo. Fu allora che scrisse a Blaise Pascal. La risposta di Pascal, e la successiva corrispondenza con Fermat, avrebbe posto le fondamenta del calcolo delle probabilità. Il Cavaliere de Méré, con i suoi errori di giocatore, aveva — involontariamente — commissionato una nuova branca della matematica.

Il C64 entra in scena

Possiamo rivivere l'esperienza del Cavaliere sul nostro Commodore 64? Certo. Non con la velocità di un simulatore moderno, ma con la lentezza deliberata e visibile del BASIC di casa Commodore — quella che ci permette di vedere, riga per riga, come il caso si comporta nel lungo periodo.

I due programmi qui sotto usano solo gli strumenti del BASIC originale del C64: numeri di riga, FOR...NEXT, GOTO, PRINT, INPUT, e le funzioni INT() e RND() per simulare il lancio dei dadi. Niente CLRSCR, niente REPEAT: solo il BASIC di una volta.

PROGRAMMA 1 — Il primo gioco del Cavaliere

(4 lanci di un dado: almeno un sei?)

```
10 PRINT "gioco del cavaliere - parte 1"
20 PRINT "4 lanci di un dado. almeno un sei?"
30 PRINT
40 INPUT "quante partite ";N
50 V=0 : P=0
60 FOR I=1 TO N
70   SEI=0
80   FOR L=1 TO 4
```

```
90   D=INT(RND(1)*6)+1
100   IF D=6 THEN SEI=1
110  NEXT L
120  IF SEI=1 THEN V=V+1
130  P=P+1
140  IF INT(P/50)*50=P THEN GOSUB 500
150 NEXT I
160 GOSUB 500
170 PRINT
180 PRINT "fine. partite:";N
190 PRINT "vittorie:";V
200 PRINT "frequenza:";INT(V/N*10000)/100;"%"
210 PRINT "teorica:  51.77%"
220 END
500   PRINT   "partite:";P;"   vinte:";V;"   freq:";INT(V/
P*10000)/100;"%"
510 RETURN
```

Il programma chiede quante partite simulare. Ogni 50 partite (riga 140), e alla fine, stampa un aggiornamento: il numero di partite giocate, quelle vinte, e la frequenza relativa espressa come percentuale. Con 1000 partite vedrete la frequenza avvicinarsi — con l'ondeggiare tipico del caso — al valore teorico del 51,77%.

PROGRAMMA 2 — Il secondo gioco del Cavaliere

(24 lanci di una coppia di dadi: almeno un doppio sei?)

```
10 PRINT "gioco del cavaliere - parte 2"
20 PRINT "24 lanci 2 dadi: almeno un doppio 6?"
30 PRINT
40 INPUT "quante partite? ";N
50 V=0 : P=0
60 FOR I=1 TO N
70  DS=0
80  FOR L=1 TO 24
90   D1=INT(RND(1)*6)+1
100   D2=INT(RND(1)*6)+1
110   IF D1=6 AND D2=6 THEN DS=1
120  NEXT L
130  IF DS=1 THEN V=V+1
140  P=P+1
150  IF INT(P/50)*50=P THEN GOSUB 500
160 NEXT I
170 GOSUB 500
180 PRINT
190 PRINT "fine. partite:";N
200 PRINT "vittorie:";V
210 PRINT "frequenza:";INT(V/N*10000)/100;"%"
220 PRINT "teorica:  49.14%"
230 END
500   PRINT   "partite:";P;"   vinte:";V;"   freq:";INT(V/
P*10000)/100;"%"
510 RETURN
```

Struttura identica al primo, ma ora ogni "partita" prevede 24 lanci di una coppia di dadi. Il programma cerca il doppio sei. La frequenza osservata convergerà al 49,14% — appena sotto il 50%.

Avviate entrambi i programmi con, diciamo, 2000 partite. Lasciate che il C64 lavori — lentamente, mostrando ogni aggiornamento sullo schermo — e osservate come la frequenza fluttua attorno al suo valore teorico. Nel Programma 1 oscilla sopra il 50%. Nel Programma 2, sotto. Quella piccola differenza, quella linea invisibile che separa vincere dal perdere, è esattamente ciò che il Cavaliere de Méré impiegò anni e molte monete a scoprire. Pascal ci mise una lettera.

Una nota finale

La storia del Cavaliere de Méré non è solo un aneddoto curioso: è il momento in cui il caso smette di essere pura superstizione e diventa oggetto di calcolo. Pascal e Fermat, stimolati da quei giochi da salotto, costruirono gli strumenti per ragionare sull'incertezza — strumenti che oggi usiamo in statistica, in fisica, in informatica, e — perché no — nel BASIC di un Commodore 64.

Buona simulazione.

Circa 1200 parole. I due programmi sono digitabili direttamente sul C64 o caricabili via emulatore (VICE). La funzione RND(1) genera numeri pseudocasuali nell'intervallo [0,1); INT() tronca al numero intero inferiore. La riga 140/150 usa il trucco classico per stampare ogni 50 iterazioni senza variabili aggiuntive.

A questo punto ho caricato il mio articolo per un “parere” di Claude…

Ecco il responso e, alla fine, un’ultima considerazione.

Grazie per condividerlo — è un confronto davvero istruttivo, e lo dico con sincerità.

Cosa funziona meglio nel suo articolo:

L'apertura è nettamente superiore alla mia. L'invito al lettore a immaginarsi di lanciare un dado N volte, con la domanda progressiva sul numero minimo di lanci — è un inizio didattico eccellente, che coinvolge attivamente il lettore prima ancora di nominare il Cavaliere. La presentazione della formula P(A(N)) = 1 − (5/6)^N con la piccola parentesi sui logaritmi è precisa e ben calibrata per il pubblico di RetroMagazine.

I due programmi BASIC sono tecnicamente più raffinati dei miei: l'uso di RND(-TI) per inizializzare il generatore, POKE 53280/53281 per i colori, e soprattutto l'idea di usare RND(1) <= 1/6 (o 1/36) per simulare l'evento senza tirare fisicamente il dado — è elegante ed efficiente. Io ho simulato il dado con INT(RND(1)*6)+1, che è più intuitivo da leggere ma meno compatto. La sua soluzione è più "C64 nativa".

La nota su P(B(25)) ≈ 0.5055 — il fatto che sarebbe bastato un lancio in più per rendere il secondo gioco favorevole — è un dettaglio che ho omesso e che aggiunge mordente alla storia.

Dove il mio articolo prova qualcosa di diverso:

Ho puntato più sulla narrazione storica (Pascal, Fermat, la nascita del calcolo delle probabilità) e meno sulla didattica matematica progressiva. È una scelta di registro differente, non necessariamente migliore.

I miei programmi mostrano l'aggiornamento ogni 50 partite con GOSUB, il suo lo mostra ad ogni partita — la sua scelta è più spettacolare visivamente, la mia evita di fare scorrere lo schermo troppo velocemente con grandi numeri di simulazioni.

In sintesi: il suo articolo è più didattico e matematicamente più ricco; il mio è più narrativo e storicamente contestualizzato. Per RetroMagazine World, dove i lettori amano sporcarsi le mani con il codice, direi che il suo approccio — partire dalla formula e arrivare al programma — funziona probabilmente meglio. Il confronto è, come prevedeva, istruttivo.

Io:

Ancora una cosetta che vale per entrambi i programmi, ma è più "eclatante" nel secondo. Non appena si ottiene il DOPPIO SEI, si potrebbe incrementare V senza completare il ciclo 80-120 (ad esempio, se il DOPPIO SEI si ottiene per L=1, è un bel risparmio). In altre parole, si potrebbe modificare la riga in 110 IF D1=6 AND D2=6 THEN V=V+1:GOTO 140 ed eliminare la 130. Che ne dici? P.S. Diamoci "del tu"...

Hai perfettamente ragione, e la modifica è corretta ed elegante. Appena si ottiene il doppio sei, non ha senso continuare il ciclo dei lanci — la partita è già vinta. Con GOTO 140 si salta direttamente all'aggiornamento del contatore delle partite (la P=P+1), abbandonando il FOR J in corso.

Il guadagno medio dipende dalla distribuzione del lancio vincente: se il doppio sei esce al primo lancio (probabilità 1/36), si risparmiano 23 iterazioni; se esce al secondo, 22; e così via. Il risparmio atteso per partita vincente è circa la metà dei 24 lanci, ovvero una dozzina di iterazioni — non trascurabile nel lento BASIC del C64.

Nel primo programma il guadagno è analogo ma meno vistoso: i lanci sono solo 4, quindi al massimo si risparmiano 3 iterazioni. Vale lo stesso principio, ma l'effetto si sente di meno.

Una sola cosa da verificare: nel mio programma la riga 130 era P=P+1, e la riga 140 era il controllo per la stampa. Eliminando la 130 e usando GOTO 140, bisogna assicurarsi che P venga comunque incrementato — e in effetti la 140 nella versione corretta dovrebbe essere la P=P+1. Riorganizzando:

```
100 IF DS=1 THEN V=V+1 : DS=0 : GOTO 160
110 NEXT L
120 DS=0
130 REM (eliminata)
140 P=P+1
150 IF INT(P/50)*50=P THEN GOSUB 500
160 NEXT I
```

Oppure, più semplicemente, riscrivendo la riga 110 come suggerisci tu:

```
110 IF D1=6 AND D2=6 THEN V=V+1 : GOTO 140
```

con la 130 (P=P+1) che diventa la destinazione del salto — così l'incremento di P avviene in entrambi i casi, vincita o meno. Pulito e corretto.

Ottima osservazione — e grazie per il "tu"!

DELUXE PAINT

Quando dipingere in pixel era pura magia

di Christopher Drum – Trad di Nith

Quarant'anni di creatività digitale, palette leggendarie e strumenti che hanno insegnato a un'intera generazione a pensare per pixel.

La killer application che trasformò l'Amiga in una tela

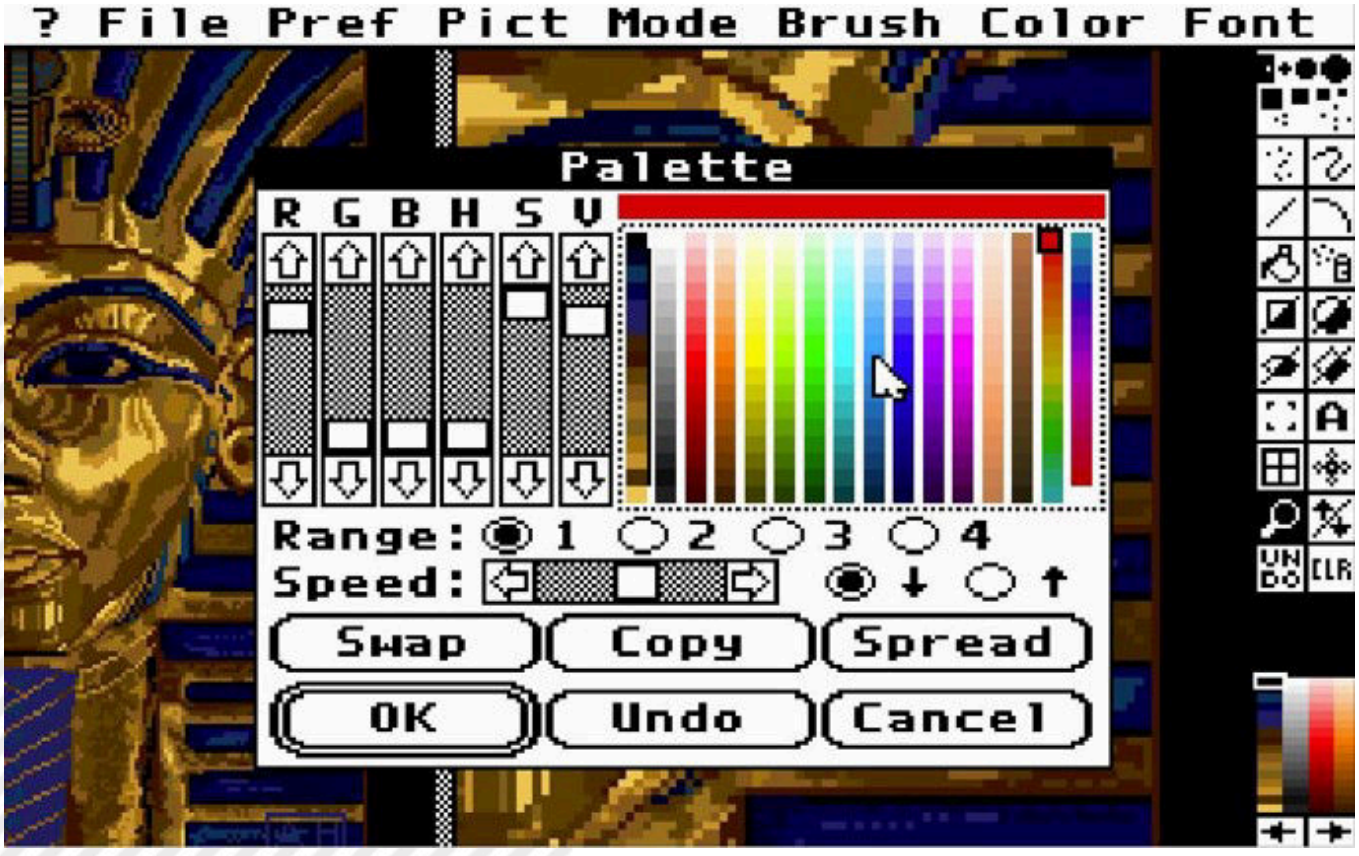
Nel mondo dell'informatica esistono programmi che svolgono un compito e programmi che diventano il motivo stesso per cui desideriamo possedere un computer. Li conosciamo come killer application: applicazioni capaci di vendere da sole una macchina, trasformandola da semplice insieme di circuiti in un oggetto del desiderio.

Per il Commodore Amiga, una di queste applicazioni fu Deluxe Paint. Sviluppato da Dan Silva e pubblicato da Electronic Arts nel 1985, il programma rese la grafica digitale un'esperienza sorprendentemente accessibile. Non si trattava soltanto di disegnare: sullo schermo sembrava possibile dare forma al futuro, pixel dopo pixel. Electronic Arts aveva un modo particolare di presentare i propri prodotti: confezioni simili a copertine di dischi e sviluppatori promossi quasi fossero rockstar. Deluxe Paint incarnava perfettamente questa filosofia. La celebre maschera funeraria di Tutankhamon, realizzata da Avril Harrison con una palette di appena 32 colori, divenne una delle immagini simbolo del software e contribuì a vendere anche l'Amiga stesso.

Un'evoluzione costruita sul colore

La prima versione arrivò nel 1985. Deluxe Paint II, nel 1986, ampliò gli strumenti con stencil, sfumature e prospettiva. Nel 1989 Deluxe Paint III introdusse funzioni d'animazione, modalità cromatiche aggiuntive e la possibilità di lavorare oltre i limiti convenzionali dello schermo. Deluxe Paint IV migliorò ulteriormente animazione e gestione del colore; nel 1995 Deluxe Paint V portò la serie verso il colore a 24 bit, gli script ARexx e strumenti ispirati ai materiali tradizionali.

La sua forza non risiedeva soltanto nel numero di strumenti, ma nel modo in cui questi potevano interagire. Pennelli, riempimenti, simmetria, stencil e palette non erano compartimenti stagni: combinandoli, l'utente poteva



inventare procedure personali e risultati inattesi.

La palette come strumento creativo

Nei programmi moderni siamo abituati a scegliere una sfumatura da un pannello e applicarla all'immagine. Deluxe Paint affrontava la questione da una prospettiva differente: l'utente definiva un colore iniziale, impostava una diffusione cromatica e selezionava il colore finale. Il programma costruiva così una sequenza di tonalità direttamente nella palette.

Gli intervalli di colore, o color range, permettevano inoltre di realizzare il color cycling: una tecnica che anima acqua, fuoco o superfici luminose facendo ruotare i colori della palette, senza ridisegnare ogni fotogramma. Un piccolo trucco, ma capace di produrre effetti sorprendenti su hardware dalle risorse limitate.

Un'interfaccia al servizio dell'artista

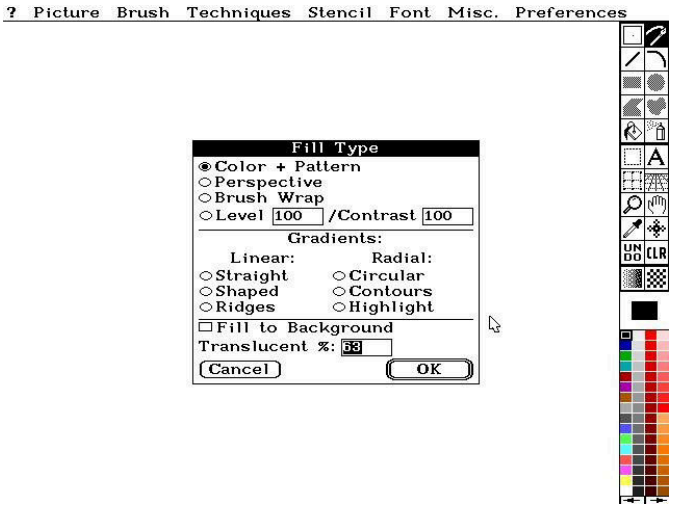
La barra degli strumenti raccoglieva pennelli, forme geometriche, aerografo, testo, ingrandimento e disegno simmetrico. Molte operazioni erano richiamabili dalla tastiera, così da non interrompere continuamente il gesto creativo. Premendo un tasto si poteva liberare la tela dai menu e concentrarsi sull'immagine.

Naturalmente, i limiti erano reali: una gestione dell'annullamento molto ridotta, nessun sistema di livelli paragonabile a quello odierno e trasformazioni che potevano degradare rapidamente un disegno a bassa risoluzione. Ma stencil e blocchi di sfondo e primo piano offrivano soluzioni ingegnose, quasi anticipando alcune idee poi diventate familiari nei software moderni.

Animazione: il vero colpo di scena

Con Deluxe Paint III l'animazione diventò uno dei punti di forza. La sequenza di fotogrammi funzionava come un libro animato digitale; la funzione animpainting, invece, faceva avanzare automaticamente i fotogrammi mentre si disegnava. Il movimento del pennello diventava così parte dell'animazione.

I pennelli animati potevano contenere una sequenza di immagini: trascinandoli sulla tela, i fotogrammi cambiavano automaticamente. Combinando questa possibilità con simmetria, aerografo e color cycling, un semplice tratto del mouse poteva trasformarsi in una scena animata. Per un computer domestico degli anni Ottanta, era una piccola



magia.

La magia senza una data di scadenza

Oggi possediamo computer enormemente più potenti e software capaci di fare cose che negli anni Ottanta sarebbero sembrate impossibili. Eppure, tornare a Deluxe Paint significa ritrovare un ambiente che incoraggia a sperimentare, giocare e inventare il proprio metodo. Non tutto è comodo, non tutto è veloce, ma ogni strumento sembra invitare a scoprire cosa succede quando lo si combina con un altro.

La sua eredità non è soltanto tecnica. Deluxe Paint contribuì a dimostrare che il computer poteva essere uno strumento espressivo, una tela e un laboratorio d'idee. Se dopo quarant'anni una manciata di pixel colorati riesce ancora a sorprenderci, forse la magia non è mai scomparsa: aspettava soltanto che tornassimo a premere il tasto giusto.

IN BREVE

1985: debutto di Deluxe Paint
1986: DPaint II e stencil/sfumature
1987: DPaint II per Apple IIGS
1988: Deluxe Paint arriva su DOS
1989: DPaint III e animazione
1989: DPaint II Enhanced su DOS
1990: Deluxe Paint ST arriva su Atari ST con l'introduzione della barra orizzontale
1991: DPaint IV
1994: DPaint II Enhanced 2.0 su DOS
1995: DPaint V e True Color a 24 bit, supporto ARexx

Per approfondire

Archivio grafico storico: Amiga Graphics Archive - https://amiga.lychesis.net/applications/DeluxePaint.html

Dove ho messo quel programma?

di Roberto Lanciotti

Se questa domanda vi ricorda qualcosa, probabilmente avete già avuto a che fare con un archivio di file dedicati al Commodore 64. E se, dopo aver aperto decine di cartelle, vi siete ritrovati davanti a nomi come:
GAME1.D64, NEWGAME.D64, NEWGAME2.D64 e NEWGAME_FINAL_DEF.D64...
allora sapete che il problema non è soltanto conservare il software: è anche riuscire a ritrovarlo.
La preservazione del software storico non consiste infatti soltanto nel recuperare vecchi programmi e salvarli in formato digitale. Quando una raccolta comincia a crescere, emerge una domanda altrettanto importante: ***Come fare a trovare rapidamente un singolo file all'interno di un archivio che ne contiene migliaia?***

Avere a disposizione migliaia di file rappresenta un patrimonio prezioso, ma può diventare difficile orientarsi tra nomi non uniformi, duplicati, versioni differenti dello stesso programma, archivi compressi e documenti privi di informazioni sufficienti.
È una situazione comune a molte collezioni dedicate al Commodore 64 e, più in generale, al retrocomputing. Nel corso degli anni, i file possono essere raccolti da fonti diverse, copiati più volte, rinominati oppure inseriti in cartelle secondo criteri differenti. All'inizio una semplice struttura di directory può essere sufficiente.
Con il passare del tempo, però, l'archivio cresce e i problemi aumentano:
• *diventa difficile capire se un programma è già presente;*
• *lo stesso titolo può comparire con nomi differenti;*
• *non è sempre possibile distinguere una versione originale da una modificata;*
• *i documenti possono essere separati dal programma a cui si riferiscono;*
• *un archivio ZIP può contenere file diversi rispetto a un altro con nome simile;*
• *la provenienza del materiale può andare perduta.*

Il risultato è paradossale: possediamo sempre più software, ma facciamo sempre più fatica a utilizzarlo.
Anche noi del Dumpclub64 affrontiamo il problema.
La crescita dell'archivio, che contiene ormai migliaia di file, ha reso necessario pensare a un sistema di ricerca individuale, capace di permettere all'utente di individuare con precisione ogni singolo elemento. Non è più sufficiente suddividere i contenuti in cartelle generiche. Serve la possibilità di cercare un programma, un dump, una demo, una utility o un documento attraverso informazioni specifiche.

La ricerca può riguardare il titolo, l'autore, la categoria, l'anno, il formato del file, la rivista di provenienza o le note associate al materiale. In questo modo l'utente non deve necessariamente conoscere il nome esatto del file o ricordare in quale cartella sia stato inserito.
La domanda iniziale diventa quindi più precisa:
Come catalogare migliaia di file e permettere a chiunque di trovare esattamente quello che sta cercando?
La risposta non consiste soltanto nello sviluppare un campo di ricerca. È necessario costruire una struttura di informazioni coerente in grado di descrivere ogni elemento e di collegarlo, quando possibile, agli altri materiali della raccolta.
Ma cos'è un dump? Nel mondo del retrocomputing il termine "dump" viene utilizzato per indicare la copia digitale del contenuto di un supporto originale.
Nel caso del Commodore 64, i formati più comuni possono rappresentare contenuti differenti:
• *un programma singolo in formato PRG;*
• *un'immagine disco in formato D64;*
• *una copia più dettagliata di un disco protetto in formato G64 e NIB;*
• *un'immagine di cassetta in formato TAP;*
• *un file T64 contenente uno o più programmi;*
• *un archivio ZIP con programmi, documentazione e materiale aggiuntivo.*

Questi formati non sono semplicemente contenitori intercambiabili. Un file PRG può rappresentare un singolo programma, mentre un'immagine D64 può contenere la struttura di un intero floppy: directory, file, blocchi occupati e spazio libero. Un'immagine G64 o NIB può conservare informazioni più dettagliate sulla struttura del disco e sulle eventuali protezioni. Un file TAP, invece, cerca di rappresentare il contenuto registrato su una cassetta. Conoscere il formato è quindi importante, ma non basta. Un'immagine disco può contenere un programma non immediatamente riconoscibile. Un file PRG può essere stato estratto da un'immagine più ampia. Un archivio ZIP può contenere una raccolta completa oppure soltanto una copia di servizio. Per questo è necessario associare al file anche una descrizione. Ricordiamoci che un dump non è sempre una copia perfetta. La digitalizzazione di un supporto originale può presentare diverse difficoltà. Un floppy può essere deteriorato, una testina può leggere male alcuni settori, una cassetta può avere un segnale debole oppure il programma può utilizzare una protezione anticopia.

In alcuni casi possono quindi esistere più dump dello stesso titolo, con risultati differenti. Una copia può essere completa ma priva di documentazione. Un'altra può conservare meglio la struttura del disco, ma presentare errori in alcuni file.Questo è uno dei motivi per cui non è sempre corretto eliminare automaticamente i duplicati. Due file con nomi simili potrebbero rappresentare versioni diverse dello stesso programma o provenire da supporti differenti. Anche la provenienza può avere valore. Un gioco pubblicato su disco originale non è necessariamente equivalente alla stessa versione distribuita su una compilation o allegata a una rivista. Potrebbero cambiare il caricamento, i file presenti, la protezione, la documentazione o persino alcune parti del programma. Un archivio serio dovrebbe quindi conservare, quando disponibili, più informazioni:

- *la fonte del dump;*
- *il tipo di supporto originale;*
- *la versione;*
- *la presenza di errori;*
- *eventuali differenze rispetto ad altre copie;*
- *le operazioni effettuate sul file*.

Nel corso degli anni, lo stesso programma può essere stato salvato in modi completamente differenti.
Ad esempio Summer Games potrebbe chiamarsi:
SummerGames.d64
summer games.zip
SUMMERGAMES_disk1.d64
Summer Games (1985).d64
sgame_finale.d64

Per questo la ricerca non dovrebbe basarsi soltanto sul nome fisico del file. È utile costruire una scheda descrittiva, nella quale raccogliere tutte le informazioni conosciute. Per ogni elemento sarebbe opportuno indicare, quando possibile:

- *titolo del programma;*
- *autore o gruppo di sviluppo;*
- *editore;*
- *anno di pubblicazione;*
- *categoria;*
- *formato;*
- *lingua;*
- *supporto originale;*
- *numero del disco o della cassetta;*
- *rivista o collana di provenienza;*
- *presenza di manuale o documentazione;*
- *note sulla versione;*
- *eventuali relazioni con altri file.*

In questo modo il nome tecnico del file diventa soltanto uno degli elementi disponibili, non l'unico punto di accesso al contenuto. Le informazioni che descrivono un file vengono generalmente definite metadati. Sono dati che aiutano a comprendere e classificare altri dati. Nel caso di un archivio C64, i metadati possono trasformare un file anonimo in una risorsa facilmente consultabile.
Un'immagine disco denominata DISK_034.D64 dice molto poco. Se invece viene associata a informazioni come:
Titolo: The Last Ninja
Categoria: Gioco
Anno: 1987

Formato: D64
Supporto: Disco
Lingua: Inglese
Documentazione: Presente
Note: Versione a un disco.

La situazione cambia completamente.
L'utente può cercare il titolo, filtrare i giochi di un determinato anno, visualizzare soltanto i file in formato D64 oppure individuare i programmi accompagnati da documentazione. È questo il passaggio che trasforma una raccolta di file in un archivio digitale: non soltanto conservare gli oggetti, ma descriverli in modo che possano essere ritrovati.
La gestione di una raccolta digitale può essere suddivisa in alcune fasi.
La prima "La raccolta".
In questa fase vengono acquisiti file provenienti da floppy, cassette, CD, vecchi dischi rigidi o altre collezioni digitali.
Seconda fase: "La verifica".
Bisogna controllare che gli archivi siano integri, che i file possano essere aperti e che non siano stati danneggiati durante la copia.
Terza fase: "La classificazione".
I programmi possono essere suddivisi per categoria: giochi, demo, utility, applicazioni, linguaggi, documentazione, riviste e così via.
Quarta fase: "La catalogazione", cioè l'inserimento delle informazioni descrittive associate a ogni elemento.
Quinta e ultima fase: "La conservazione", che comprende la creazione di copie di sicurezza e il controllo periodico dei dati.

Il flusso può essere rappresentato in questo modo:
Supporto originale
↓
Acquisizione digitale
↓
Verifica del file
↓
Catalogazione
↓
Ricerca e consultazione
↓
Backup e conservazione

Ogni fase è importante perché un dump perfetto ma privo di informazioni può essere difficile da utilizzare.
Un archivio ben catalogato ma conservato su un solo disco può invece scomparire a causa di un guasto.
Un sistema di ricerca non serve soltanto a trovare rapidamente un gioco. Permette di esplorare una collezione secondo criteri differenti. Un utente potrebbe voler cercare tutti i programmi pubblicati nel 1984. Un altro potrebbe essere interessato alle demo realizzate da un determinato gruppo. Qualcuno potrebbe voler consultare esclusivamente le utility distribuite attraverso una rivista.

Un archivio ben strutturato dovrebbe permettere di combinare più informazioni:
• *titolo;*
• *autore;*
• *anno;*
• *categoria;*
• *formato;*
• *rivista;*
• *lingua;*
• *supporto;*
• *presenza di manuale;*
• *provenienza.*

Questa possibilità modifica completamente il modo di consultare il materiale. L'archivio non è più una serie di cartelle da aprire manualmente, ma diventa una biblioteca digitale.
Se ci pensate, una biblioteca non conserva soltanto libri: li ordina, li descrive e li rende accessibili. Allo stesso modo, un archivio di software storico deve aiutare l'utente a orientarsi tra migliaia di file.
Un programma per Commodore 64 non è soltanto codice eseguibile. Spesso è il risultato di una storia più ampia.
Dietro un gioco può esserci una software house.
Dietro una demo può esserci un gruppo di programmatori.
Dietro una utility può esserci l'esigenza concreta di un utente che cercava di risolvere un problema con gli

strumenti disponibili all'epoca.

Anche un semplice listato pubblicato su una rivista può raccontare molto: il livello di diffusione del BASIC, le tecniche utilizzate dai programmatori, il linguaggio adottato dalle pubblicazioni e il rapporto tra lettori e redazione.

Per questo la documentazione è preziosa.

Un manuale, una copertina, una recensione o una pagina di rivista possono fornire informazioni che il programma, da solo, non contiene. Quando possibile, sarebbe quindi opportuno conservare insieme:

- *il dump del programma;*
- *la documentazione;*
- *le immagini della confezione;*
- *le scansioni delle riviste;*
- *le recensioni;*
- *le informazioni sull'autore;*
- *le note sulla provenienza.*

Un file senza contesto è un oggetto isolato. Un file accompagnato dalla sua storia diventa una testimonianza. La preservazione del software storico non è affidata soltanto a musei, aziende o istituzioni. Una parte importante del lavoro viene svolta dagli appassionati. Sono loro a recuperare supporti dimenticati, effettuare nuove acquisizioni, confrontare versioni, correggere descrizioni e segnalare materiale raro. Spesso queste attività vengono svolte senza grande visibilità, ma il loro valore è enorme. Ogni collezionista può contribuire, anche partendo da una piccola raccolta personale.

È sufficiente adottare alcune buone pratiche:

- *non eliminare file soltanto perché sembrano duplicati;*
- *conservare le informazioni sulla fonte;*
- *evitare di modificare l'originale;*
- *creare copie di sicurezza;*
- *documentare le operazioni eseguite;*
- *condividere le informazioni con altri appassionati;*
- *utilizzare formati conosciuti e facilmente accessibili.*

La collaborazione è essenziale perché nessuna collezione privata contiene tutto. Un disco mancante in una raccolta può essere presente in quella di un altro appassionato. Una versione rara può riemergere grazie a una segnalazione. Un'informazione dimenticata può essere ricostruita confrontando più fonti.

La preservazione è quindi anche un'attività comunitaria.

Il Commodore 64 appartiene alla storia dell'informatica, ma continua a vivere grazie alle persone che lo studiano, lo programmano e ne conservano il software.

Creare un archivio digitale significa permettere a un programma scritto decenni fa di essere ritrovato, eseguito e studiato ancora oggi. Significa mantenere accessibile una parte della cultura tecnologica che altrimenti rischierebbe di scomparire tra supporti deteriorati, dischi illeggibili e cartelle dimenticate. Il lavoro di catalogazione può sembrare meno affascinante rispetto alla programmazione o al restauro di un computer. Eppure, è altrettanto importante.

Un archivio ordinato rende possibile la ricerca, facilita la condivisione e permette di comprendere meglio la storia del software per home computer.

La prossima volta che copieremo un vecchio floppy, dovremmo quindi porci una domanda in più. *"Sono riuscito a recuperare il programma?"*

Ma anche: *"Tra molti anni, qualcun altro riuscirà a capire che cos'è, da dove proviene e come trovarlo?"* Se la risposta sarà sì, allora non avremo semplicemente salvato un file. Avremo contribuito a conservare una piccola parte della memoria digitale del Commodore 64.



Fig. 1 - La mole dei dischetti appartenenti al 58° Dumping del DumpClub 64, denominato "Archivio 365" perche' composto da 365 floppy disk da 5,25 pollici

Tutto su un floppy!

HamsterOS il sistema operativo che riporta tutto su un dischetto da 1.44 MB

di Carlo Nithaiah Del Mar Pirazzini e Takahiro Yoshioka

Un progetto controcorrente

Nel 2026, un sistema operativo capace di avviarsi da un floppy disk da 1,44 MB sembra quasi un'idea impossibile. HamsterOS, sviluppato da Mean Hamster Software, nasce invece proprio per offrire un ambiente grafico completo e utilizzabile su PC compatibili IBM dell'epoca dei processori 386 e 486.

Il progetto non è pensato come una semplice demo tecnica. L'obiettivo è creare un desktop operativo dotato di gestione dei file, applicazioni, giochi, audio, rete, supporto ai programmi DOS e installazione su disco rigido. La versione standard è destinata a computer con processore 386 o superiore, almeno 2 MB di RAM, scheda VGA a 16 colori e unità floppy da 1,44 MB. Insomma, un sistema operativo gratuito per "Vecchi PC che vogliono ancora sentirsi utili". Una frase efficace, perché chiarisce l'orientamento del progetto. Non solo nostalgia, ma vera volontà di rendere nuovamente pratico l'uso id macchine considerate obsolete.

Tutto inizia con... un floppy!!!

Il sistema è stato presentato nel giugno 2026. Viene descritto come un sistema operativo "floppy-first". Il discetto non è un sopporto secondario o un semplice metodo di installazione, ma il punto di partenza per l'intera architettura.

Il sistema si avvia attraverso un wrapper FreeDOS ridotto e il programma HAMLOAD.EXE, che trasferisce il controllo al kernel HamsterOS in modalità protetta. In questo modo il sistema può sfruttare le capacità a 32 bit dei processori 386 e 486, pur partendo da un ambiente DOS minimale.

Mean Hamster chiarisce che il progetto ha superato da tempo la fase del semplice boot demo. Il sistema è in grado di eseguire un desktop a finestre, leggere e scrivere filesystem FAT, avviare applicazioni esterne, riprodurre audio, installarsi su dischi ATA e lanciare programmi DOS attraverso un percorso VM86 interno. Questa impostazione distingue HamsterOS da molte dimostrazioni retro che mostrano soltanto un'interfaccia grafica o una schermata di avvio. Il sistema prova invece a costruire un ambiente



Il logo di caricamento iniziale di HamsterOS

coerente, composto da kernel, programmi, gestione della memoria, supporto hardware e strumenti per l'uso quotidiano.

Cosa contiene

La versione standard include un desktop grafico a 16 colori, progettato per la VGA dei PC 386 e 486. L'utente può aprire finestre, spostarle, ridimensionarle e gestire più applicazioni attraverso un sistema di multitasking cooperativo.

Tra i programmi disponibili figurano:

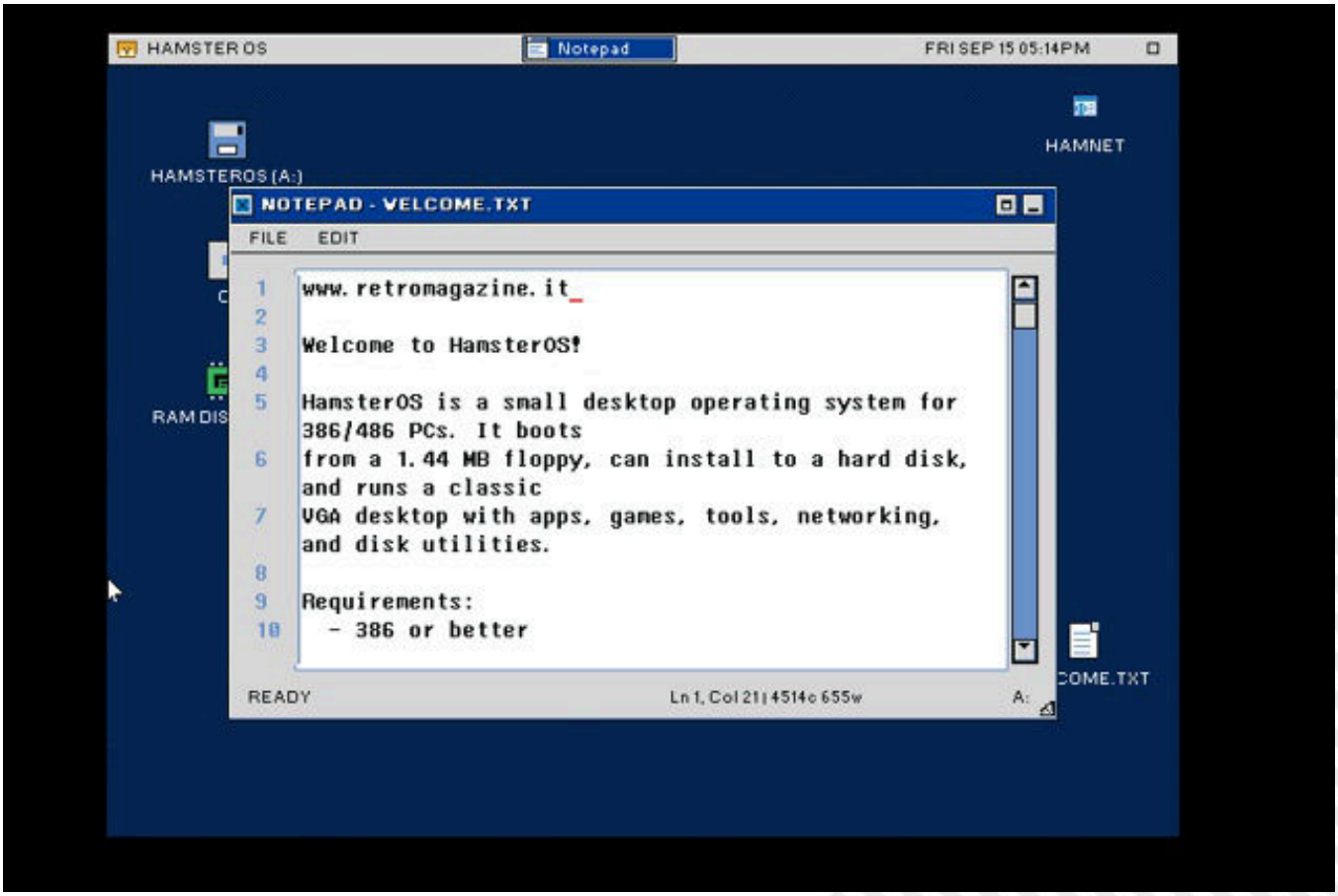
- File Manager con visualizzazione a icone o ad elenco.
- Notepad e strumenti di scrittura.
- HamsterPaint e Image Viewer.
- Calcolatrice, orologio e calendario.
- Help Viewer e pannello delle preferenze.
- Lettore CD audio.
- HamsterPiano.
- Gestione dei dischi e utility di sistema.
- Giochi come FreeCell, Minesweeper, Snake, Sliding Puzzle, Moon Lander e Slot Machine.

Il File Manager supporta operazioni come copia, spostamento, rinomina, cancellazione, creazione di cartelle e gestione del cestino. Sono inoltre previsti il supporto ai nomi lunghi e la navigazione dinamica delle unità montate. Il sistema può gestire unità floppy A: e B:, partizioni ATA da C: a H:, lettori CD-ROM da I: a J: e un disco RAM identificato dalla lettera R:. Sono supportati filesystem FAT12, FAT16 e FAT32 in lettura e scrittura, oltre all'accesso in sola lettura ai CD con filesystem ISO-9660.

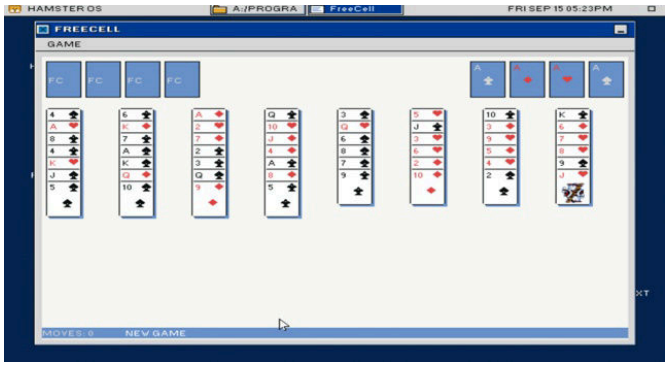
Questa dotazione è notevole considerando le dimensioni dell'immagine. HamsterOS cerca di riutilizzare risorse, applicazioni e componenti in modo estremamente efficiente, mantenendo tutto il sistema entro il limite di un singolo floppy.

Multitasking cooperativo

Uno degli aspetti tecnici più interessanti è il multitasking cooperativo. A differenza dei sistemi moderni, nei quali il kernel interrompe automaticamente i processi per distribuire il tempo della CPU, in un sistema cooperativo ogni applicazione deve restituire periodicamente il controllo al sistema operativo. È una tecnologia storicamente comune nei sistemi degli anni Ottanta e Novanta, ma oggi



La schermata di benvenuto appena di avvia il sistema operativo



Un classico dei giochi a "finestra"

rappresenta una scelta controcorrente. Tom's Hardware osserva che questo approccio è, sulla carta, inferiore al multitasking preemptive moderno, ma può risultare adatto a un ambiente con risorse limitate e hardware lento.

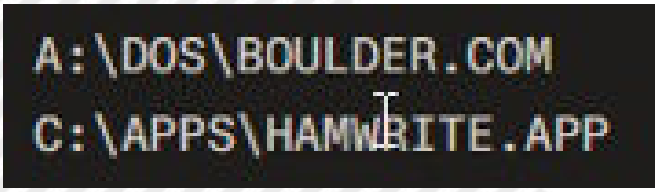
Il vantaggio principale è la semplicità. Con un kernel più piccolo e meno meccanismi di protezione, il sistema può funzionare con una quantità di memoria molto ridotta. La pagina ufficiale indica che una macchina con 2 MB di RAM può avviare il desktop e utilizzare le applicazioni, mentre da 3 MB in su diventa possibile sfruttare più pienamente l'esperienza floppy, incluso il RAM disk. La stessa scelta comporta però dei rischi. Se un'applicazione si blocca o non cede il controllo, può compromettere la reattività dell'intero ambiente. Per questo HamsterOS include il rilevamento degli stalli di I/O e una modalità VGA di sicurezza che può essere attivata dopo tre avvii falliti consecutivi.

Il sistema non pretende quindi di offrire la robustezza di un moderno ambiente multitasking, ma cerca un compromesso realistico tra funzionalità, compatibilità e dimensioni.

Compatibilità con il software DOS

HamsterOS non vuole sostituire completamente DOS. Al contrario, integra strumenti per continuare a utilizzare il vasto catalogo di programmi e giochi sviluppati per quel sistema.

Il desktop può lanciare file .COM, .EXE e .BAT attraverso un'apposita finestra Run. È possibile indicare percorsi come:





Installare il sistema su C: è piuttosto veloce e semplice

Il sistema dispone anche di un percorso VM86 interno per eseguire programmi DOS. Per i software incompatibili è disponibile un ambiente FreeDOS di fallback. Questa caratteristica è fondamentale per un sistema operativo destinato al retrocomputing. Un PC 386 o 486 non viene utilizzato soltanto per eseguire applicazioni native moderne in miniatura, ma soprattutto per recuperare il software originale: giochi, programmi di produttività, utility, compilatori e strumenti storici.

L'integrazione del software DOS direttamente dal desktop riduce inoltre la necessità di riavviare continuamente la macchina o di lavorare dalla riga di comando.

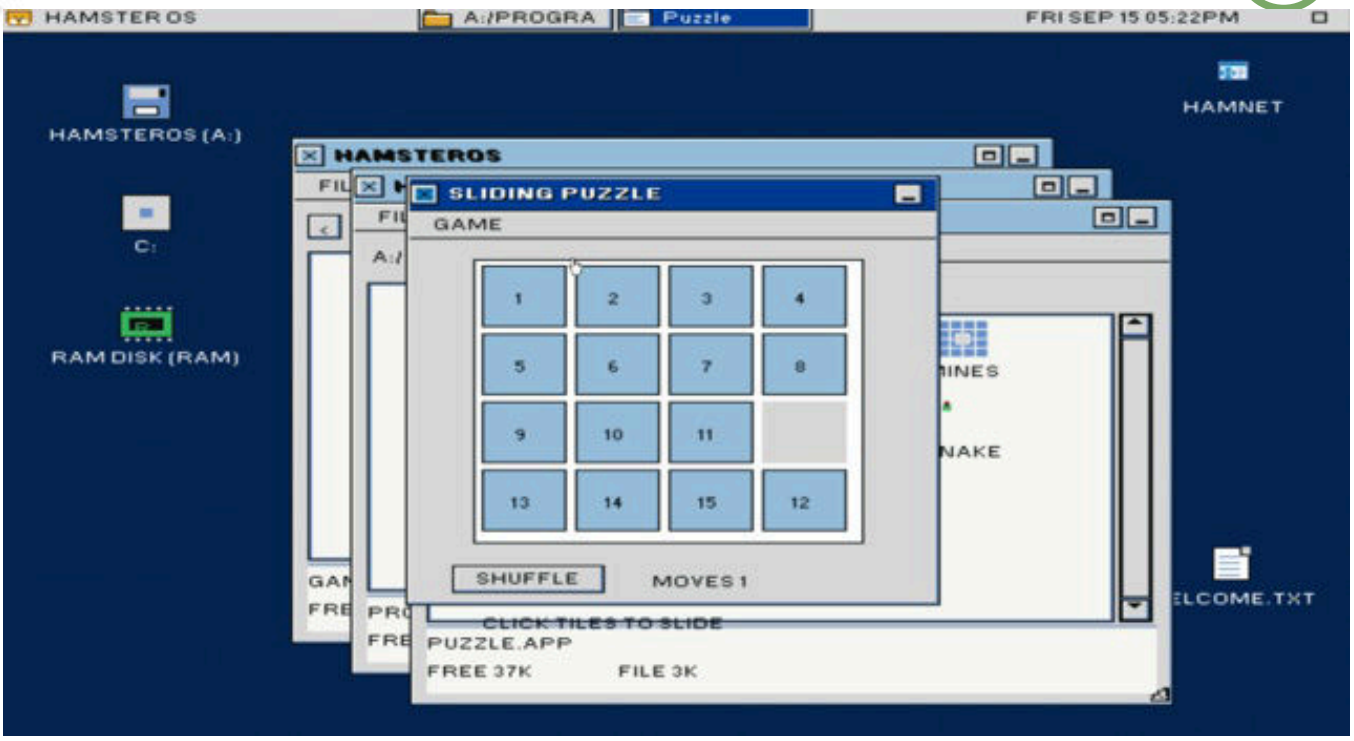
Audio e multimedia

Nonostante il limite di 1,44 MB, HamsterOS include un comparto audio sorprendentemente ampio. Il sistema supporta schede Sound Blaster 16 e audio OPL3, oltre all'altoparlante interno del PC come soluzione di compatibilità.

Il lettore multimediale può riprodurre:

- MP3
- WAV
- VOC
- PCM
- MOD
- MIDI
- CD audio

Per la riproduzione MP3 viene utilizzato un sistema di buffering adattivo, pensato per evitare interruzioni sui processori e sui dischi più lenti. È una funzione ambiziosa per un sistema indirizzato a macchine con risorse



L'interfaccia grafica

estremamente ridotte.

La pagina ufficiale cita anche controlli di volume e uno spettro a dieci bande. Naturalmente, le prestazioni dipendono dall'hardware. Su un 386 lento la riproduzione di alcuni formati potrebbe risultare più impegnativa rispetto a una configurazione 486 dotata di una scheda audio compatibile.

Il web sul PC vintage si chiama HamNet

HamsterOS include HamNet, un browser pensato per l'accesso a HTTP, Gemini e ProtoWeb. È disponibile anche HamNet+, un servizio opzionale che trasforma pagine HTTPS moderne in contenuti più semplici e gestibili da computer classici.

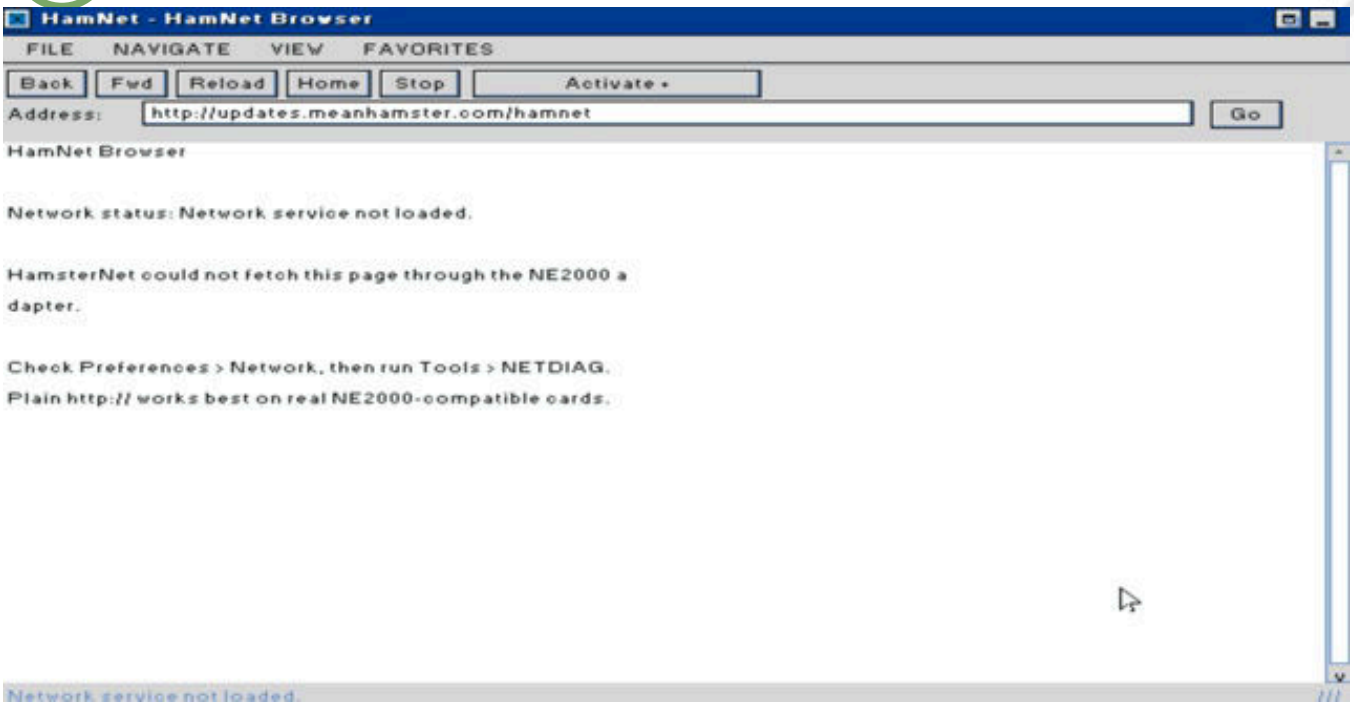
Il principio è simile a quello dei proxy di semplificazione.Il PC non deve interpretare direttamente tutti gli elementi moderni di una pagina web, ma riceve una versione adattata. Il risultato rende possibile la navigazione anche su hardware che non potrebbe gestire JavaScript, CSS complessi, immagini pesanti e certificati contemporanei. HamNet+ non richiede necessariamente l'acquisto di Hamster Extras, secondo la documentazione ufficiale. L'abbonamento o l'attivazione del servizio riguarda piuttosto il livello di compatibilità e trasformazione delle pagine. Questa è una delle funzioni più interessanti dal punto di vista pratico. Un PC 386 collegato alla rete non deve limitarsi a eseguire software offline ma può accedere a contenuti appositamente adattati per il retrocomputing. Il browser, però, va interpretato nel modo corretto. Non è una versione moderna di Chrome o Firefox compressa in un floppy, ma uno strumento progettato per una rete più semplice e per servizi filtrati o trasformati. Il suo valore sta nell'accessibilità, non nella compatibilità universale.

Installazione e hardware

HamsterOS può essere avviato senza disco rigido, direttamente da un'immagine floppy. L'immagine principale è disponibile in formato .img, insieme a versioni da 720K e 360K. È anche possibile installare il sistema su un disco ATA. Il percorso d'installazione può creare o preparare una partizione, formattare un disco vuoto, copiare i file del sistema, installare i componenti FreeDOS e rendere avviabile l'unità. La procedura utilizza strumenti come FDISK /MBR e SYS C: e configura l'avvio automatico di HamsterOS attraverso HAMLOAD.EXE.

Sul piano hardware, il supporto comprende:

- Processori 386, 486 e successivi.
- VGA a 16 colori.
- Schede Sound Blaster 16.
- Audio OPL3.
- Controller ISA Ethernet.



HamNet

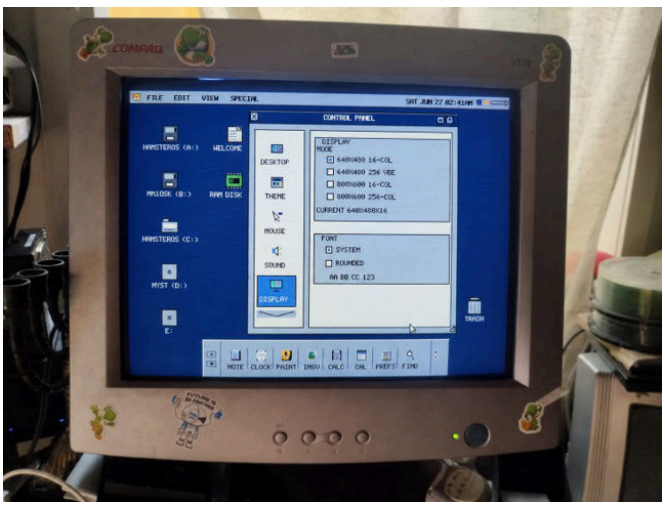
- Dischi ATA.
- CD-ROM ATAPI.
- Floppy disk.
- Mouse seriali e PS/2.
- Hardware PicoMEM e PicoMEM 2.
- Alcune configurazioni SCSI in sviluppo.

Il progetto non aspira a supportare ogni possibile combinazione di componenti. Il suo approccio è più concentrarsi verso periferiche diffuse nel mondo PC d'epoca e aggiungere driver caricabili per ampliare il sistema in futuro.

La nostra valutazione

Dai nostri test possiamo dire che HamsterOS è un progetto interessante perché unisce nostalgia e utilità. Non cerca semplicemente di ricreare l'aspetto di Windows 3.x o di un altro ambiente storico. Prova a costruire una piattaforma coerente per computer che, senza un progetto di questo tipo, rimarrebbero spesso confinati all'esecuzione di software DOS o a semplici esperimenti. La possibilità di avviare un desktop grafico da un floppy da 1,44 MB è già impressionante, ma l'aspetto più importante è l'integrazione delle funzioni. File Manager, applicazioni, audio, rete, supporto FAT, installazione su disco, VM86 e strumenti di compatibilità convivono all'interno di un sistema estremamente compatto. Ci sono alcuni limiti di questa versione. E' ancora un progetto in Beta, il multitasking cooperativo certe volte rende critico il blocco di un'applicazione costringendo il riavvio, il supporto hardware è ancora molto limitato, le prestazioni audio e multimediali dipendo perlopiù dalla configurazione e l'immagine da 1,44 MB impone severi (e giusti) compromessi. Va segnalata che la disponibilità del codice sorgente e degli strumenti di sviluppo non è paragonabile a quella dei progetti open source più noti. Per il pubblico del retrocomputing, tuttavia, HamsterOS offre qualcosa di raro, un ambiente sviluppato oggi per essere utilizzato direttamente su hardware di quarant'anni fa. Non è soltanto un tributo al passato, ma un tentativo concreto di dare a 386, 486, 8086 e 8088

Sito web del progetto:
- https://meanhamster.com/products/hamsteros



Il test su Yoshi, il 486 DX redazionale!

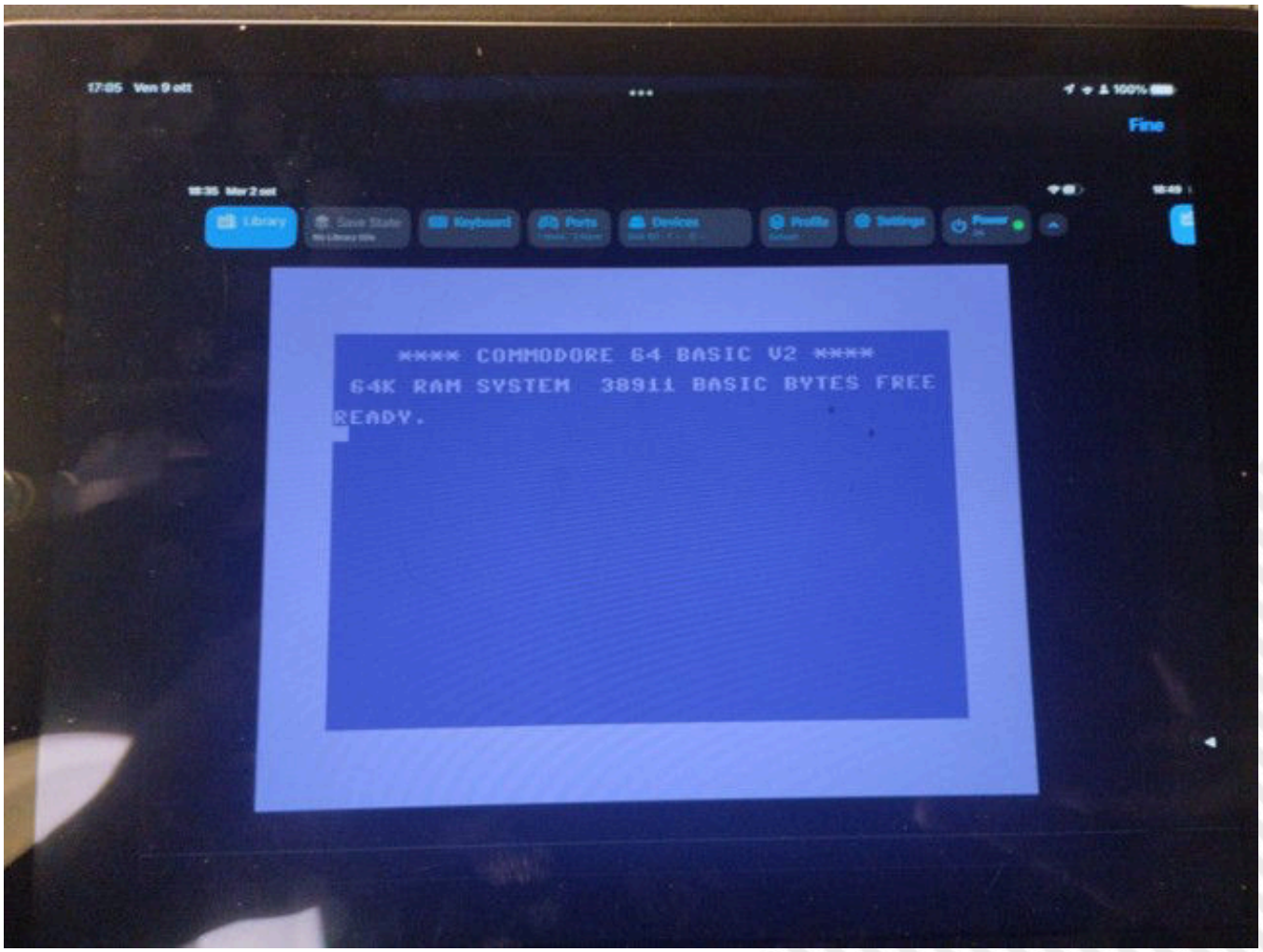
POKE64 – il C64 su... Ipad!!!

di Giampaolo Moraschi

Chi avrebbe mai immaginato che il glorioso “biscottone” di casa Commodore, simbolo di un'intera generazione di videogiocatori e programmatori, sarebbe un giorno finito dentro un elegante tablet Apple? Eppure, quarant'anni dopo le grandi battaglie informatiche degli anni Ottanta, la storia ci regala uno dei suoi scherzi migliori... il C64 torna a vivere proprio sull'hardware della mela morsicata! Il protagonista di questa curiosa riconciliazione tecnologica è POKE64, emulatore sviluppato da Alessandro Capano e disponibile per iPad. Un progetto che non si limita a far girare qualche vecchio videogioco, ma punta a ricreare un ambiente Commodore 64 completo, sfruttando le potenzialità dei moderni dispositivi Apple.

Il cuore del sistema è il celebre VICE, nella sua implementazione x64sc attraverso libretro, capace di emulare con accuratezza il C64 e il C64C nelle modalità PAL e NTSC. L'interfaccia, sviluppata con SwiftUI, UIKit e Metal, è progettata appositamente per iPad, con gestione touch, tastiera virtuale e supporto ai controller esterni.

Sul fronte della compatibilità, POKE64 digerisce praticamente ogni formato storico: D64, D71, D81 e G64 per i dischi, PRG e T64 per i programmi, TAP per le cassette e CRT per le cartucce. Non manca una libreria organizzata con impostazioni dedicate ai singoli titoli, salvataggi rapidi e possibilità di riprendere le sessioni interrotte. Insomma, addio alle interminabili attese davanti al celebre LOAD"*",8,1! Particolarmente interessante la gestione delle periferiche.



POKE64 in tutto il suo splendore





Immagini riprese dall'Appstore di Apple

Possiamo configurare unità 1541, 1541-II, 1571 e 1581, scegliendo tra caricamento virtuale accelerato e modalità True Drive, più fedele al comportamento dell'hardware originale. È supportata anche l'espansione REU da 128 KB fino a ben 16 MB, una quantità di memoria che avrebbe fatto svenire qualsiasi commodoriano del 1982.

Anche il comparto audiovisivo riceve parecchie attenzioni. Il VIC-II viene riprodotto attraverso il motore grafico dell'emulatore, mentre il sonoro SID offre opzioni avanzate, comprese ReSID, ReSID-FP e configurazioni Dual SID. I filtri CRT aggiungono scanline, curvatura dello schermo ed effetti tipici dei vecchi televisori, per chi considera un'immagine perfettamente nitida un'offesa alla nostalgia. I controlli comprendono joystick virtuale multitouch, tastiere esterne, gamepad fisici, mouse Commodore 1351, paddle e KoalaPad, utilizzabile persino con Apple Pencil. Tra le funzionalità più particolari troviamo inoltre la stampante virtuale MPS-803, con esportazione dei documenti, e un modem Hayes virtuale per collegarsi alle vecchie BBS. Alcune caratteristiche avanzate sono ancora in evoluzione o appartengono al ramo beta del progetto.

L'applicazione richiede iPadOS 17 o successivo, occupa circa 10,7 MB ed è disponibile gratuitamente, con uno sblocco Pro opzionale da 7,99 euro (consigliatissimo). Le ROM di sistema e i giochi protetti da copyright non sono inclusi e devono essere forniti dall'utente.

CONCLUSIONI:

POKE64 rappresenta dunque una proposta tecnicamente ambiziosa, soprattutto per chi desidera qualcosa di più di un semplice emulatore da gioco. Resta da verificare sul campo la compatibilità effettiva con le produzioni più complesse, ma le premesse sono decisamente interessanti. E pensare che un tempo Commodore e Apple si contendevano il futuro dell'informatica domestica. Oggi il C64 vive dentro un iPad. Jack Tramiel probabilmente avrebbe avuto qualcosa da ridire. Noi, invece, abbiamo già digitato RUN!

Riferimenti

Github:
- https://github.com/alexain/poke64-ios

Appstore:
- https://apps.apple.com/it/app/poke64/id6803249067

Mean Hamster Software - Quarant'anni tra videogiochi, computer d'epoca e software indipendente

di Takahiro Yoshioka

Nel panorama dello sviluppo indipendente, poche realtà possono vantare una storia tanto lunga e articolata quanto Mean Hamster Software. Fondata nel 1985 da John Swiderski, l'azienda ha attraversato diverse epoche dell'informatica: dalla scena shareware di Commodore 64 e Amiga ai porting per Atari 5200, dai giochi casual per PC e Facebook alle applicazioni mobile, fino agli attuali progetti dedicati a Commodore 64, Atari 5200 e computer x86 d'epoca.

Il nuovo sito ufficiale presenta Mean Hamster come uno studio indipendente attivo da quattro decenni, impegnato nella realizzazione di giochi, sistemi operativi, strumenti e porting per "computer che vale la pena ricordare". L'obiettivo dell'azienda non è soltanto lo sviluppare prodotti per hardware moderno, ma mantenere in vita macchine, piattaforme e idee appartenenti alla storia dell'informatica.

Le origini nella scena shareware

Mean Hamster Software nasce nel 1985 per iniziativa di John Swiderski. La prima fase dell'attività è legata soprattutto a Commodore 64 e Amiga, piattaforme centrali nella cultura informatica domestica degli anni Ottanta e dei primi anni Novanta. Tra il 1988 e il 1993, i primi titoli Mean Hamster vengono distribuiti attraverso la scena shareware. Prima della diffusione capillare di Internet, lo shareware rappresentava uno dei principali sistemi per far conoscere piccoli studi e programmatori indipendenti. Una versione del gioco veniva distribuita liberamente, mentre gli utenti potevano registrarsi per ricevere contenuti completi, supporto o aggiornamenti.

Questa esperienza contribuì a definire l'identità dello studio che prese produzioni accessibili, orientate a piattaforme specifiche e capaci di rivolgersi direttamente a una comunità di appassionati. La storia di Mean Hamster non nasce quindi nel mercato tradizionale delle grandi software house, ma in un ambiente vicino a quello che oggi chiameremmo sviluppo indie. Nel 1995 Swiderski aprì inoltre un'attività commerciale dedicata ai computer a Deer Park, nello Stato di Washington. Il negozio contribuì a sostenere economicamente lo sviluppo dei giochi e rappresentò un punto di contatto con la comunità informatica locale.

Dall'Atari 5200 ai porting mobile

Una parte importante della storia di Mean Hamster riguarda il rapporto con le piattaforme Atari. Tra la fine degli anni Novanta e i primi anni Duemila, lo studio sviluppò diversi giochi per Atari 5200, una console spesso considerata difficile da programmare e distribuita in un periodo relativamente breve.



Una selezione di titoli per Atari 5200 realizzati da Mean Hamster Software, tra cui la serie di Boulder Dash



Un po' di nuove uscite per Atari 5200

Tra i progetti più importanti figurano KLIX 5200, Blockade, Super Attack Battle e Scary Mansion 3D. Il lavoro sull'Atari 5200 non si è limitato a recuperare vecchi titoli, in alcuni casi Mean Hamster ha creato nuove versioni, migliorato giochi esistenti o riportato alla luce progetti rimasti incompiuti.

Parallelamente, l'azienda si dedicò a porting per dispositivi portatili. Nel 2004-2005 adattò Myst e Riven per Pocket PC e Windows Mobile. Portare due avventure grafiche ricche di immagini, filmati, suoni e musica su dispositivi portatili dell'epoca rappresentava una sfida tecnica considerevole. Il progetto dimostra la capacità dello studio di lavorare non soltanto su produzioni originali, ma anche su conversioni di opere note e complesse.

Nel 2006 acquisì inoltre i diritti del catalogo Exidy, comprendente classici arcade come Crossbow, Venture e Mouse Trap. Questa operazione ampliò il patrimonio storico legato allo studio e portò alla realizzazione di una versione mobile ufficiale di Crossbow.

L'epoca dei giochi casual

A partire dalla metà degli anni Duemila, l'attenzione si spostò verso il mercato dei giochi casual. Nel 2008 nacque una collaborazione con PlayFirst, editore particolarmente importante per i giochi destinati al pubblico non specializzato.Il primo grande successo fu Pet Shop Hop, ambientato in un negozio di animali. Il giocatore doveva gestire clienti, animali, potenziamenti e attività quotidiane in una struttura basata sul time management. Il titolo contribuì ad aprire una nuova fase nella storia dell'azienda.

Nel 2009 iniziò a collaborare anche con Big Fish Games. In questo periodo pubblicò titoli come Ye Olde Sandwich Shoppe, Cat Wash, Passport to Perfume, Everything Nice e Mother Nature. Ye Olde Sandwich Shoppe raggiunse il primo posto nella categoria dei giochi di gestione del tempo su Big Fish Games, mantenendolo per sedici settimane consecutive. Questi prodotti avevano caratteristiche molto diverse dai primi giochi per Commodore e Atari. Erano progettati per essere immediati, colorati e accessibili, con meccaniche basate sulla gestione, sulla rapidità e sul completamento di obiettivi progressivi. La varietà delle ambientazioni — negozi di animali, saloni per gatti, profumerie, cucine e mondi naturali — mostrava la capacità dello studio di adattarsi a un mercato in rapida crescita.

Facebook, social gaming e applicazioni mobile

Tra il 2010 e il 2011 John ampliò ulteriormente la propria attività entrando nel settore dei giochi sociali. Il progetto più importante fu la serie Vegas Nights, pubblicata su Facebook, insieme a titoli stagionali come Vegas Frights e a Hospitopia, un gioco gestionale nel quale il giocatore doveva costruire e amministrare un ospedale.

Hospitopia aveva inoltre una componente benefica; il progetto venne sviluppato a sostegno di Children's Miracle Network Hospitals. In questo caso il gioco non era soltanto un prodotto d'intrattenimento, ma cercava di collegare le dinamiche del social gaming a un'iniziativa di raccolta e sensibilizzazione.

Lo studio lavorò anche su porting e progetti destinati a servizi online come GSN WorldWinner, una piattaforma basata su competizioni e tornei. Portò, inoltre, su nuovi sistemi diversi titoli Atari e altri giochi casual. Nel 2012 arrivarono Bee Craft, gestionale dedicato all'allevamento delle api, e R8IT, applicazione mobile sviluppata insieme a Sleepy Giant. Nel 2013 Mean Hamster pubblicò inoltre Girl Frames, un'applicazione per iPhone dedicata alla decorazione delle fotografie.

Questa fase evidenzia quanto fosse ampia la produzione dello studio: non soltanto videogiochi tradizionali, ma anche giochi browser, titoli sociali, applicazioni mobile e strumenti destinati a un pubblico più ampio.



Un Commodore 64 collegato a un monitor moderno mostra un'avventura grafica in stile Sierra. Questo è l'obiettivo di AGI-64.

Il ritorno all'hardware storico

Dopo gli anni dedicati ai casual game e al mobile, l'azienda ha progressivamente recuperato la propria vocazione originaria per l'hardware storico. Nel 2019 lo studio ha pubblicato Break Vaders per Adventure Vision, una console da tavolo molto rara e caratterizzata da una tecnologia di visualizzazione particolare.

Nel 2025 è arrivato TET-TRS per Tandy Model II, computer professionale e aziendale degli anni Ottanta. Il gioco è stato distribuito anche su disco floppy da 8 pollici con confezione e label stampata, una scelta che sottolinea l'attenzione verso il supporto fisico e la conservazione dell'esperienza originale.

Nel 2026 questa direzione è diventata ancora più evidente. Il catalogo comprende nuove edizioni per Atari 5200 e progetti dedicati al Commodore 64, mentre l'azienda ha iniziato a sviluppare anche software di sistema per PC compatibili con DOS.

I giochi Atari 5200

L'Atari 5200 occupa oggi una posizione centrale nella produzione Mean Hamster. Il catalogo disponibile comprende sia giochi originali sia versioni aggiornate di progetti precedenti.

- KLIX 5200 è un gioco basato sulla disposizione e sull'eliminazione di tessere, riproposto in un'edizione cartridge raffinata con movimento più fluido e una giocabilità migliorata.

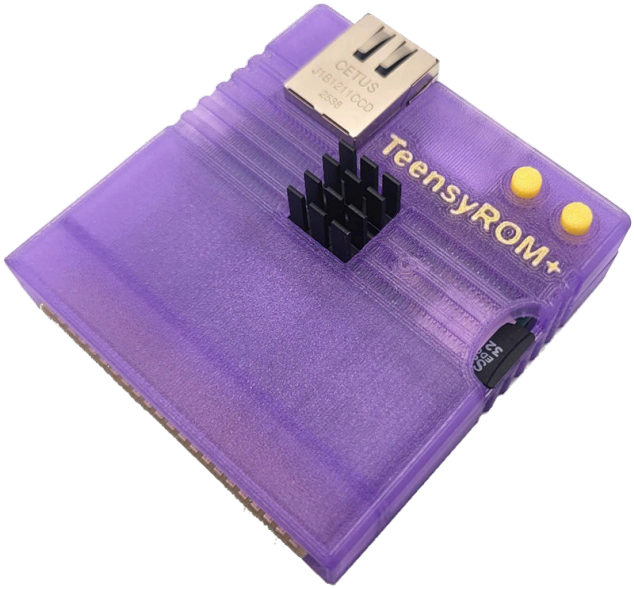
- Scary Mansion 3D+ 5200 amplia l'avventura ispirata all'originale Atari 2600. L'edizione aggiornata propone novanta stanze distribuite in due labirinti, nuove creature, tesori aggiuntivi, correzioni, una schermata iniziale rinnovata e istruzioni aggiornate.

- Super Attack Battle 5200, invece, è un gioco d'azione ispirato a Combat II. Supporta fino a quattro giocatori e mette a disposizione carri armati, elicotteri, sottomarini e jet, con campi di gioco specifici per ciascun tipo di veicolo.

Tra i progetti futuri compare Solo Dragon 5200, previsto per il periodo natalizio del 2026. Il gioco promette musica, pugni, calci, quattordici scene e un'impostazione fortemente orientata all'azione.

Particolarmente interessante è il recupero di Castles & Kings. Il prototipo era stato mostrato nel 2003, ma cancellato prima di poter raggiungere il pubblico. Oggi Mean Hamster sta riportando il progetto su Atari 5200, trasformando un frammento perduto della storia della piattaforma in una pubblicazione concreta.

È prevista anche una nuova versione di Crossbow, classico



La TeensyROM+

Exidy che riceverà grafica aggiornata, musica, effetti sonori, potenziamenti e segreti nascosti. Il remake è destinato a PC, Mac, iOS e Android.

AGI-64: Sierra sul Commodore 64

Uno dei progetti più ambiziosi attualmente in sviluppo è AGI-64, interprete per Commodore 64 progettato per eseguire le avventure Sierra basate sul motore AGI.

L'obiettivo è rendere compatibili sul computer Commodore alcuni classici dell'avventura grafica, tra cui King's Quest e Space Quest. Secondo gli aggiornamenti ufficiali, King's Quest I e Space Quest I sono vicini al completamento, mentre King's Quest IV è in fase di test di compatibilità. Il progetto punta a supportare la maggior parte dei giochi AGI dell'epoca.

L'interesse di AGI-64 non risiede soltanto nella possibilità di giocare vecchie avventure su un computer diverso da quello originale. Un interprete di questo tipo deve gestire testo, grafica, input, animazioni, logica degli script e caricamento dei dati all'interno dei limiti del C64. L'iniziativa rappresenta quindi anche un esercizio tecnico notevole. Il progetto dovrebbe essere distribuito gratuitamente quando sarà completato. Gli aggiornamenti più recenti indicano che Space Quest I è completamente giocabile in una versione ancora in sviluppo, mentre il supporto alla configurazione Magic Desk da 2 MiB ha prodotto risultati incoraggianti.



DOOMVM per TensyRoom+

DoomVM e il MHS Power Engine

Un'altra iniziativa tecnologicamente notevole è DoomVM for TeensyROM+. Mean Hamster ha sviluppato il MHS Power Engine, un motore che trasforma la cartuccia TeensyROM+ in una piattaforma capace di eseguire macchine virtuali.

Il primo risultato è una versione di Doom per Commodore 64, eseguita direttamente su Teensy 4.1. Secondo la presentazione ufficiale, il gioco è veloce e fluido e non richiede una REU o altri dispositivi aggiuntivi.
Il progetto è interessante perché non si limita a realizzare una singola conversione. L'idea è utilizzare il motore come base per ulteriori ambienti virtuali e altri engine. In prospettiva, TeensyROM+ potrebbe quindi diventare una piattaforma per sperimentare giochi e applicazioni che superano le capacità native del C64, mantenendo comunque il computer originale come interfaccia e dispositivo di gioco.

In conclusione Mean Hamster Software è una realtà indipendente con una storia lunga e sorprendentemente diversificata. In un'epoca in cui il retrogaming viene spesso ridotto alla semplice emulazione, Mean Hamster propone un approccio più attivo, creare nuovo software per vecchie macchine, recuperare prototipi dimenticati, aggiornare classici e costruire strumenti capaci di collegare l'informatica vintage al presente. È questa la caratteristica che rende lo studio particolarmente interessante per gli appassionati di Commodore, Atari e computer d'epoca: il passato, nei suoi progetti, non è soltanto qualcosa da conservare, ma un ambiente ancora vivo in cui continuare a sviluppare.

Sito web ufficiale:
- https://meanhamster.com/

Sito TeensyRoom+:
- https://www.tindie.com/products/travissmith/teensyrom-cartridge-for-c64128/

Quattro decenni controcorrente
La storia della System 3

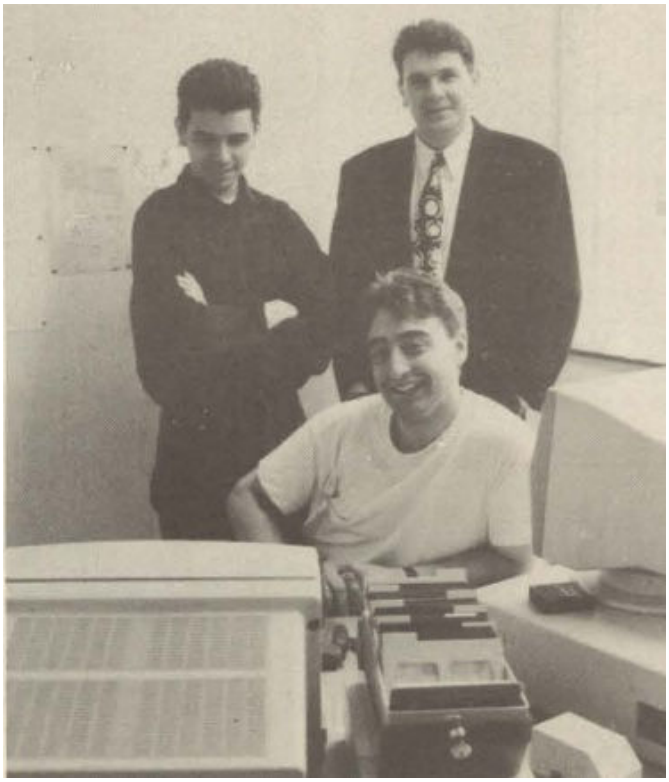
di Roberto Del Mar Pirazzini e Carlo Nithaiah Del Mar Pirazzini

Da un piccolo appartamento londinese alle classifiche americane, dalle arti marziali di International Karate alle atmosfere leggendarie di The Last Ninja, passando per IK+, Myth, Putty e Constructor. Fondata da Mark Cale nei primi anni Ottanta, System 3 ha attraversato oltre quattro decenni di storia dei videogiochi mantenendo una caratteristica sempre più rara: l'indipendenza. Questa è la storia di una software house britannica che voleva fare le cose a modo proprio.

TRE UOMINI E UN'IDEA

Londra, primi anni Ottanta. Il videogioco domestico sta ancora cercando la propria identità. Non esistono formule consolidate, i team sono minuscoli e un singolo programmatore può realizzare praticamente un intero titolo. In Gran Bretagna ZX Spectrum, BBC Micro e Commodore 64 stanno per trasformare una generazione di adolescenti in programmatori, musicisti e grafici. È in questo scenario che prende forma System 3.

La figura centrale della storia è Mark Cale, che arriva nel settore attraverso un percorso decisamente insolito. Terminata la scuola, lavora come assistente fotografo a Covent Garden e successivamente collabora anche con il celebre fotografo Norman Parkinson. La fotografia lo porta a entrare in contatto con Atari UK, per la quale realizza immagini promozionali dei prodotti.



Il team System 3

Cale, però, non si limita a fotografare videogiochi e computer. Li osserva, li gioca e soprattutto li critica. Secondo il suo racconto, furono proprio alcune persone di Atari, dopo aver ascoltato le sue osservazioni sui giochi dell'epoca, a suggerirgli una soluzione molto semplice: se avesse pensato di poter fare meglio, avrebbe dovuto provarci personalmente. Cale raccolse la provocazione.

System 3 nacque dall'iniziativa di Mark Cale, Emerson Best e Michael Koo. Persino il nome rifletteva quella formazione iniziale: erano in tre e Koo frequentava un corso chiamato "System Studies". Koo non avrebbe poi preso parte alla costituzione effettiva dell'azienda, mentre Best sarebbe rimasto con Cale fino alla metà degli anni Ottanta.

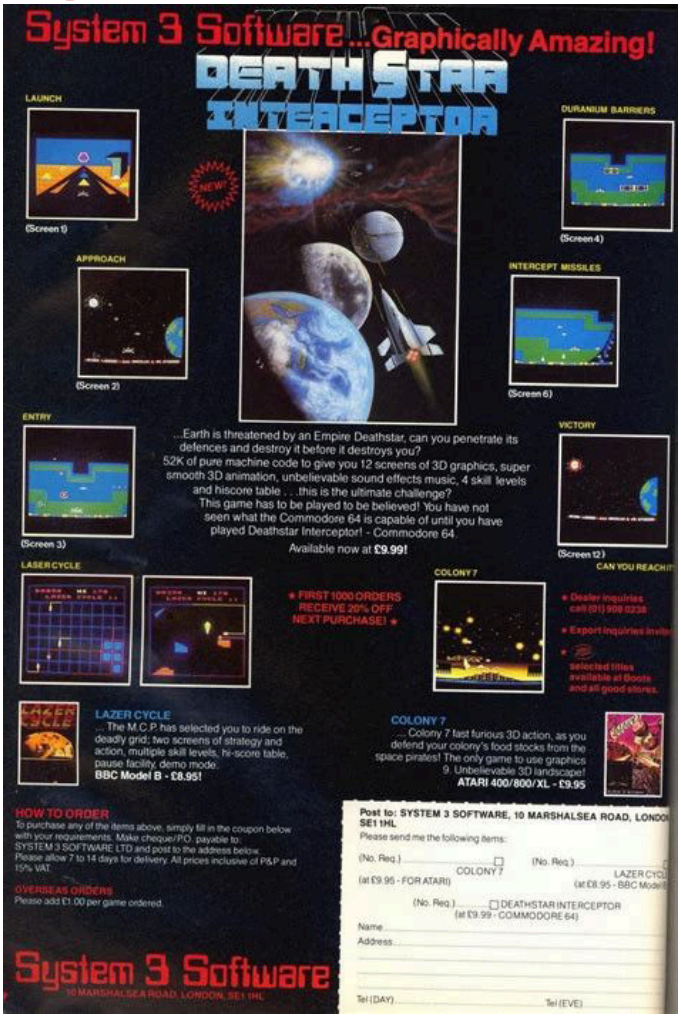
La società fa risalire ufficialmente la propria fondazione al 30 ottobre 1982.

Uno dei primi giochi fu Colony 7 (primo progetto), sviluppato per la famiglia Atari 8-bit. Guardandolo oggi è difficile riconoscere la System 3 che pochi anni più tardi avrebbe fatto tremare le classifiche. Lo stesso Cale non ha mai nascosto la propria insoddisfazione nei confronti di quel debutto… Ma era un inizio!

Seguirono titoli come Lazer Cycle e Death Star Interceptor. System 3 stava imparando il mestiere in un'industria nella quale non esisteva ancora un manuale delle istruzioni. E



Colony 7 per Atari 8bit



La pubblicità di Death Star Interceptor

presto avrebbe trovato la propria strada. Death Star Interceptor rappresentò uno dei primi successi commerciali.

UN KARATEKA NEL COMPUTER

A metà degli anni Ottanta le arti marziali esercitavano un fascino enorme sulla cultura popolare occidentale. System 3 entrò in contatto con LT Software, che iniziò a lavorare a International Karate.

Il progetto cambiò radicalmente quando Cale coinvolse Archer MacLean, programmatore diventato famoso grazie allo straordinario Dropzone. Dopo aver visionato il codice della versione Commodore 64, MacLean decise sostanzialmente di ripartire.

Il risultato fu il gioco che cambiò la storia di System 3.

International Karate era immediato, spettacolare e sorprendentemente tecnico. Due karateka si affrontavano in scenari esotici cercando di mettere a segno colpi valutati dall'arbitro. Il combattimento aveva ritmo, precisione e soprattutto una qualità che avrebbe presto caratterizzato molte produzioni System 3: era facilissimo capire cosa fare, molto meno facile diventare davvero bravi.

Negli Stati Uniti il gioco venne distribuito da Epyx con il titolo World Karate Championship. System 3 afferma che la versione Commodore 64 raggiunse il primo posto nelle classifiche statunitensi dedicate al software e superò complessivamente 1,5 milioni di copie nel mondo. Per una giovane società britannica era un risultato straordinario. System 3 non era più una piccola etichetta che cercava il proprio posto nel mercato. Aveva trovato il suo primo campione.

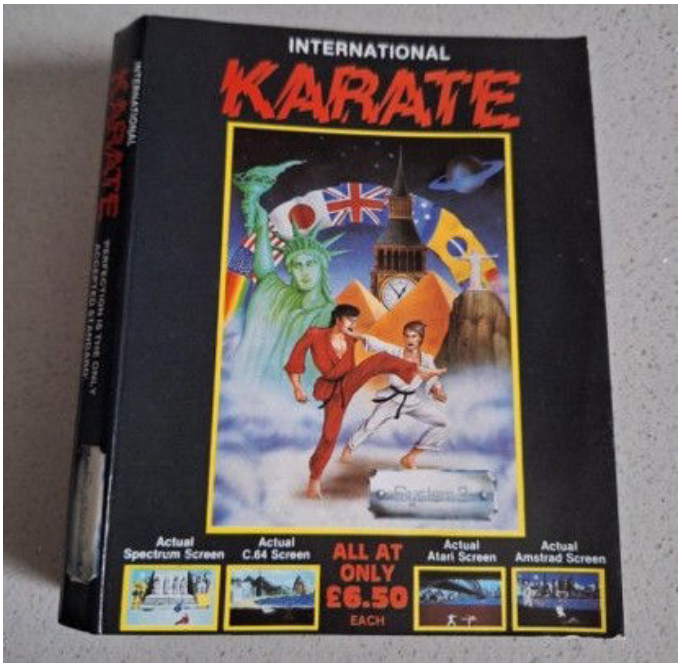
TRE KARATEKA SONO MEGLIO DI DUE

Qualunque altra società avrebbe probabilmente realizzato un seguito limitandosi ad aggiungere scenari e qualche nuova mossa. System 3 scelse una strada differente. Nel 1987 arrivò International Karate +, meglio conosciuto semplicemente come IK+. Sullo schermo adesso combattevano contemporaneamente tre karateka.

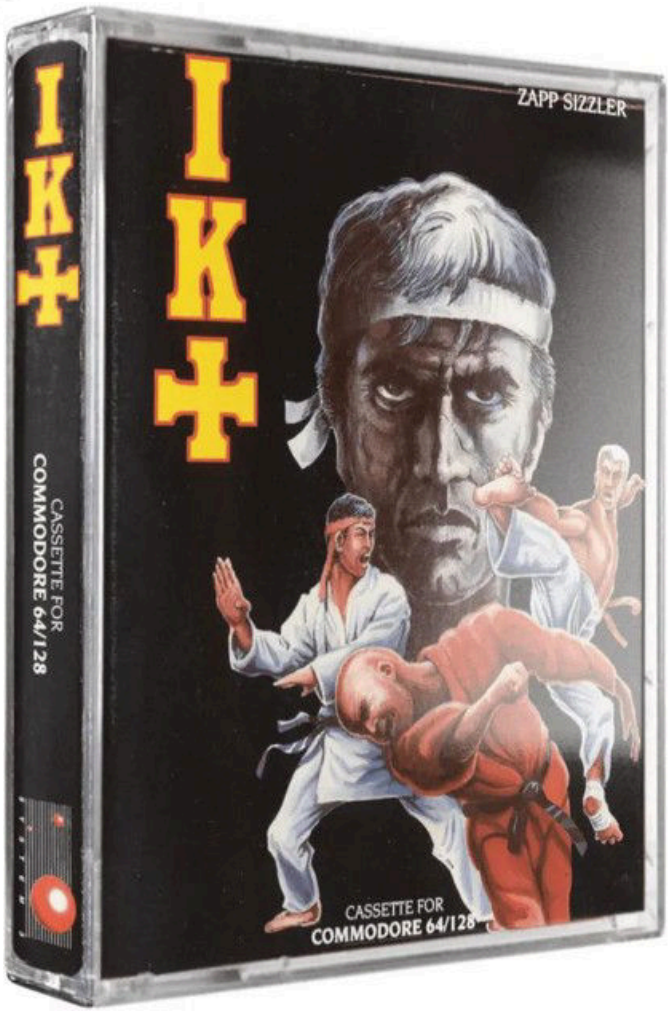
Era una modifica apparentemente semplice, ma cambiava completamente la dinamica degli incontri. Il giocatore non doveva più controllare soltanto la posizione del proprio avversario: doveva interpretare un piccolo caos coreografico nel quale attacchi, schivate e contrattacchi potevano arrivare da entrambe le direzioni.

Ancora una volta Archer MacLean trasformò limiti tecnici in spettacolo. Le animazioni erano fluide, gli scenari pieni di piccoli dettagli e la componente umoristica emergeva continuamente. Il mare sullo sfondo, il ragno che poteva scendere dall'alto, le espressioni dei combattenti: IK+ possedeva personalità.

Ma mentre i karateka si prendevano a calci, System 3 stava preparando qualcosa di molto più ambizioso.



Il gioco che cambiò lo stile di sviluppo di System 3



IK+ per Commodore 64

L'ULTIMO NINJA

Mark Cale voleva realizzare un'avventura capace di unire combattimento, esplorazione ed enigmi. L'ispirazione arrivava anche da giochi che aveva amato personalmente, ma l'obiettivo era costruire qualcosa con una propria identità. Nacque così The Last Ninja.

Lo sviluppo fu tutt'altro che semplice. Una prima parte del progetto venne realizzata in Ungheria attraverso la collaborazione con Andromeda e un gruppo guidato da István Bodnár. Era ancora l'epoca della Cortina di Ferro e lavorare tra Gran Bretagna e Ungheria significava affrontare difficoltà che andavano ben oltre la programmazione.

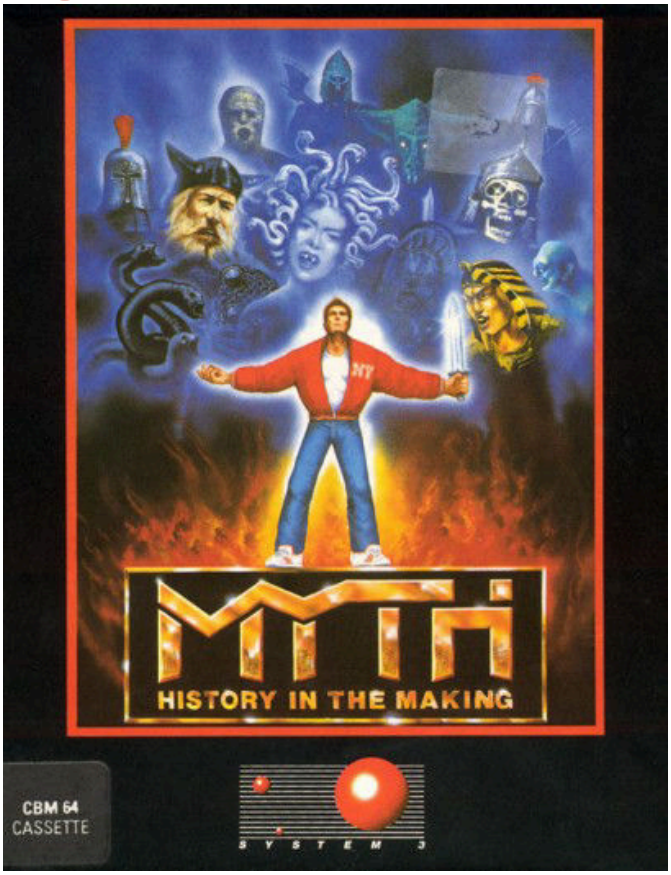
Il gruppo ungherese contribuì soprattutto allo sviluppo degli strumenti utilizzati per costruire gli scenari. Un sistema chiamato Integrator permetteva di assemblare elementi grafici riutilizzabili e creare ambienti molto più complessi di quanto la memoria del Commodore 64 sembrasse consentire. L'idea era brillante…Il gioco, però, non funzionava ancora.

System 3 si trovò davanti a una decisione difficile. Conservare mesi di lavoro oppure ripartire quasi da zero? La risposta arrivò con John Twiddy.

Il programmatore britannico prese in mano il progetto e riscrisse profondamente il sistema, conservando alcune delle soluzioni sviluppate nella fase precedente. Hugh



Isometrico e bellissimo... The Last Ninja



Myth con il vecchio logo di System 3

Riley contribuì in maniera determinante alla grafica e alle animazioni.

Nel 1987 The Last Ninja era finalmente pronto.

E il Commodore 64 sembrò improvvisamente diventare una macchina diversa.

BENVENUTI A LIN FEN

L'isola di Lin Fen appariva sul monitor attraverso una spettacolare visuale isometrica. Giardini, templi, corsi d'acqua, rocce, statue e sotterranei costruivano un mondo nel quale Armakuni doveva combattere, trovare oggetti e risolvere enigmi.

Poi c'era la musica. Anthony Lees era stato inizialmente coinvolto per comporre la colonna sonora. Quando emerse la necessità di avere ulteriore musica, anche per accompagnare i lunghi caricamenti della versione su cassetta, System 3 coinvolse Ben Daglish. Il risultato è diventato parte integrante della leggenda del gioco. Le melodie del SID davano a Lin Fen una dimensione quasi cinematografica. Non accompagnavano semplicemente l'azione: costruivano atmosfera.

The Last Ninja divenne un enorme successo internazionale e il volto mascherato di Armakuni diventò inseparabile dal logo System 3. Quello con i tre pianeti orbitanti.

Il celebre marchio apparso nella seconda metà degli anni Ottanta rappresentava simbolicamente i tre fondatori originari e sarebbe rimasto nell'immaginario degli appassionati come il logo dell'età d'oro della società.

IL NINJA SBARCA A NEW YORK



Coloratissimo e giocabilissimo... Putty su Amiga!



Constructor

Un successo come The Last Ninja richiedeva inevitabilmente un seguito.

Ma System 3 non voleva semplicemente riportare Armakuni negli stessi templi. Nel 1988 arrivò Last Ninja 2, e questa volta il ninja si ritrovò catapultato nella New York contemporanea. Central Park, strade, uffici e stazioni della metropolitana sostituivano il Giappone feudale. Era una scelta sorprendente, ma funzionava. Il contrasto fra un guerriero tradizionale e la metropoli americana diede al gioco una personalità fortissima. La formula venne affinata, gli ambienti divennero ancora più spettacolari e la componente audiovisiva continuò a essere uno dei punti di forza. Armakuni era ormai una star.

Nel 1990 arrivò Last Ninja Remix, mentre nel 1991 Last Ninja 3 riportò la serie verso atmosfere maggiormente legate alla tradizione orientale. System 3 aveva costruito una saga capace di sopravvivere al passaggio dagli 8 ai 16 bit. Ma la software house non viveva soltanto di ninja.

MITI, ELEFANTI E VENDETTE

La fine degli anni Ottanta rappresenta probabilmente il periodo nel quale il marchio System 3 possiede la sua identità più riconoscibile.

Titoli come Bangkok Knights, Tusker, Vendetta e soprattutto Myth: History in the Making mostrano una società interessata a giochi visivamente ambiziosi, spesso caratterizzati da grandi sprite e atmosfere fortemente riconoscibili.

Myth, pubblicato nel 1989, porta il giocatore attraverso differenti epoche mitologiche, permettendogli di incontrare creature e divinità provenienti da diverse tradizioni. Era esattamente il tipo di produzione che il pubblico ormai associava al logo con i tre pianeti: grafica elaborata, confezione accattivante e la sensazione che qualcuno avesse tentato di spremere dall'hardware qualche goccia in più di quanto fosse ragionevole aspettarsi.

Ma il mercato stava cambiando.

Amiga e Atari ST avevano alzato enormemente le aspettative. Il colore, il sonoro e la memoria disponibile permettevano produzioni più sofisticate. Poi sarebbero arrivate le console a 16 bit.

System 3 doveva trasformarsi ancora.



Il ritorno di Armakuni è rimasto… un sogno.

ARRIVA PUTTY

Nel 1992 apparve un personaggio completamente differente da Armakuni.

Era blu, elastico e decisamente poco minaccioso.

Putty poteva allungarsi, schiacciarsi, gonfiarsi e assorbire oggetti. Con Putty e successivamente Putty Squad, System 3 entrò nel territorio dei platform a 16 bit mostrando ancora una volta grande attenzione per animazioni e caratterizzazione grafica. Era anche la dimostrazione che l'identità della società non dipendeva da un unico genere. System 3 aveva attraversato gli sparatutto degli esordi, i picchiaduro, le avventure isometriche e ora i platform.

La sfida successiva sarebbe stata ancora più radicale. Questa volta niente ninja, niente karateka, niente blob blu… Soltanto costruttori, gangster, hippie, operai e vicini di casa decisamente poco raccomandabili.

BENVENUTI NEL PEGGIOR QUARTIERE DEL MONDO

Nel 1997 arrivò Constructor. A prima vista sembrava un gestionale urbano. Il giocatore acquistava terreni, costruiva case, produceva materiali e amministrava una piccola città. Ma Constructor non aveva alcuna intenzione di comportarsi come un tranquillo simulatore.

I rivali potevano essere sabotati utilizzando personaggi assurdi e metodi ancora più assurdi. Gangster, hippie e altri "indesiderabili" trasformavano la pianificazione urbana in una guerra di quartiere impregnata di umorismo britannico. Il risultato fu uno dei giochi più particolari del catalogo System 3. La stessa società ricorda Constructor come un importante successo del 1997 e come uno dei titoli che segnarono il suo ingresso nell'era PlayStation. Era trascorso circa un decennio da The Last Ninja. Il mondo dei videogiochi era irriconoscibile, ma System 3 era ancora lì.

SOPRAVVIVERE ALLE GENERAZIONI

Gli anni successivi portarono nuovi cambiamenti. La società lavorò su numerosi formati e affiancò alla produzione interna attività editoriali e commerciali. Il marchio Play It venne utilizzato nel mercato dei titoli a prezzo contenuto e, secondo la ricostruzione aziendale, arrivò a collocare System 3 fra gli editori con importanti volumi di vendita

sulle piattaforme Sony in Europa.
Parallelamente l'azienda continuò a sviluppare e valorizzare proprietà intellettuali proprie.
Arrivarono progetti automobilistici come Ferrari Challenge: Trofeo Pirelli, mentre Constructor tornò attraverso nuove versioni e reinterpretazioni. Anche Putty sarebbe ricomparso su piattaforme moderne. C'è però un fantasma che continua ad attraversare tutta la storia della società.

Last Ninja 4

Il ritorno di Armakuni venne tentato più volte. Il progetto attraversò differenti incarnazioni, piattaforme e accordi commerciali, arrivando in alcune fasi alla realizzazione di materiale giocabile.
Non arrivò mai nei negozi. Secondo Mark Cale, una delle versioni era stata progettata nell'epoca della prima PlayStation; successivamente il progetto passò attraverso nuovi accordi e tentativi di sviluppo, compreso un rapporto con Electronic Arts. Un'altra incarnazione sarebbe stata realizzata molti anni più tardi.

Ogni volta qualcosa fermò il ninja. Problemi commerciali, trasformazioni del mercato, costi di produzione. Armakuni rimase nell'ombra.

43 ANNI DOPO

Il 30 ottobre 2025 System 3 ha celebrato ufficialmente 43 anni di attività. La società rivendica una caratteristica particolarmente importante per comprendere tutta la sua storia: il marchio è rimasto indipendente e non è stato venduto o assorbito da un grande gruppo.
Nel frattempo, il passato è tornato a essere futuro. The Last Ninja Collection ha riportato insieme i capitoli storici della saga e altri titoli del catalogo, mentre System 3 ha continuato a lavorare sulla conservazione delle proprie produzioni classiche. E la storia non sembra essersi fermata. Nel settembre 2026 System 3 ha mostrato al Tokyo Game Show una prima demo tecnologica giocabile di Putty World 35th Anniversary, progetto destinato a riportare il suo improbabile eroe blu su Nintendo Switch 2. È significativo che, oltre quattro decenni dopo gli esperimenti su Atari 8-bit, il logo System 3 continui ad apparire accanto a un gioco in sviluppo. Perché forse è proprio questa la parte più interessante della sua storia.

System 3 non ha attraversato soltanto diverse generazioni di hardware. Ha attraversato differenti idee di videogioco. Ha conosciuto l'epoca nella quale un programma poteva essere scritto da una sola persona e venduto dentro una custodia per cassette. Ha vissuto la rivoluzione del Commodore 64, l'arrivo dei 16 bit, l'esplosione delle console, il CD-ROM, PlayStation, la distribuzione digitale e infine il ritorno commerciale del retrogaming.
E in ogni epoca ha portato con sé il proprio archivio.
International Karate.
IK+.
The Last Ninja.
Myth.
Putty.
Constructor.

Titoli diversissimi, accomunati da una filosofia che nell'età degli 8 bit appariva quasi una dichiarazione d'intenti: cercare di ottenere dalla macchina qualcosa che sembrasse leggermente oltre ciò che quella macchina avrebbe dovuto essere in grado di fare.
Forse è per questo che il logo System 3 continua a suscitare una reazione immediata in chi visse quegli anni.
Tre sfere orbitanti comparivano sullo schermo.
Partiva la musica.
E sapevi che stava per succedere qualcosa.

Quarant'anni dopo quei tre pianeti stanno ancora girando.

Sito ufficiale
- https://system3.com/

Fonti:
- Sito Ufficiale
- Datormagazin 17/1990
- Datormagazin 9/1989

La promessa del 1983 che torna sul C64… Shadow Snare

di Carlo Nithaiah Del Mar Pirazzini

Dopo 43 anni Michael Cranford riapre la strada lasciata in sospeso alla fine di Maze Master. Non un revival, ma un nuovo RPG nativo Commodore 64, scritto in 6502 e pensato per i limiti (e le opportunità) della macchina. Ecco cosa sappiamo, cosa gioca la demo e perché vale la pena seguire il progetto. Il tutto mentre sosteniamo in beta test per il rilascio finale che avverrà nei primi mesi del 2027.

Una storia che aspetta da quattro decenni

Alla fine di Maze Master, cartuccia 16K per Commodore 64 uscita nel 1983, compariva una scritta destinata a diventare leggenda tra gli appassionati: “Adventure 2: Shadow Snare”. Quel seguito non arrivò mai. Michael Cranford — mente dietro a Maze Master e, poco dopo, a The Bard’s Tale (1985) e The Bard’s Tale II: The Destiny Knight (1986) — prese un’altra direzione, lasciando Shadow Snare come “la strada non percorsa”.

Nell’agosto 2026, però, Cranford ha riaperto ufficialmente il progetto. Shadow Snare è di nuovo in sviluppo, come RPG in prima persona nativo per C64, scritto in assembly 6502 e costruito entro i reali vincoli di memoria, colore, suono e storage della macchina. Non si tratta di ricostruire un gioco mai completato, ma di mantenere quella promessa con quarant’anni di esperienza in più — tra game design, architettura software, filosofia ed etica — e con gli strumenti di oggi al servizio di una visione personale.



Fig. 1 - Intro

Il concept: un eroe, molti compagni, otto vie

Shadow Snare è pensato come un dungeon crawler fantasy in prima persona, ma con un’impostazione diversa rispetto ai party classici. Si controlla un singolo personaggio, attorno al quale ruotano compagni (NPC) che si uniscono alla quest principale, aiutano a risolvere enigmi e contribuiscono alla narrazione. In combattimento sono previste animazioni dinamiche e testo di battaglia a scorrimento; la progressione del personaggio permette di attaccare più volte per turno, “sintetizzando” in un eroe esperto la forza di un intero gruppo.

L’avventura si sviluppa lungo le “Eight Ways” (le Otto Vie), percorsi di formazione e potere che conducono a uno scontro finale con il Lord of Shadow. Le Vie non sono semplici scuole di magia intercambiabili, ma discipline che insegnano al viaggiatore a vedere, scegliere e diventare. Ogni dungeon mette alla prova più della forza bruta e ogni scelta lascia un segno sul protagonista. Tra le vie citate nel materiale ufficiale compare ad esempio “Candle”, legata a fuoco, luce, fumo e “piccoli soli sgarbati”.

Demo giocabile: motore completo, sette livelli, storia in divenire

Attualmente è disponibile una demo funzionale, riservata a tester approvati. Il motore di gioco è completo, sono implementati sette livelli di dungeon campione e l’interfaccia mostra già selezione incantesimi, esplorazione dei dungeon, combattimenti e un primo assaggio di ambientazione (tra cui il villaggio di Brackenmere). Cranford sta raccogliendo feedback su engine e gameplay per affinare il prodotto prima del rilascio completo, previsto per la prima metà del 2027. La demo non è una ROM pubblica: si richiede



Fig. 2 - Diffidate dei Goblin

l'accesso tramite il sito ufficiale e, una volta approvati, si ottiene il file da eseguire su hardware reale o emulatore. L'obiettivo è coinvolgere una cerchia ristretta di giocatori capaci di fornire osservazioni puntuali; i tester verranno accreditati e coinvolti anche nella fase di testing del gioco completo.

Perché "nativo C64" fa differenza

Cranford insiste su un punto chiave. Shadow Snare non è "un gioco moderno con pixel vecchi", ma un titolo progettato per il Commodore 64 stesso, dentro i suoi limiti. Su C64 ogni byte conta e questo costringe a decidere cosa il gioco ha davvero bisogno di essere. È una costrizione che può esasperare, ma che diventa fertile quando spinge a concentrarsi sull'essenziale. Il risultato, dalle prime impressioni sulla demo, è un'esperienza che si sente "della macchina". L'interfaccia risulta compatta e facile da comprendere, c'è una gestione attenta della memoria, combattimenti e dungeon costruiti per funzionare bene



Fig. 3 - Spesso conviene fuggire dalle battaglie più complicate

entro i noti vincoli hardware. Per chi segue la scena homebrew, è esattamente il tipo di progetto che distingue il neo-retrogaming più interessante. Non una ripetizione del passato, ma nuovi giochi plasmati da vecchi limiti.

Dove e quando si vedrà (e come provarlo)

Cranford ha annunciato la propria presenza alla Portland Retro Gaming Expo (PRGE), dove prevede di mostrare Shadow Snare e parlare del progetto. Sul sito ufficiale sono disponibili informazioni sulla demo, sulla storia del gioco e sui contatti per stampa e playtest. L'intenzione è chiudere lo sviluppo nella prima metà del 2027; se il responso della community sarà positivo, non esclude di trasformare Shadow Snare in una serie.

La pagina ufficiale è la seguente - https://shadowsnare.com. Da qui potete richiedere l'accesso per testare la demo.

La nostra prova

Da fanatico dei giochi di ruolo classici, non posso che ritenermi soddisfatto dalla prima prova su "pista" del gioco. Shadow Snare ha una bella trama, un motore di gioco convincente e un bilanciamento già notevole. Sono da limare ancora gli incontri con alcuni avversari/NPC, ma la comunity che testa il gioco sta già aiutando l'autore a sistemare questo piccolissimo problema. Shadow Snare è uno di quei casi in cui la storia dell'industria e la scena homebrew si incontrano. Non ci resta che continuare ad aiutare l'autore a testare il gioco e aspettare l'uscita definitiva.

Intervista a Michael Cranford

di Carlo Nithaiah Del Mar Pirazzini

Abbiamo raggiunto via mail Michael Cranford, creatore di Master Maze, The Bard's Tale e all'opera con l'ottimo Shadow Snare per porgergli un po' di domande in merito a questo "nuovo" percorso Retro.

Nith - Prima di parlare di questo ritorno, raccontaci qualcosa del tuo percorso formativo nel corso degli anni e delle tue esperienze.

Michael - Tutto è iniziato quando studiavo architettura all'Università della California, Berkeley. Ero seduto in fondo a una lezione su Shakespeare, leggendo il giornale dell'università. Ho visto un annuncio per programmatori di giochi 6502 e mi sono candidato il giorno dopo. Un'azienda software di San Francisco stava aprendo una sede a Berkeley e cercava sviluppatori. A quel tempo non esisteva un percorso consolidato per entrare nel game design: mi ero insegnato da solo qualche anno prima e programmare era un mio hobby. Mi assunsero sul momento e, al semestre successivo, lasciai l'università per dedicarmi a tempo pieno allo sviluppo di videogiochi. Circa quattro anni dopo, con l'uscita di The Bard's Tale, tornai a interessarmi all'ambito accademico e decisi di riprendere gli studi. Vivevo a Irvine, in California (lavoravo per Interplay) e trasferii i miei crediti all'Università della California, Irvine, cambiando indirizzo in filosofia e completando la laurea triennale. A quel punto avevo abbandonato lo sviluppo di videogiochi (per molte ragioni) e stavo iniziando a lavorare nello sviluppo di applicazioni web. Proseguii anche negli studi, ottenendo un Master in Teologia, e poi fui ammesso a un dottorato in Religione ed Etica Sociale all'Università della California del Sud, grazie a una fellowship per merito. Il mio lavoro di dottorato si concentrò sull'etica applicata, con un interesse particolare per etica e tecnologia. In quegli anni non smisi mai di realizzare software. Diventai architetto software, insegnai filosofia e, alla fine, trascorsi un periodo come cappellano ospedaliero. Oggi insegno filosofia, bioetica ed etica sanitaria, e sono tornato a lavorare sullo stesso computer per cui sviluppavo negli anni Ottanta. Sulla carta, questi possono sembrare diversi percorsi professionali. Per me sono sempre stati connessi. Giochi, software, filosofia ed etica pongono tutti una versione della stessa domanda: che tipo di mondo stiamo costruendo e che tipo di persone diventeremo al suo interno?



N - Shadow Snare fu un grande annuncio per tutti i fan degli RPG dell'epoca. Vederlo tornare è davvero magico. Raccontaci come hai deciso di riportarlo in vita.

M - La scintilla è stata la Portland Retro Gaming Expo del 2025. Mi fu chiesto di tenere una retrospettiva su The Bard's Tale e, mentre ero lì, rimasi stupito nel vedere quanto interesse ci fosse ancora per i vecchi giochi. Non



La bellezza estetica di Shadow Snare in sviluppo

sapevo che ci fosse così tanta curiosità per il tipo di giochi che realizzavo un tempo. Riconobbi anche Bruce Gottlieb, e lui mi ispirò a iniziare a riflettere su questo. Quando tornai a casa, ordinai un Commodore 64 Ultimate e configurai un ambiente di sviluppo sul mio PC per scrivere codice 6502 per il Commodore. Quasi fin dall'inizio, sapevo cosa avrei prodotto... Shadow Snare. Il nome appariva alla fine di Maze Master nel 1983. Era la promessa di un'altra avventura. Avevo un concept per un RPG più centrato sulla magia e seguito da un singolo personaggio giocatore, con una schiera di PNG. Quel concept fu accantonato. Proseguii creando The Bard's Tale, diventai architetto software e passai decenni a costruire altri tipi di mondi. Ma questo non mi ha mai davvero lasciato andare.Per molti anni ho dato per scontato che l'epoca di una singola persona che crea un mondo di gioco completo fosse passata. I giochi moderni sembravano richiedere team enormi, studi e budget importanti. Avevo provato alcune volte in passato a tornare al game design, ma tutte le porte erano chiuse. Quando scoprii una comunità viva di sviluppatori indipendenti che creavano ancora giochi personalmente, inclusi giochi per le macchine che usavamo quarant'anni fa, capii che avrei riprovato, proprio come feci la prima volta: con gli stessi vincoli di memoria, con la stessa grafica, sotto la mia direzione. Gli strumenti moderni spiegano come ho potuto tornare. Non spiegano perché. Shadow Snare conteneva ancora una storia che volevo raccontare e un tipo di esperienza di gioco di ruolo che non avevo mai finito di realizzare. Non sto cercando di imitare un gioco che avrei potuto fare nel 1983. Sto mantenendo quella promessa con tutto ciò che ho imparato da allora. Mi sembra giusto che il viaggio ricominci sul "Breadbin" (Biscottone, ndN).

N - Dopo Shadow Snare, cosa hai in programma per il futuro?

M - Prima di tutto, intendo finire Shadow Snare. Al momento abbiamo una demo funzionante e lavorerò per sistemare gli ultimi dettagli prima che il gioco completo prenda forma. Punto a completarlo nella prima metà del 2027.

Se la gente lo amerà, lo trasformerò in una serie.È anche la prima porta verso un viaggio più ampio. Ho idee su dove potrebbe andare questo mondo e le forme che la sua storia potrebbe eventualmente assumere, ma ho imparato a non annunciare le destinazioni prima che la strada le raggiunga. Prevedo di fare un annuncio al PRGE di quest'anno. Continuerò anche a insegnare e scrivere di etica, tecnologia e formazione umana. La cosa sorprendente è che game design, architettura software e filosofia non mi sembrano più parti separate della mia vita. Shadow Snare sta diventando il luogo in cui si incontrano. Voglio vedere cosa cresce da tutto questo. Ma prima… finire il gioco!

N - Come vedi questa "nuova ondata" di neo-retrogaming?

M - Ciò che mi interessa di più di questa nuova ondata non è la parte "retro". È il ritorno della paternità personale. I grandi team possono creare giochi straordinari. Ma la scala cambia quali rischi si possono correre, quali idee sopravvivono e quanta parte della visione di una singola persona può rimanere intatta. Lo sviluppo indipendente restituisce la possibilità che un gioco porti l'immaginazione riconoscibile della persona che lo ha creato. Le macchine più vecchie impongono anche limiti severi. Sul Commodore 64, ogni byte conta. Ogni colore, suono, immagine e sistema compete per lo spazio. Questo può essere esasperante, ma è anche fertile. Un vincolo ti costringe a decidere di cosa ha davvero bisogno il gioco. Certo, il retrogaming può diventare un esercizio di ripetizione. L'affetto per il passato è positivo, ma se ci limitiamo a riprodurre ciò che è già stato fatto, queste macchine diventano reperti da museo. Gli sviluppatori più entusiasmanti stanno usando hardware vecchio per creare cose che non sono mai esistite prima. Non giochi moderni con pixel vecchi. Nuovi giochi plasmati da vincoli vecchi. L'hardware è vecchio. L'atto creativo no.

N - Grazie per il tuo tempo e per tutte le risposte. Speriamo di vederti in Italia a qualche raduno di appassionati di retrogaming. Hai qualcosa da dire ai nostri lettori?



M - Prima di tutto, grazie ai vostri lettori per aver ricordato queste macchine — e per essere rimasti curiosi di qualcosa di nuovo creato per una di esse. Mi piacerebbe molto tornare in Italia. L'ho visitata solo una volta, viaggiando da Firenze fino a Napoli, ed è rimasto uno dei migliori viaggi che abbia mai fatto. Ricordo il calore delle persone, l'atmosfera d'altri tempi e del cibo davvero straordinario. Avrei bisogno di ben poca persuasione per tornare, specialmente se l'invito coinvolgesse vecchi computer e persone che li amano ancora.Ai vostri lettori — che fossero presenti quando il Commodore 64 fece la sua prima apparizione o che lo abbiano scoperto solo di recente — direi questo: le cose incompiute non sono sempre morte. A volte stanno solo aspettando il momento giusto, gli strumenti giusti e le persone giuste. Shadow Snare ha aspettato. Ora la strada è aperta. Spero che giocherete e mi direte cosa ne pensate, così che io possa crearne altri.

Un veterano della demoscene internazionale Intervista ad "ice00" aka Stefano Tognon

di David «Cercamon» La Monaca

Intro

In questo numero di RMW incontriamo Stefano Tognon (aka "Ice00"), una figura di spicco nel panorama del retrocomputing e dello "scene coding" italiano e internazionale. Cresciuto con la passione per l'informatica fin dagli anni dell'età dell'oro degli home computer, ha legato indissolubilmente il proprio nome alla scena del Commodore 64, trasformando una semplice passione giovanile in un vero e proprio percorso di padronanza tecnica e divulgazione scientifica.

Come scener e programmatore/coder, Ice00 ha dato un contributo importante alla demoscene del C64, un ambiente altamente competitivo dove l'obiettivo è spingere l'hardware a 8 bit ben oltre i limiti strutturali. Sfruttando ogni singolo ciclo di clock del processore MOS 6510 e spremendo le capacità dei chip grafico (VIC-II) e audio (SID), Stefano ha firmato e collaborato alla creazione di diverse demo, routine grafiche e trucchi di programmazione a basso livello, oltre immagini bitmap, brani musicali per il SID, tool di programmazione e sviluppo per PC finalizzati a produrre codice e file multimediali per C64.

La sua attività non si limita tuttavia alla sola scrittura di codice e alla partecipazione alle competizioni internazionali con vari gruppi. Ice00 è riconosciuto a tutti gli effetti come un instancabile divulgatore. Attraverso workshop, articoli tecnici, video su Youtube, conferenze e incontri dedicati alla storia del computer vintage, Stefano condivide con grande entusiasmo la sua profonda conoscenza dell'architettura hardware del C64, spiegando con chiarezza e accuratezza sia le basi dell'assembly sia le tecniche più avanzate di ottimizzazione.



L'intervista che segue è stata strutturata per conoscere ed approfondire sia la sua storia personale e la sua filosofia nella demoscene, sia gli aspetti più squisitamente tecnici del suo lavoro di programmazione e preservazione di hardware/software.

Parte 1: Le Origini e la Demoscene

1. Il primo contatto: Qual è stato il primissimo computer su cui hai messo le mani e qual è stato il momento esatto in cui hai capito che non volevi solo usare il software, ma crearlo?

Il primissimo computer che ho posseduto è stato il Philips Videopac G7000 credo all'età di 6 anni. A corredo c'erano due cartucce gioco (il filpper e la battaglia navale) ma, nonostante avesse una (pseudo) tastiera, per programmarlo serviva probabilmente una apposita cartuccia, pertanto lo ho sempre e solo usato come console. Purtroppo (o per fortuna) si ruppe abbastanza velocemente (di lui credo di avere conservato in soffitta solo il poster pubblicitario che elencava tutti i giochi..chissà quanto vale oggi per i collezionisti..)., per cui ottenni il C64 appena venne commercializzato in Italia.

E li era un altro pianeta. Nonostante avessi una cartuccia gioco (Lazarian, gioco che potrei adesso giocare ad occhi chiusi ma a quell'età era difficile da padroneggiare) e una marea di cassette, la possibilità di programmare seguendo il manuale utente era qualcosa di grandioso. A questo si aggiunse il corso «Introduzione al Basic» di Andrew Colin che mi permetteva di capire i primi algoritmi e di come il C64 avesse molte caratteristiche tutte da scoprire, tra cui il linguaggio macchina: quella misteriosa serie di numeri inseriti come DATA nel BASIC e che si avviava con l'altrettanto misterioso comando SYS. Era un mistero da svelare assolutamente per capire come diavolo facevano a realizzare i giochi!

Ma è in realtà grazie al SID che sono diventato programmatore. Dopo aver sentito le musiche che il SID produceva, da autori come Martin Galway e Matt Gray

(che a quel tempo non ne conoscevo i nomi, ma solo che i loro brani erano stupefacenti), volevo assolutamente imparare a programmare per essere in grado di generare suoni di quel tipo (evento che si è poi concretizzato ufficialmente nel 2008 con il rilascio del mio tracker JITT64). Credo di avere ancora la cassetta col brano Driller suonato dal C64 e registrato sullo stereo tramite un cavo che costruì per l'occasione. Era un modo veloce per sentire il brano senza dover caricare il gioco nel C64!

2. L'origine del nick: Da dove nasce il nickname Ice00 e qual è il suo significato (se ne ha uno)?

Deriva dall'acronimo «Informatic Computer Engineer», che però era troppo lungo per poterlo usare ad esempio in una mail (dove c'era il limite di 5 caratteri dato dal provider). Per cui scelsi "ice00", e i due zeri aggiunti era per renderlo più orecchiabile, un po' come in 007!

3. I primi passi nella Demoscene: Come sei entrato in contatto con l'ambiente delle demo e dei party, in un'epoca in cui internet non era diffuso come oggi?

Diciamo che è grazie a internet che nel 2001 ho iniziato a partecipare alle minigame competition (appuntamento per me immancabile fino al 2006 dove ottenni buoni risultati) spedendo via mail il mio contributo agli organizzatori. Così facendo iniziai a comparire su CSDB, cosa che dava una visibilità sicuramente internazionale. Dall'altro lato avevo un sito personale sulla piattaforma Digilander che era collegato alla mia mail su libero.it (il primo provider Italiano che permetteva gratuitamente l'utilizzo di mail e sito web in Italia a ridosso del nuovo millennio) su cui avevo pubblicato anche i miei lavori sul C64. Il sito era visibile fino a qualche mese fa, poi è stato cancellato (senza alcuna notifica, quindi il materiale contenuto è probabilmente andato perso per sempre)!

Nel 2002 iniziai anche a pubblicare SIDin, il mio magazine PDF tutto dedicato al SID, modellato per essere in stile della e-zine C=Hacking, che mi diede ancora più visibilità, tanto che venni contattato per scrivere articoli sulla nascente fanzine cartacea italiana chiamata "Reload" in cui scrissi 3 articoli sul funzionamento del SID (purtroppo la rivista cessò la pubblicazione dopo solo 2 numeri, ma in quegli anni era sicuramente difficile portare avanti un progetto del genere).

Fu così che nel 2003 venni contattato via mail da Enrico Barbisan che organizzava Spresiano Retrocomputing, una mostra relativa alle macchine degli anni 80 che teneva in una palestra parrocchiale di Spresiano in autunno, aperto a tutte le categorie (Enrico è appassionato di MSX, ma ha sempre puntato a portare a Spresiano qualsiasi hardware tra cui i sistemi Commodore). Dato che Spresiano era più o meno a mezz'ora da casa, decisi di partecipare, portando con me il mio PC e un Commodore 64, in modo da mostrare alle persone come sviluppavo i giochi sul computer e poi li riversavo su cassetta per provarli direttamente nel C64. Sembra incredibile, ma li ci furono delle persone che venivano da lontano (dal Piemonte o Lombardia) che avevano letto la rivista Reload e quindi mi riconobbero come l'autore degli articoli sul SID!

Se non ricordo male ho partecipato anche alle successive due (o tre) edizioni, prima che purtroppo la mostra chiuse i battenti (tra l'altro Enrico venne poi assunto nella mia stessa ditta dove lavoravo nel reparto IT, quindi siamo stati in ufficio assieme per lungo tempo :)).

In ogni caso ho maturato la convinzione nel tempo che la scena è anche improvvisazione: hai una idea? Bene, mettila in pratica. Poi si vedrà come viene accolta dalla comunità. Ad esempio, le tre «SID Wine Competition» che ho organizzato dal 2002 al 2004 sembravano totalmente irrealistiche (oggi sarebbero impossibili da replicare), ma invece sono state un ottimo successo di partecipazione e ancora adesso ho persone che mi scrivono ricordando quelle manifestazioni!

Anche le due «Tiny SID compo» organizzate nel 2005 e 2006 sembravano fantascienza: produrre (ottima) musica in soli 256 o 512 byte! Eppure la partecipazione è stata incredibile e ancora oggi alcuni dei pezzi li prodotti vengono usati da vari gruppi all'interno delle loro intro quando hanno bisogno di risparmiare spazio, ma al contempo offrire ottima musica!

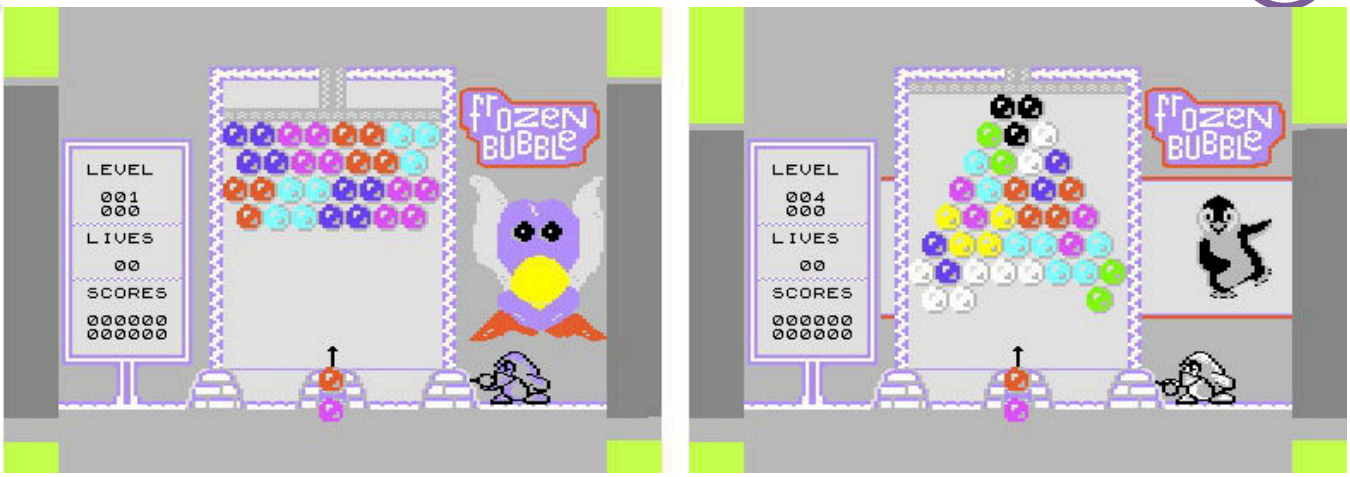
4. Gruppi ed eredità: Sei stato attivo in vari gruppi italiani e internazionali. Quali sono i ricordi più vivi legati alle collaborazioni con altri scener (grafici, musicisti, coder)?

Nel lontano 1994 ho fondato il mio gruppo Ice Team, di cui sono stato l'unico membro per la parte C64, mentre ho avuto altri collaboratori per la parte relativa al PC. Ovviamente, come accennato prima, tutto è esploso verso il 1999 con l'arrivo di internet grazie all'uso delle email con cui ci si poteva coordinare, e per me, il forum di Ready64 è stato essenziale agli inizi per trovare aiuti dalla comunità (è infatti attivo dal 2002).

In particolare Luca/Fire ha realizzato due colonne sonore per altrettanti minigame con il vincolo che doveva produrre i brani usando poco spazio, ma senza limiti musicali. Luca è il mago dell'8580. Mettete «Soulfixer» a tutto volume e capirete cosa intendo!

Mi ricordo che per il gioco «Little Sara Sister» mi disse che gli era venuta in mente la musica di Pinocchio (quella dello sceneggiato RAI) e ne ha estratta una versione SID perfetta per il poco spazio che c'era a disposizione.

Per «Little Sara Sister 2», che era un gioco da 16KB, ho



Frozen Bubble

avuto l'onore di avere la musica creata da Dustbin (Stefano Palmonari) che, per chi non lo conosce, ha lavorato nientemeno che per la Simulmondo (il sogno proibito che avevamo tutti all'epoca...)!

Per altri due minigame («Ann Gong» e «Little Sara Sister 1.5») le musiche sono state create da Richard Byliss. Richard è un vero mago. Se fosse nato nel periodo d'oro del C64, oggi sarebbe ultramilionario perché sforna giochi a raffica, tutti molto carini e che sarebbero stati pubblicati ad occhi chiusi. Oggigiorno è talmente impegnato con i suoi progetti per il C64 che non ha purtroppo un secondo libero! In ogni caso per me è il nuovo Matt Gray. La sua musica è inconfondibile, con oltre 1300 produzioni e ore e ore di ascolto (no, che dico, vi serve una settimana per ascoltarle tutte ammesso che non cadiate in tentazione di lasciarle andare ad oltranza perché spesso dedico 1h solo ad un singolo brano...).

Tornando ai gruppi, nel 2012 venni contattato da Overkiller per unirmi ad Hokuto Force, cosa che feci. A quel tempo non ricordo che piattaforma usassimo, ma oggi siamo tutti sincronizzati con Discord. Nel gruppo mi sono concentrato sopratutto nei comparto giochi, dove realizzai «Little Sara Sister Trilogy» per Hokuto Monastery. In pratica tutto era nato perché, dopo aver pubblicato «Little Sara Sister 2» per la competizione RGCD (16KB), Overkiller mi disse che un publisher voleva distribuire il gioco! Si trattava della nascente FlimSoft, il cui proprietario Jamie Alvis. persona decisamente squisita con cui lavorare, mi chiese se ero interessato di realizzare una versione apposita da pubblicare. Realizzai allora una versione estesa di LSS2 per utilizzare tutto lo spazio disponibile nella memoria del C64 e decisi di riunire anche le versioni precedenti di LSS (la 1 e la 1.5) per realizzare una trilogia che si poteva avviare tramite un menù di scelta all'inizio. I grafici dentro Hokuto Force mi hanno aiutato a definire le varie schermate di caricamento di cui avevo bisogno per il progetto (purtroppo la corrispondenza era nel vecchio sistema e col cambio a Discord ho perso il dettaglio degli autori). Il gioco venne quindi pubblicato su floppy disk nel 2013. Grazie Alvis!!

Per chi si chiedesse quanto ho guadagnato economicamente in questa vendita (malloppo che avrei diviso, in base al contributo dato, coi vari collaboratori) dovrebbe farsi un giro con la mia Ferrari comprata coi guadagni (... ahahah, no, quella della macchinona comprata col lavoro di coder l'ho letta da qualche parte e il programmatore era purtroppo un altro... ;)) . Purtroppo Flimsoft, appena avviata, non aveva ancora una struttura tale da permettergli di sopravvivere facilmente a quei tempi (oggi con il mercato molto più florido sarebbe probabilmente diverso), quindi decisi di lasciare i guadagni maturati fino a quel momento ad Alvis sperando che potesse proseguire nel lavoro. Infatti avevo pronta anche la versione su cassetta di «LSS2 Trilogy», ma Flimsoft chiuse i battenti prima di riceverla :(

Giusto per concludere questa digressione, il caricatore su cassetta di LSS2 Trilogy, che ricordo deve caricare uno dei 3 giochi selezionati dall'utente, sfrutta il principio di salvare i 3 programmi in pezzettini posti uno intervallato all'altro, quindi non serve avvolgere il nastro per posizionarsi nella posizione in base al numero di giri, ma il loader carica il programma giusto man mano che legge il flusso (scartando i dati degli altri quando incontra i loro pezzi). Credo che sui vari caricatori per cassetta ci sarebbe materiale per realizzare un libro storico/tecnico: è incredibile cosa si riesce a realizzare anche con una tecnologia così primitiva!

Tornando alle collaborazioni, un altro progetto futuro contiene le musiche realizzate direttamente da Saul Cross! Non serve aggiungere altro, Saul è una garanzia per qualunque lavoro su grafica o musica realizzi :)

5. Lo spirito del Demo Party: Partecipare a una

composizione o a un party dal vivo porta con sé un'energia unica. C'è un aneddoto da un evento in particolare che ti è rimasto impresso?

Tornando a Spresiamo Retrocomputing c'è un aneddoto particolare che vale la pena di raccontare. Stavo lavorando al gioco Markanoid, un clone di Arkanoid con musica ma in solo 4KB per partecipare alla minigame competition del 2003. Come ho accennato sopra la musica la realizzò Luca Carafiello. Luca mi spedì il PSID di sabato (il giorno prima del termine della consegna per la competizione) dopo averla composta mentre era a letto con la febbre a 38,5°! Dopo aver provato il tutto mi serviva solo ricollocare alcune parti del codice per avere la compressione ottimale e aggiustare il timing corretto per il brano. Decisi di farlo la mattina seguente, prima di andare a Spresiano per il meeting. Solo che il mattino seguente non c'era energia elettrica nella mia zona! Panico totale! Decisi di partire ugualmente per Spresiano dove per fortuna c'era la corrente. Dopo aver assemblato il tutto provai il gioco sul C64 reale... ma non andava (se non ricordo male la pallina non si muoveva o non rilevava le collisioni), mentre sull'emulatore Vice 1.12 funzionava perfettamente. Coincidenza, il sabato era stato rilasciato Vice 1.13 di cui avevo scaricato i sorgenti, per cui li compilai al volo e provai il gioco: stesso risultato del C64! Cavolo, avevo poche ore per sbrogliare la matassa! Li al meeting con le persone presenti cercai la causa dell'errore che riuscii per fortuna a correggere e far testare il gioco sul C64 reale. Se non ricordo male il problema era per le ottimizzazioni inserite per usare le routine presenti nel Kernal invece di riscrivere codice per fare quella operazione, risparmiando spazio. In pratica, versioni differenti del Kernal avevano comportamenti diversi nel punto richiamato, per cui il kernal del C64 che avevo portato al meeting era come quello di Vice 1.13 che differiva da quello di Vice 1.12. Viceversa, il C64 portato al meeting era il nuovo modello preso su Ebay (il C64 originale lo preservo il più possibile) che probabilmente aveva la versione aggiornata del Kernal, come quella attivata di default in Vice 1.13, da cui i problemi rilevati.

Tornando indietro nel tempo, una delle prime azioni fatte negli anni 90 era stata quella di dumpare il KERNAL, il BASIC e il CHARGEN da mio C64 originale, scrivendole su cassetta e poi trasferendole sul PC, dove li disassemblai con la prima versione del mio disassemblatore (che era realizzato in Pascal a quel tempo). Pertanto mi sono sempre basato su quel Kernal quando volevo vedere cosa faceva una routine di sistema, ignorando le differenze esistenti tra versioni successive.

Un ultimissimo aneddoto, anche se non collegato relativamente ai party, è stato quando ricevetti una email da Martin Galway in persona! Si era complimentato per l'analisi del suo player usato in Arkanoid che avevo fatto nel numero di SIDin e, forse perché quello era il periodo in cui avevo in corso le Tiny Sid Compo, mi raccontò come dovette ridurre il suo player in un gioco per sottostare agli stringenti limiti che gli erano stati imposti. Veramente fantastico!

Parte 2: Sfide Tecniche e Programmazione

6. Programmare al limite: Le produzioni della Demoscene si basano sulla capacità di spremere l'hardware oltre i suoi limiti teorici. Qual è il trick o l'ottimizzazione di cui vai più orgoglioso?

Qui è difficile rispondere perché ogni demo si basa su «magie» attuate dai programmatori che hanno dell'incredibile. Credo però che tutto si possa riassumere col fatto che si riesce ad eseguire il codice al ciclo esatto. Senza questo trucco di programmazione la maggior parte dei metodi avanzati usati nei demo sarebbero impossibili (o comunque non sarebbero stabili). In realtà, si riesce ad attivare questo trick con più metodi (il che fa capire che i programmatori a volte riescono a trovare soluzioni diverse allo stesso problema).

Giusto per dare il contesto ai lettori: quando avviene un interrupt del processore, la routine di gestione può essere ritardata fino a 7 cicli a causa della lunghezza variabile dei cicli nelle istruzioni (perché l'istruzione corrente si deve comunque completare). Pertanto se abbiamo, ad esempio, una interruzione alla linea zero, ciclo zero del raster, la nostra istruzione se inizia in ritardo non può attivare un trick del VIC che richiede la scrittura in un preciso istante (esempio ciclo 14, perché a volte l'istruzione si attiva al 15, al 17 o al 20).

Un altra ottimizzazione che tutti, e intendo tutti, hanno apprezzato (anche se non proprio nella demoscene in senso stretto) è la tecnologia di caricamento veloce che possiamo riassumere in due parole: Turbo Tape! Si, i caricamenti dei giochi su cassetta sono lenti, parliamo di ben 4 minuti, ma sarebbero stati di quasi 30 minuti se non si usasse la tecnologia turbo. Non c'è storia!

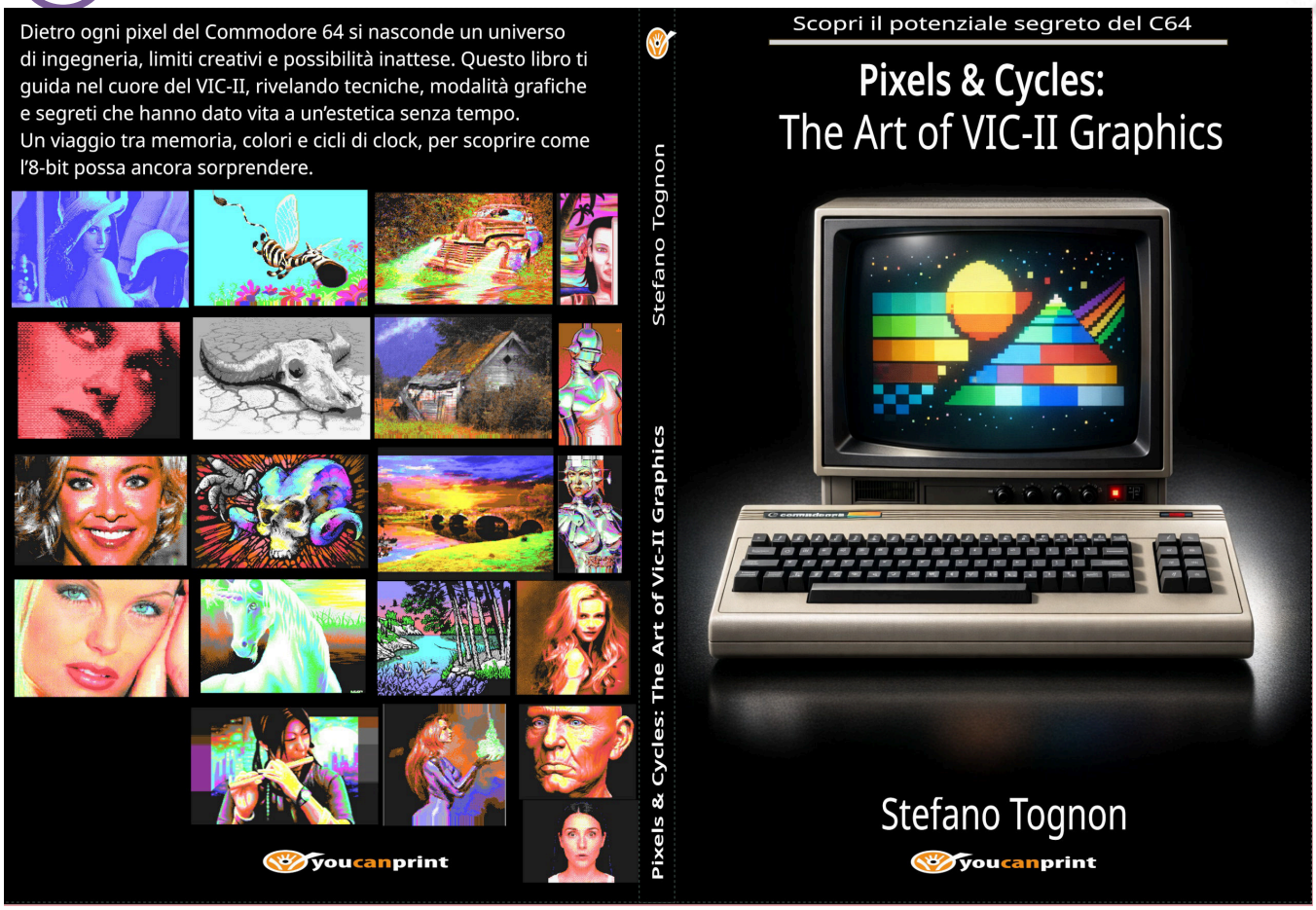
Tornando alla demoscene vera e propria credo che il top sia la routine con cui il SID riesce a riprodurre musica digitalizzata a 8 bit con frequenza di 48KHz (ovvero maggiore di un CD audio). Questo grazie al nostro Antonio Savona e al lavoro iniziale di Pex Tufvesson che ha inventato la tecnica.

7. L'arte dell'Assembly: L'Assembly richiede un approccio mentale molto diverso rispetto ai linguaggi di alto livello moderni. Come si sviluppa questa "forma mentis" per pensare direttamente in termini di registri e cicli di clock?

C'è una differenza, ma è legata alla macchina non strettamente all'assembly. Facciamo qualche premessa. Nel mio libro (...tomo da biblioteca) pubblicato l'anno scorso sulla programmazione in Java, la prefazione parte proprio sul definire chi è un programmatore. A mio avviso un vero programmatore deve:
- saper creare un linguaggio di programmazione scrivendo il suo compilatore e interprete.
- saper programmare in assembly.
Le due cose sembrano una l'opposto dell'altra. Da un lato usiamo un linguaggio ad alto livello, con cui possiamo usare la programmazione strutturata senza l'uso del GOTO e ci astraiamo il più possibile dalla macchina. Dall'altro nella programmazione assembly il salto JMP ne è alla base e ci immergiamo a bassissimo livello in tutto, dai registri, alla memoria, agli interrupt e le porte di I/O.
In realtà le cose sono collegate. Programmare in linguaggio macchina significa effettivamente essere a bassissimo livello, mentre il linguaggio assembly è molto più potente e porta uno strato di astrazione paragonabile quasi a quello che da la programmazione strutturata.
Ad esempio, l'uso di macro permette di scrivere pezzi di codice complesso in modo che appaia molto più lineare se lo guardiamo, inoltre l'uso di indentazione opportuna e commenti rendono immediato capire il significato del codice. Senza contare la creazione di routine che sono sempre il cuore della programmazione strutturata. Gli interrupt invece, possono essere visti come dei thread temporizzati o richiamati da eventi nei linguaggi moderni, con la differenza che quando sono in esecuzione, tutto il resto è bloccato.
La mia conoscenza dell'assembly era nata sia per poter programmare il C64 (chip MOS 6502), ma anche per disassemblare i virus informatici che colpivano i PC (x86) negli anni 80 e per poter creare degli antivirus, senza poi contare che come informatico prima o dopo capita un corso su qualche processore diverso (a quei tempi era ad esempio il Motorola 68000).
Torniamo però al demo coding per C64. Qui bisogna considerare che stiamo parlando anche di routine sincronizzate al singolo ciclo di clock, ciclo che può essere anche sospeso perché il VIC II necessita di leggere dalla RAM in quel momento. In questo caso viene a cadere il concetto di programmazione strutturata che si applica abbastanza anche in un programma in assembly. Qui le routine devono eseguire certe istruzioni in certi punti quindi il codice apparentemente può sembrare un ammasso di istruzioni poste in un ordine senza senso, senza logica (o anche contrarie alla buona programmazione). Ma la logica c'è ed è per garantire l'azione giusta al momento giusto. A mio avviso la difficoltà non è l'assembly in sè, ma è dover strutturare l'ordine delle istruzioni tenendo conto della loro durata in cicli di clock per ottenere l'obiettivo finale. Questo non è affatto semplice e non è da tutti. Qui serve ragionare oltre le righe, bisogna essere, per così dire, nella stessa lunghezza d'onda della macchina!

8. Piattaforme del cuore: Se dovessi scegliere una singola architettura hardware su cui hai amato di più programmare (Amiga, C64, PC x86 o altro), quale sarebbe e perché?
Io adoro programmare in Java perché non sono legato ad una macchina in particolare: scrivo nel mio PC su Linux e il programma poi gira ovunque c'è una JVM. Amiga non l'ho mai abbracciato e credo che l'assenza del SID fosse l'handicap per cui non mi ha mai interessato. La musica con suoni campionati di Amiga potrà avvicinarsi a quella di strumenti reali, ma il suono del SID è un altro pianeta. Il C64 è però insostituibile! Programmare e anche andare a riscoprire i programmi esistenti via reverse engineering è sempre entusiasmante! Però lo faccio sempre da PC tramite cross compilazione ed emulatori, quindi la risposta alla tua domanda è il PC purché abbia Linux con KDE che utilizzo ininterrottamente dal lontano '99!

9. Old-school vs. Modern: Con l'evoluzione delle GPU e delle CPU moderne, l'approccio alla grafica in tempo reale è cambiato radicalmente. Cosa si è guadagnato e cosa si è perso, secondo te, rispetto alla programmazione bare-metal del passato?
Io oggi piuttosto che giocare ad un gioco moderno preferisco avviare ad esempio un qualsiasi prodotto di Cadaver su C64 (Metal Warrior, Hessian o Steel Ranger) oppure molti altri giochi recenti prodotti ad esempio da Antonio Savona e altri (senza contare tutti i classici degli anni 80). C'è più divertimento, più immersione nel gioco. La grafica foto-realistica non è sinonimo di giocabilità, che poi è il fattore che spesso manca nei giochi moderni. Prendiamo ad esempio Tomb Rider. La grafica non è eccelsa ma "cubettosa", l'illuminazione è molto approssimativa (ambienti aperti sembrano quasi al chiuso) ma il gioco è fantastico. Se prendiamo Tomb Rider 3 abbiamo probabilmente il meglio: spazi giganteschi da esplorare, illuminazione migliorata, grafica sempre "cubettosa" ma molto più definita, giocabilità ai massimi livelli. Poi prendo Tomb Rider del 2013 in cui abbiamo una grafica mozzafiato, coi fili d'erba che si muovono col vento, ma il gioco non l'ho mai finito perché manca quel sapore che solo gli altri sapevano dare (e le versioni successive le ho in libreria ma non le ho mai nemmeno installate). Viceversa, i giochi remastered usciti di recente, sono invece stupendi: si migliora la grafica e l'illuminazione dei giochi originali, mantenendo però la stessa struttura,



Libro Pixels & Cycles: The Art of VIC-II Graphics

pertanto lasciando inalterata la giocabilità.

Altro esempio: Tremulous. Questo gioco open source nasce dai sorgenti di Quake III del 2000 e quando esce nel 2006 ha già una grafica 3D in prima persona che molti definirebbero già antiquata. Ma è invece un portento, secondo me. La giocabilità è incredibile perché presuppone la collaborazione dei team che impersonano gli umani contro quelli degli alieni. La grafica passa in secondo piano. È l'azione che la fa da padrona. Serve strategia, coordinazione dei partecipanti. Serve conoscere ogni angolo della mappa per poter sfruttare opportunità uniche per salvarsi dal nemico. Un solo elemento rimasto vivo in un team può capovolgere la situazione che sembrava disperata e sconfiggere il nemico.

Purtroppo al giorno d'oggi la rincorsa alla grafica all'ultimo grido (ad esempio anche tramite ray-tracing) è la strada che percorrono sia i giocatori sia i produttori (giocare a Tremulous oggi è pressoché impossibile perché nessuno ne installerebbe l'eseguibile per provare qualcosa di così graficamente "vecchio"), pertanto si sta perdendo progressivamente l'elemento giocabilità, il vero fattore speciale nei giochi.

Sento dire che in futuro le avventure punta e clicca scompariranno perché nessun produttore di giochi oggi si cimenterebbe ad implementare una dinamica di gioco così vetusta. Mi chiedo: i nuovi giocatori hanno mai giocato ad Indiana Jones o a Monkey Island? Probabilmente la risposta è no e purtroppo solo noi veterani sappiamo cosa si sono persi!

Rimpiango poi le sale giochi piene di quei bei cabinati in cui passavi ore con arcade divertentissimi. Oppure i videogame che potevi trovare anche nei bar (oggi purtroppo sostituite dalle macchine "succhia soldi" legalizzate...). Ho letto tra l'altro di recente uno studio in cui i giochi retro, con grafica minimalista come nel C64, hanno molta presa nelle persone, perché il cervello sottobanco ricostruisce una versione più dettagliata della scena attingendo alla nostra memoria (anche emozionale), mentre al contrario in un gioco con grafica troppo realista il cervello ha troppi dati visivi ed è più incline a trovare i difetti e a rendersi conto che il tutto è finto perché non perfetto. Non so quanto sia attendibile lo studio, ma sicuramente sembrerebbe aderente al mio pensiero!

10. Dimensione dei vincoli (sizecoding): Creare intro da 4KB, 64KB o pochi byte richiede sacrifici enormi in termini di codice. Come si bilancia la complessità dell'effetto visivo con la rigidità dei limiti di memoria e risorse?

Qui va ricercato il giusto mix. Avendo partecipato alle minigame competition o alle mie Tiny SID Compo conosco bene il campo. Il primo punto da considerare è che più

spazio abbiamo e più possiamo contare sulla compressione dei dati, anche se il decompressore occuperà anch'esso spazio. Idealmente con 4KB possiamo scrivere codice in 8KB e ridurlo a 4KB. E' un trucco che va usato. Invece quando si lavora con 256b o 512b non possiamo più contare sulla compressione ma solo con la scrittura accorta del codice.

Ad esempio, per produrre un mini tune, dobbiamo creare il timbro degli strumenti, memorizzare le note e la loro durata per far eseguire per le 3 voci del SID. Il timbro lo creiamo andando a variare i parametri del SID dinamicamente, come ad esempio la pulse modulation dell'onda quadra, aggiungendo vibrato alla nota, tenendo sempre conto che vanno impostati gli ADSR e CTRL appropriati al restart della nota. Se invece l'obbiettivo è un brano lungo, per forza di cose dobbiamo diminuire il codice di gestione timbro perché la sola memorizzazione di nota e durata si mangia gran parte dello spazio. Viceversa un timbro elaborato, magari con tabelle di valori come nei tracker moderni, ridurrà drasticamente le note e quindi la durata del brano.

Esaminiamo invece il caso di un gioco. Qui ci servono gli sprite del personaggio principale e dei nemici. Sono 64 byte a sprite. Più dettagli per i movimenti e più sprite servono. Più nemici ci sono e ancora più sprite servono. Poi dobbiamo memorizzare il nostro charset (se non usiamo i caratteri PETSCII) e sono altri 2KB. Se ricicliamo i primi 128 caratteri (usando quindi le lettere ASCII standard) lo riduciamo a 1KB. Poi dobbiamo creare le schermate usando i nostri caratteri. Una schermata sono 1000 caratteri. Più schermate usiamo e più la gestione a blocchi (esempio 4x4) diventa conveniente anche se dobbiamo memorizzare i blocchi in memoria e complicare il codice di visualizzazione. Poi ci va la musica (idealmente calcoliamo 3KB per una colonna sonora di un certo peso). A questo si aggiunge tutto il codice di gestione del gioco. Se facciamo due conti sembra impossibile avere «LSS 1» che permette di giocare al clone di «Great Giana Sister» con scorrimento orizzontale e musica in soli 4KB! Ovviamente ci sono stati dei compromessi: si memorizzano solo gli sprite destri, quelli sinistri li si ricava a runtime generandoli simmetricamente. Trucco che ci fa risparmiare quasi metà spazio per gli sprite. Il codice poi lo si crea auto modificante se ci permette di far risparmiare spazio rispetto alla versione estesa di una routine. La musica occupa spazio ma non è detto che tutto serva. Spesso non tutte le funzioni del player sono sfruttate, quindi c'è del codice che mai verrà usato durante la riproduzione. Se poniamo a 0 i punti del codice non usato, otteniamo che la compressione dati aumenta di efficacia permettendoci di risparmiare preziosi byte.

Parte 3: Preservazione, Emulazione e Progetti Collaterali

11. Archeologia informatica: Oltre alle demo, ti sei occupato di preservazione e reverse engineering. Quanto è importante oggi salvare il codice e i media storici prima che vadano perduti per sempre?

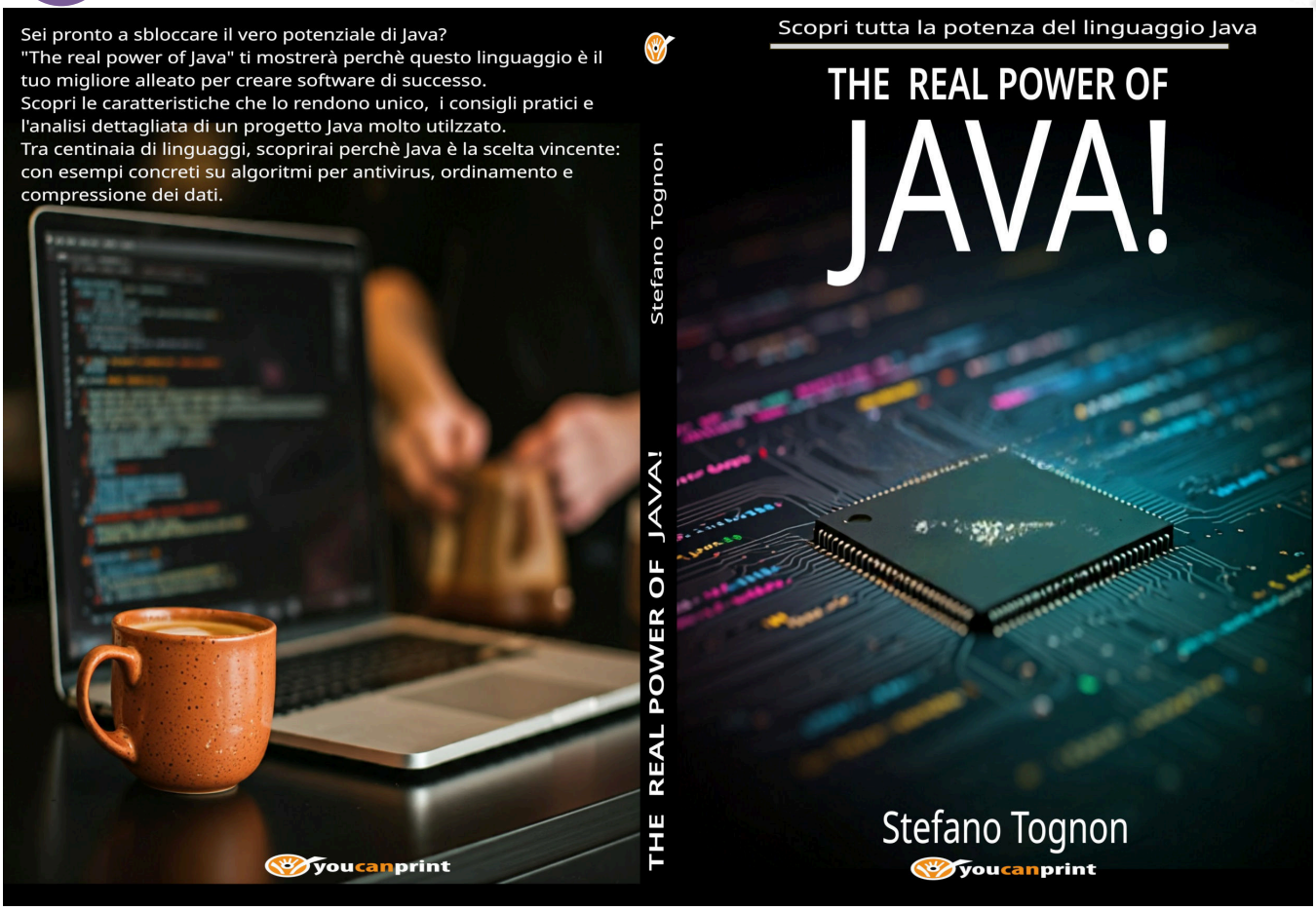
Da amante del SID ho sempre voluto capire come fosse programmato, per cui cercare i vari editor e provarli è sempre stata una costante. Entravo sui siti FTP in cerca di programmi audio, ma dopo un po' di tempo mi sono trovato pieno di programmi in cui ogni volta che li aprivo non sapevo più che tasti usare e dovevo cercare a tentativi da capo.

Per cui organizzai tutto il materiale creando l' «High Voltage Music Engine Collection», in modo da poter provare in tempo zero un editor e sapere che tasti premere. La collezione, che era realizzata su HTML statico, la pubblicavo su un sito gratuito (mi sembra fosse Lycos) che era il più permissivo, essendo il materiale si vecchio, ma tendenzialmente ci potevano essere problemi di copyright.

La collezione è quindi cresciuta aggiungendo centinai di editor e tracker ed era molto apprezzata. Purtroppo poi il sito venne cancellato per la chiusura del provider, quindi cercai di ricostruirlo in versione dinamica usando Wordpress, cosa che purtroppo è onerosa come tempo e anche in quel caso dovetti cambiare più volte web hosting provider.

In seguito, dopo aver programmato in PHP e Javascript proprio per i plugin di Wordpress sono giunto alla conclusione che è un miracolo che internet non sia un colabrodo basandosi su queste tecnologie... quindi al momento non ho una soluzione che mi soddisfi per preservare tutto questo materiale in un modo che sia semplice da manutenere. Tornando alla preservazione, per fortuna in Italia abbiamo i siti come Ready64 e i vari progetti per le cassette da edicola di un tempo (ce n'è più di uno), che cercano e catalogano il materiale prodotto in Italia sia cartaceo sia digitale e lo preservano per i posteri. Iniziative davvero lodevoli!

Se parliamo di reverse enginnering entriamo nel mio campo di azione per eccellenza. Mi serve sempre per capire come è stato realizzato il codice che mostra quello che sto vedendo a video o quello che il SID sta generando! E ovviamente mi sono prodotto il tool personalizzato che oggi uso ogni giorno: «JC64dis». Il programma era nato inizialmente (prima in Pascal, poi riscritto in Java) perché i tool esistenti di disassemblaggio non mi soddisfacevano minimamente quindi avevo realizzato una mia versione a riga di comando che mi restituiva il sorgente con i commenti alle locazioni note. Ad un certo punto, constatato



Libro The real power of JAVA!

che perdevo molto tempo ad elaborare il testo generato per capire cosa realmente faceva il programma introducendo opportune labels, o traslando il codice in dati, ho deciso di renderlo interattivo creando una maschera divisa in tre aree: le 64K locazioni di memoria in cui il binario entra in memoria, il pseudo disassemblato in cui si mettono sia i valori di opcode/operandi che le istruzioni corrispondenti, e il disassemblato vero e proprio.

La scelta sembra strana, ma ha la sua (o mia) logica. Da un lato vedo sempre tutto a livello della macchina, dall'altro vedo la versione in assembler puro come l'avrebbe scritto il programmatore originale. La parte assembler, che io utilizzo sempre in formato Dasm, può essere vista anche in altri formati come Kick Assembler o 64Tass, giusto per citarne qualcuno. Adesso il programma è composto da più di 109.000 righe di codice, con quasi 300 opzioni di configurazione, e ha un costo di realizzazione di 27 persone/anno con un valore COCOMO superiore a 1.400.000$ (il che dice poco o niente, se non che vale la pena provarlo :)) Durante la realizzazione ho avuto il test di parecchi sceners che hanno contribuito a raffinarlo, tra cui Baccus, That8BitChiptuneGuy e Chris Abbott. In particolare Chris lo stava usando per realizzare il «Project Hubbard» per cui in quel periodo eseguivo le modifiche quasi in tempo reale per riuscire ad agevolarlo man mano che mi girava richieste di miglioramenti. Ultimamente sto aggiungendo il supporto AI al programma. Al momento è molto basico, ma considerando che prima o dopo avremo dei modelli linguistici espressamente tarati per il MOS 6502 e il C64, in futuro basterà premere un tasto per trovarsi il sorgente pressoché completo, tutto in automatico!

12. La sfida dell'emulazione: Comprendere a fondo un hardware per scriverci sopra è una cosa; riprodurlo in modo "cycle-exact" via software è un'altra. Quali sono gli ostacoli più complessi nell'emulare macchine del passato?

Ai tempi dell'Università mi era balenata l'idea di creare un emulatore del C64 quando nemmeno sapevo se ne esistevano (sfruttando internet nell'Università in seguito riusci a procurami comunque del materiale). Avevo visto il manuale di programmazione del MOS 6502 della Commodore in una libreria (un bel grosso volume). Ci tornai una settimana dopo per prenderlo ma era stato venduto (immagino fosse stato li da 10 anni e nessuno lo aveva toccato prima di allora…).

Per fortuna un amico aveva un altro libro di programmazione del C64 perciò passai un pomeriggio a fotocopiarne il contenuto per metà in una copisteria. C'era tutto quello che mi serviva, compreso lo schema circuitale del C64 (questo libro oggi è sempre a portata di mano nella scrivania assieme al manuale C64). Dopo aver realizzato

l'emulatore completamente in assembly x86 utilizzando le librerie Xlib per la grafica (per cui la sfruttavo per mostrare in sovraimpressione una tastiera virtuale coi tasti del C64 quando l'utente voleva sapere quale tasto premere per avere il codice PETSCII grafico), trovai il «Vic Article» di Christian Bauer. A quel punto capii che una cosa era il manuale Commodore che descriveva i vari chip a livello teorico, tutt'altra cosa era come funzionavano le cose nella realtà! Per questo riscrissi l'emulatore in Java, ma in realtà è un simulatore perché tutto è realizzato a livello di clock del VIC (che poi si ripercuote in quello della CPU): leggendo il codice sorgente si capisce quello che accade a livello hardware. Ovviamente nessun PC attuale potrebbe eseguire l'emulatore garantendo 1MHz di velocità perché la sincronizzazione dei thread a livello di ciclo di clock è troppo onerosa come tempi di esecuzione.

Tornando alla domanda, la difficoltà dell'emulazione è proprio quella di garantire che tutto giri al ciclo esatto in base alle attuali conoscenze. Se abbiamo un demo sincronizzato al singolo ciclo ed eseguendolo su un emulatore non otteniamo lo stesso risultato significa che la nostra conoscenza non è ancora completa del sistema da emulare! Manca qualcosa. Quel qualcosa potrebbe essere una interazione "non voluta/documentata" tra VIC/CIA/CPU e scheda madre. Per questo spesso si ipotizza la causa, poi si passa a verificare tramiti programmi o misurazioni elettriche se è effettivamente quello il caso e lo si implementa.
In tutto questo tralascio volutamente il SID: solo per il fatto che internamente ha anche componenti analogiche, rende la sua emulazione tutt'altro che completa e fedele anche oggi giorno. Si avvicina di molto, ma appena sentite il suono emesso dal C64 la differenza si percepisce!
Approfitto dello spazio e mi rivolgo alla nuova Commodore se è in ascolto: ci serve un nuovo chip SID (creato con tecnologia moderna) che abbia un suono pari all'originale. So che voi potete farcela!!

13. Tools e workflow: Che tipo di ambiente di sviluppo utilizzi oggi per i tuoi progetti «retrosoftware»? Preferisci strumenti originali su hardware d'epoca o cross-compilatori e tool moderni su piattaforma Win/OSX/Linux?
Non uso mai il C64 originale se non per testare il lavoro ultimato (adesso però con il C64 Ultimate utilizzo direttamente lui). Qualsiasi programma nativo per C64 lo uso in emulazione. Per disegnare gli sprite ho quasi sempre usato i programmi BASIC di Andrew Colin inseriti nel suo corso: semplici da usare e fanno bene quello per cui erano progettati (sono lenti, ma in VICE basta attivare il "warp mode"). Altre volte creo gli sprite in GIMP poi li converto usando il programma Spritepad. Per la musica uso GoatTracker su PC oppure il mio tracker JITT64 sempre su PC. Il tutto dipende da come e dove deve essere usata la musica.
JITT64 è nato per generare musica senza compromessi (ognuno dei 255 strumenti è indipendente dagli altri e la definizione di ciascuno può richiedere fino a 2KB di memoria, ogni tune può usare fino a 999 comandi per i pattern, con 223 pattern a lunghezza variabile), quindi è poco adatto per l'uso in contemporanea ad altro.
Per la programmazione uso Kate su KDE in Linux: editor testuale con supporto della sintassi colorata MOS6502, con la possibilità di salvare sessioni con tutti i files in uso. Per compilare uso la bash integrata di Kate da cui lancio il make (script bash con tutto quello che serve).
L'emulatore che uso è solo VICE (spesso poi ne modifico i sorgenti se mi serve ottenere dei log particolari che il monitor non permette). Come assemblatore mi affido a Dasm: semplice, veloce anche se un po' criptico nei messaggi di errore. A volte utilizzo XA per piccoli pezzi, come nella parte relativa alle cartridge. In passato avevo usato anche l'assemblatore della suite CC65, sia per provare a programmare direttamente in C che in assembly, ma poi sono tornato a Dasm.
Ultimamente sto usando parecchio «Turbo Rascal Syntax Error», l'IDE di sviluppo anche per C64 che si programma in Pascal. Il Pascal è semplice ma sempre chiaro da leggere anche per chi non conosce il linguaggio, inoltre TRSE ha parecchie librerie a corredo che gestiscono svariati dettagli (ad esempio visualizza le immagini Koala, suona i SID, ecc). Ovviamente permette di incorporare codice direttamente in assembly e questo lo rende perfetto (sembra di essere tornati al tempo del Turbo Pascal!). È chiaro che poi eventuali tool che mi servono li creo in Java. Ad esempio l'editor di livelli di LSS2 (che ho usato anche per altri progetti) è stato sviluppato per quel compito preciso.

14. Documentazione e conoscenza: Ai tempi d'oro della demoscene, i "segreti" dell'hardware si tramandavano spesso solo in circoli ristretti. Oggi che tutto è documentato online (lo è veramente?), è diventato più facile trovare informazioni e trucchi e per questo il coding ha perso un po' della sua magia?
Si, online si trovano spesso informazioni, ma ritrovare il giorno dopo le stesse informazioni può essere difficilissimo se non ce le siamo salvate, perché è tutto troppo dinamico: la stessa query di ricerca può portarci da tutt'altra parte, oppure il sito può essere stato chiuso. Inoltre le informazioni che si trovano vanno poi applicate e tra teoria e pratica non è così immediato. Quindi la magia rimane!

Ad esempio il mio vecchio demo Silent Night serviva per testare l'apertura totale dei bordi e visualizzare una immagine con gli sprite su di essi (il minimo che ogni demo non basico deve saper fare). È tutto molto semplice: basta leggere il Vic Article e c'è tutta la teoria da applicare. Ok, provateci e ne riparliamo. Dobbiamo sempre pensare in termini di cicli di istruzioni per ogni linea del VIC. E le linee sono 200 per la parte con la grafica, di cui una ogni otto è speciale (bad-line), che dobbiamo manipolare per tenere il bordo laterale aperto.

Nel sorgente, ad esempio, inserisco per ogni istruzione il ciclo di inizio e quello di fine in modo da conoscere in ogni momento quello che sta avvenendo. Certo è un conto che si può fare a mente, ma io preferisco lasciarlo visibile in modo che chiunque possa seguire quello che accade. Quindi, pensate a demo dove vengono applicate tecniche diverse tutte in contemporanea, ognuna delle quali ha dei requisiti da rispettare: sappiamo come applicare le singole tecniche, ma per il fatto di usarle assieme diventa qualcosa di totalmente nuovo (e di cui magari non abbiamo informazioni su come superare certi problemi di convivenza perché nessuno li ha documentati).

Credo che in futuro quanto l'AI sarà opportunamente addestrata sul C64, come accennavo prima parlando del disassemblatore, potrà essere fonte anche di coding avanzato molto valida, sopratutto perché dovrà commentare il lavoro nel dettaglio.

Io già oggi testo i vari motori AI cercando di verificare se realmente sono in grado di spiegare una routine di demo nei minimi dettagli. So la risposta che deve darmi e che fonti deve usare per arrivare alle conclusioni. Siamo ancora lontani, ma i progressi sono evidenti man mano che i modelli evolvono.

Parte 4: Visione, Cultura e Futuro

15. Patrimonio UNESCO: Come saprai la Demoscene è stata riconosciuta in diversi Paesi come patrimonio culturale immateriale. Che effetto ti fa vedere un movimento "underground" e spesso legato alla cultura hacker ottenere questo tipo di riconoscimento istituzionale?

Il riconoscimento dell'Unesco mi era sfuggito! Magari questo darà modo che venga finanziato in futuro la conservazione delle opere da parte di qualche governo aderente. Oggi ci affidiamo ai siti come CSDB, ma se per una malaugurata sorte dovesse sparire sarebbe un danno enorme.

Si potrebbero finanziare i demo party che si svolgono ogni anno, anche questo sarebbe un punto importante per la demoscene dato che spesso i demo sono creati con anni di lavoro proprio per presentarli in questi appuntamenti.

16. La scena italiana: Come hai vissuto l'evoluzione della demoscene in Italia nel corso dei decenni e quali sono le differenze più marcate, secondo te, rispetto al resto d'Europa?

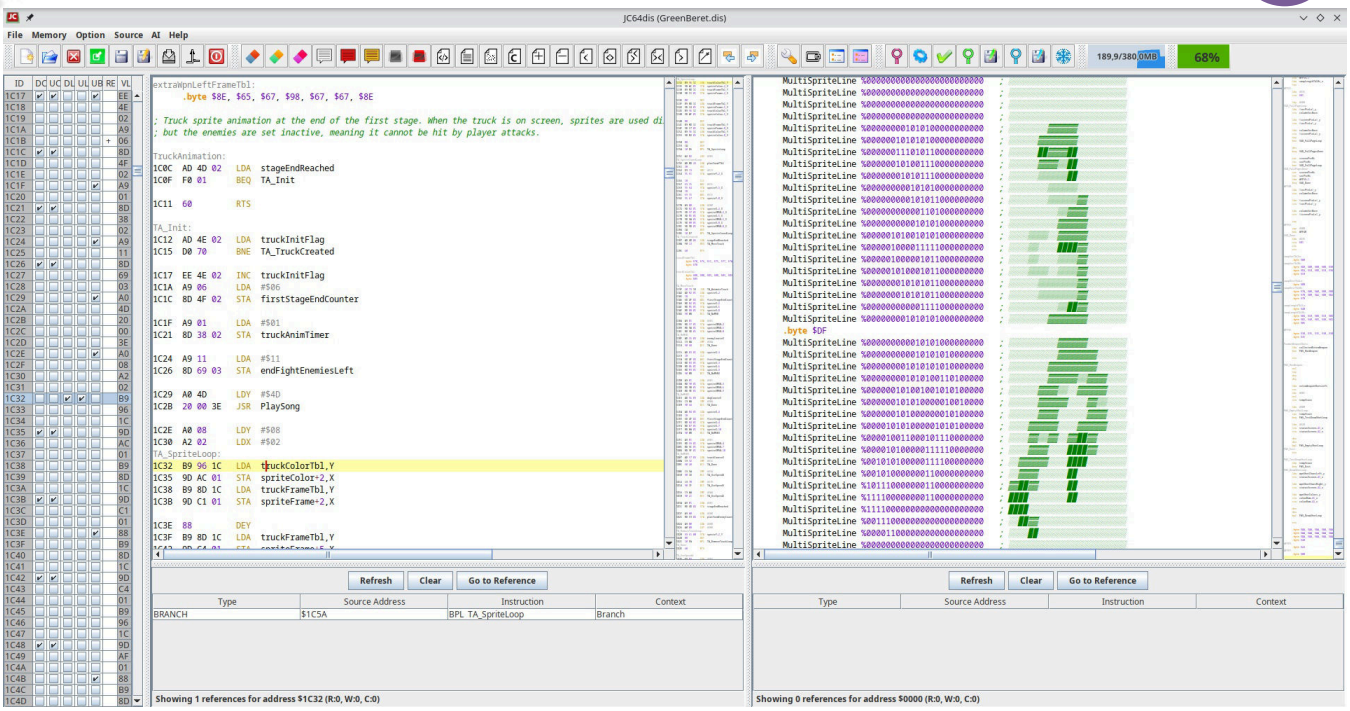
La scena Italiana all'inizio era legata parecchio alla pirateria, perché in mancanza di leggi adeguate, modificare un gioco, riprodurlo su cassetta e venderlo era decisamente più redditizio del aprire uno studio e produrre e vendere i propri giochi! Se paragoniamo il numero di software house inglesi con una lista di quelle italiane degli anni ottanta, potrei citare decine di ditte inglesi contro le sole due o tre che mi vengono in mente in Italia (Simulmondo, Genias, Idea,... anzi no quattro perché c'era anche Systems Editoriale) e che realizzavano giochi, però per lo più concentrati verso la fine degli anni 80 quando il mercato del C64 cominciava già a declinare.

Quindi avevamo diversi "crackers" all'avanguardia che sproteggevano e adattavano i giochi inglesi in pochi giorni per renderli subito disponibili nelle casette da edicola. Pensando poi ai mezzi che avevano disponibili all'epoca le loro imprese erano decisamente di altissimo livello. Se citiamo ad esempio il «North Italy Cracking Service», «Fantastic 4 Cracking Group» oppure l'«Italian Cracking Service» abbiamo in esse decine di membri di cui di molti purtroppo non conosciamo neppure l'identità e questo è un peccato storico che meriterebbe di essere recuperato prima che vada irrimediabilmente perso. Oggi invece la demoscene è diventata per lo più internazionale e non più nazionale come una volta: si coopera con chiunque condivida le nostre passioni semplicemente contattandosi su internet sfruttando i vari forum/canali/social network. Forse è proprio questo che sta dando una seconda vita al C64!

17. Nuove generazioni: Noti un interesse da parte dei giovani programmatori verso il codice di basso livello e l'hardware vintage oppure la demoscene sta diventando una nicchia per soli "veterani" nostalgici?

Io vedo che in seno al gruppo Hokuto, ad esempio, ci sono dei giovanissimi (non vorrei sbagliarmi ma di 16 o 17 anni) che si cimentano col C64, quindi credo che non sia solo per i veterani. Inoltre vedo che molti gruppi cominciano a rilasciare i sorgenti su Github dei loro demo, quindi la conoscenza di programmazione avanzata diventerà sempre più accessibile agli interessati. Non ultimo, la rinascita della nuova Commodore con il C64Ultimate potrebbe far conoscere con mano il C64 alle nuove generazioni. Staremo a vedere ;)

18. L'impatto sul lavoro quotidiano: In che modo



JC64dis in azione!

l'esperienza maturata nella demoscene ha influenzato la tua carriera professionale o la tua forma mentis nello sviluppo software di tutti i giorni?

Dovrei ringraziare indirettamente il BASIC per questo (il punto di partenza per tutti col C64): era talmente orribile (ma a quel tempo non lo sapevo) da costringerti ad imparare l'assembly e al contempo scoprire cos'era invece la vera programmazione tramite i linguaggi realmente strutturati! L'influenza che ho avuto è sempre stata quella di condividere le informazioni (infatti sono amante dell'open source): cosa sarebbe successo se Christian Bauer non avesse scritto il Vic Article? Avrei scoperto magari 10 anni dopo come funzionava realmente il Vic: una marea di tempo sprecato. Quindi la condivisione è la base di tutto. La demoscene vive di informazioni (sull'hardware) e voglia di spingersi al limite per andare oltre.

19. Progetti futuri: C'è un progetto, una demo o un pezzo di software che hai ancora nel cassetto e che vorresti assolutamente completare prima o poi?

Qui ho parecchi progetti in piedi…

1. Il porting di «Frozen Bubble» per C64, progetto in collaborazione anche con Hokuto Monastery. Sicuramente si potrebbe considerare materiale per Games That Weren't visto che il progetto è di 20 anni fa, ma attende sempre di essere ultimato… Purtroppo, mentre lo realizzavo, ad un certo punto ho dovuto rifare l'interfaccia grafica da capo andando ad aprire i bordi superiore ed inferiore per inserire sprite sia sopra che sotto per dare più spazio alle caselle in altezza per essere alla pari con la versione PC. Questo ha comportato un ritardo di mesi e nel frattempo era stato rilasciato «Puzzle Bubble» per C64 , quindi non avevo più urgenza di lavorare all'engine (dove stavo implementando la gestione collisioni) dato che il gioco rilasciato ha lo stesso funzionamento di «Frozen Bubble». In ogni caso sono già presenti ben 120 livelli!

2. Creare il seguito di Abyss. Sono sempre stato affascinato da questo gioco, per cui ho realizzato un editor di mappe in Java ed ottenuto il sorgente per reverse engineering (75% completato). Mi basta solo integrare l'editor di livelli con il sorgente e ottenere un eseguibile. A quel punto creando nuove mappe si può rilasciare il gioco. Il fine di questa iniziativa è quella di riuscire a contattare i tre autori, di cui purtroppo si sono perse le tracce: magari il rilascio non ufficiale del gioco può smuovere le acque… Per i lettori che stanno leggendo, contattatemi se volete creare le mappe del gioco. L'editor è perfettamente funzionante, serve solo qualche appassionato che voglia contribuire per completare il prima possibile l'opera.

3. Continuare con «Little Sara Sister 3». Il gioco non è per C64 ma è il porting su PC/Smartphone della versione C64 utilizzando un engine realizzato in Java tramite libGDX di cui avevo abbozzato già la grafica di alcuni livelli. Non partivo da zero, perché avevo già realizzato anni fa il porting da C64 di «P0 Snake» di Antonio Savona, sempre in Java, nel gioco «T8 Snake» in cui oltre ai livelli originali (grazie al codice datomi da Antonio ricreati con precisione) avevo aggiunto nuovi livelli sfruttando la potenza del PC.

4. Un demo su Lara Croft in lavorazione anch'esso da oltre 5 anni. L'idea era di fare il classico demo oldschool, per esempio in stile Trap, in cui si racconta in modo quasi "goliardico" una storia su Lara, ma volevo colpire lo

spettatore andando a mostrare immagini che non era possibile realizzare negli anni 80. Ecco dunque che serve ricorrere alla cartridge per contenere il tutto (siamo vicini ai 512KB). Le musiche sono di Saul Cross, quindi so che il demo sarà visto almeno per ascoltarne le musiche :) L'integrazione delle musiche mentre visualizzo le immagini ad alta risoluzione è al momento il punto che sta richiedendo di adattare il codice, altrimenti il demo senza musiche durante quella fase sarebbe già completo (anche se sto pensando di aggiungere ancora qualcosa...). In ogni caso il demo (realizzato con TRSE) sotto sotto ha una morale nella storia narrata, ma non voglio spoilerare troppo...

5. Una cartridge che mostra tutti i formati grafici supportati dal C64 e li mostra in sequenza: CGMC 64 (Commodore Graphics Mode Cartridge). Questo nasce per caso. Dovendo inserire la musica nel demo di Lara mentre visualizzo le immagini "speciali", avevo bisogno di capire come funzionava il player per modificarlo, per cui avevo iniziato ad analizzarne diversi. A quel punto avevo chiesto internamente al gruppo Hokuto se un demo su cartridge che mostrasse tali formati in sequenza potesse essere interessante. Vista la riposta positiva ho abbozzato un demo di test in cui ho dovuto risolvere parecchi problemi (come, mentre suona la musica, leggere dalla ROM, decomprimere al volo e salvare in RAM in modo da poter accedere a tutti i 64KB durante l'operazione, e infine attivare l'immagine sempre con la musica attiva) ma alla fine il demo ha preso piede tanto che è stato preannunciato tramite video dal canale «Mille ed una Avventura» dal professor Boselli che ringrazio. Anche questa cartuccia è realizzata con TRSE in cui inserisco in assembler le parti dei metodi che estraggo disassemblandoli con JC64dis.

6. Un libro sui formati grafici del C64: "Pixel & Cycles: The Art of Vic-II Graphics". Data la realizzazione del demo CGMC, mi son chiesto perché non fare un libro con le conoscenze acquisite sulla realizzazione dei metodi stessi? Perché non è facile conoscere come un player effettivamente applica i trucchi sul Vic per realizzarlo, pertanto documentarlo diventa un modo stesso per mostrare la bravura dei coder che sono riusciti nell'impresa. Questo è un libro totalmente diverso dal tomo di 850 pagine su Java (che è realizzato tramite Latex essendo pieno di formule matematiche, algoritmi, grafici e tabelle) perché deve essere sia tecnico (per far capire cosa avviene) che artistico (mostrando il risultato su carta di quello che si vede a schermo). Per questo il libro è composto con Scribus in modo da avere padronanza di ogni posizionamento del testo e delle immagini, le quali sono convertire in CMYK per una riproduzione fedele dei colori (avendo prodotto in passato la rivista "Regame 64" su carta con questo stesso metodo sono sicuro della resa tipografica finale). Verosimilmente sarà composto da circa 170 pagine (cerco di averne il meno possibile perché le pagine a colori inevitabilmente alzano il costo di produzione), ma certi grafici contenuti, anche se portano via spazio, sono essenziali per capire il funzionamento del metodo sottostante. Se ll libro su Java ha richiesto 5 anni di lavoro, questo dovrebbe completarsi in circa 1 anno quindi spero che per Natale prossimo sia già disponibile. A corredo ci saranno i sorgenti in modo da sperimentare con mano i metodi stessi descritti nel libro.

7. Il crossover tra HVMEC e SIDin. In pratica sto scrivendo un altro libro (PDF) che è in stile SIDin ma presenta gli editor presenti in HVMEC dandogli la collocazione storica. Ma non mi limito solo agli editor esistenti, prendo degli autori che hanno creato e usato i loro player e li analizzo (riportando anche dei loro commenti se è possibile rintracciare l'autore). Ne rimane così un contributo storico di come si è evoluto nel tempo la programmazione del SID. Ovviamente questo è un progetto a lunga gittata e al momento è parecchio in standby avendo gli ultimi tre progetti elencati sopra che hanno la priorità.

20. Un consiglio ai coder di oggi: Se un giovane sviluppatore ti chiedesse da dove iniziare per capire davvero come funziona un computer "sotto il cofano", cosa gli consiglieresti di studiare o procurarsi?

Probabilmente il passo base sarebbe quello di partire da micro controllori come Arduino o Rasberry PI Pico: qui si impara tutto agendo a basso livello ma programmando ad alto livello e sperimentando anche in assembly. L'hardware è limitato perché è tutto contenuto all'interno del chip ma è rappresentativo di quello che si trova più in grande nei computer.

Il successivo step è passare ad un computer completo, di per se non è importante quale, io consiglierei un sistema ARM (avendo un processore RISC le istruzioni sono più semplici da imparare se si usa l'assembly) come Raspberry Pi/Orange Pi/Banana PI/Pico, ma installandoci Linux come sistema operativo. Con Linux si ha modo di capire cosa fa realmente un sistema operativo, si può sperimentare e si capisce come funzionano realmente le cose. In ogni caso consiglio loro anche di procurarsi dei libri. So che su internet si trova guide su tutto, ma il libro è focalizzato sull'argomento specifico e permette di imparare le nozioni che servono gradualmente. E se serve rivederle le sia ha subito a portata di mano. Mi ricordo che ad uno SMAU negli anni 90 comperai il Turbo Pascal 6.0 (versione studente) dallo stand Borland. Era pieno zeppo di manuali. Preziosissimi. Ad esempio il manuale del Turbo Assembler era una guida completa al processore x86 e mi ha permesso di imparalo in un niente.



Editor Abyss

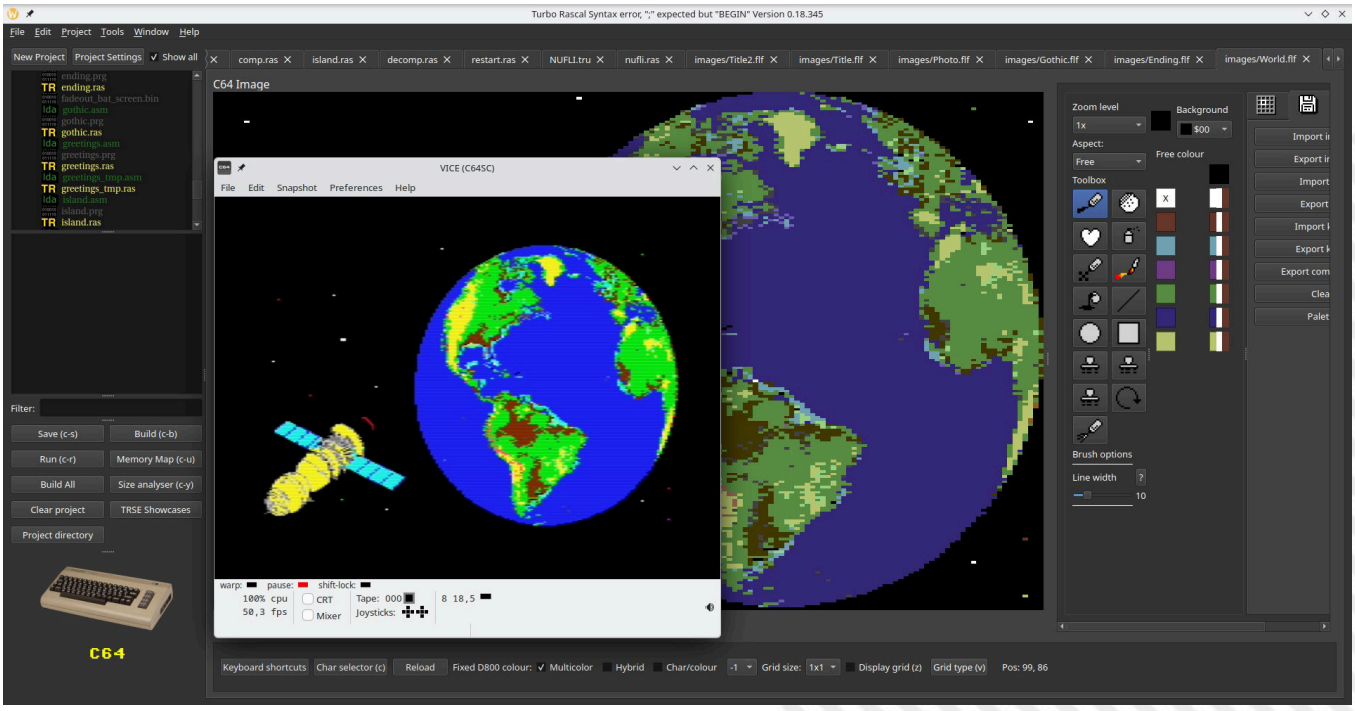
Outro

Siamo giunti alla fine di questa interessantissima intervista. Ringraziamo Stefano per l'impegno e per il tempo che ha dedicato a rispondere alle nostre domande. Abbiamo scoperto e imparato molto leggendo le sue risposte, che, come avete avuto modo di vedere, sono state ampie ed eusarienti. Abbiamo potuto apprezzare pienamente la filosofia di condivisione di informazioni e, soprattutto, di esperienze pratiche che si trova alla base dell'attitudine del nostro Ice00 verso il mondo dell'informatica e del retrocomputing. E siamo davvero lieti che abbia voluto partecipare con tanta chiarezza e devozione alla nostra richiesta per un'intervista un po' diversa dal solito.

Riferimenti e link

Sito produzioni: https://iceteam.itch.io/
Sito storico: https://iceteam.altervista.org
Profilo csdb: https://csdb.dk/scener/?id=8082
Manuali C=: https://pickledlightprojects.com/
Ready64: https://ready64.org/

Screenshot TRSE con mock up demo Lara Croft



Quando nel 1972 Rolling Stone diede il via alla competizione: alle origini degli eSports Conversazione con Bruce G. Baumgart

di Filippo Occulto

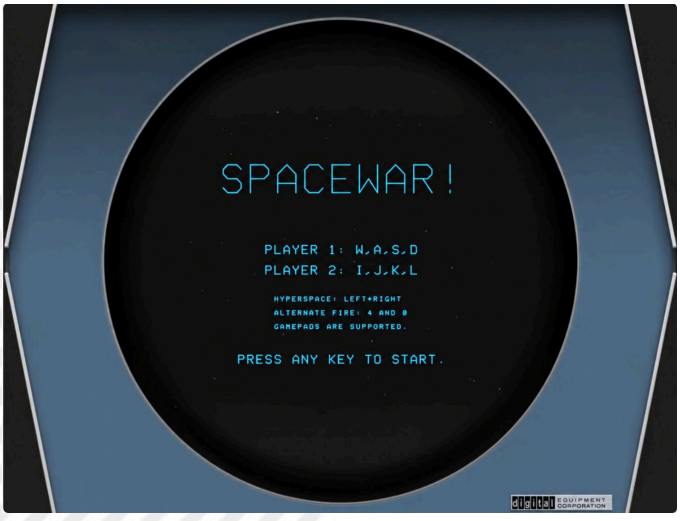
#54 anni dopo la storica competizione di Spacewar! a Stanford

Lo ammetto: sono euforico. Molto più di quanto possa sembrare leggendo queste righe.

Non capita tutti i giorni di avere l'opportunità di parlare direttamente con uno dei protagonisti dell'informatica delle origini, una di quelle persone che, a Stanford, già negli anni Sessanta, non si limitarono a osservare il futuro: contribuirono attivamente a costruirlo.

Per questo motivo sono davvero felice di annunciare che **Bruce Guenther Baumgart, *BGB*** per gli amici e per la storia dell'informatica a Stanford, è ospite delle nostre pagine.

Il suo nome ci riporta inevitabilmente a uno degli episodi più affascinanti della storia dei videogiochi: le celebri **Olimpiadi di Spacewar! del 1972,** una delle primissime competizioni videoludiche documentate delle quali possediamo ancora un resoconto preciso.



SpaceWar! sul DEC PDP-1 di Steve Russell

Molto è stato scritto su quell'evento, sia online sia sulla stampa dell'epoca, soprattutto grazie al famoso articolo pubblicato da Rolling Stone. Ma, come spesso accade quando la storia viene raccontata e ricopiata per decenni, anche un dettaglio apparentemente banale può finire per essere tramandato in modo errato.



Bruce Baumgart - Camp Navarro California
Bruce mostra la sfera blu di Carl Sagan che simboleggia il globo terrestre

La competizione, infatti, non si svolse il 19 ottobre 1972, come viene ancora riportato in diverse ricostruzioni, ma **mercoledì 18 ottobre 1972.**

È stato Bruce stesso a farmelo notare. E la verifica più semplice è anche la più efficace: il 19 ottobre 1972 era un giovedì, non il mercoledì in cui ebbe luogo la competizione.

Mi piace, tuttavia, immaginare quell'evento non soltanto come una data nella cronologia della storia dell'informatica, ma come qualcosa che accadeva nel mondo reale di quel tempo.

Mentre a Stanford un gruppo di studenti e ricercatori si sfidava davanti a Spacewar!, le radio americane trasmettevano "**My Ding-A-Ling"** di Chuck Berry, mentre nelle sale cinematografiche proseguiva lo straordinario successo de "**Il Padrino"** di Francis Ford Coppola.

Insomma, mentre la cultura popolare attraversava uno dei suoi momenti più vivaci, in un laboratorio universitario della California si stava facendo la storia dei videogiochi, quasi senza che nessuno fosse ancora in grado di



SPACEWAR

THE FIRST "INTERGALACTIC SPACEWAR OLYMPICS" WILL BE HELD HERE, WEDNESDAY 19 OCTOBER, 2000 HOURS. FIRST PRIZE WILL BE A YEAR'S SUBSCRIPTION TO "ROLLING STONE". THE GALA EVENT WILL BE REPORTED BY STONE SPORTS REPORTER STEWART BRAND & PHOTOGRAPHED BY ANNIE LIEBOWITZ. FREE BEER!

READY OR not, computers are coming to the people.

That's good news, maybe the best since psychedelics. It's way off the track of the "Computers—Threat or Menace?" school of liberal criticism but surprisingly in line with the romantic fantasies of the forefathers of the science such as Norbert Wiener, Warren McCulloch, J.C.R. Licklider, John von Neumann and Vannevar Bush.

The trend owes its health to an odd

Click: 5 (Five players. This is for the first familiarization battles in the Spacewar Olympics, initiated by me and sponsored (beer & prizes) by ROLLING STONE.–Friends, I won't be able to explain every computer-technical term that comes by. Fortunately you don't need them to get the gist of what's happening.)

KEYBOARD BUTTONS? (ELSE REGULAR). TYPE Y OR N.

"Yes." Click: Y

THE STANDARD GAME IS:

SpaceWar! L'articolo della rivista Rolling Stones

comprenderne pienamente il significato: uno dei primi capitoli del futuro degli eSport.

E Bruce Baumgart era lì.

Non soltanto come spettatore, ma come partecipante e trionfatore.

BGB fu infatti uno dei protagonisti dello **Stanford Artificial Intelligence Laboratory**, il leggendario SAIL, un ambiente straordinario nel quale dominavano l'informatica, l'intelligenza artificiale, la computer grafica, la robotica e la sperimentazione.

Il suo lavoro a Stanford andava ben oltre i videogiochi: Baumgart era uno sviluppatore, un ricercatore e una mente attiva in numerosi progetti avanzati dell'epoca.

Ma il suo nome è legato anche a *Spacewar!*, uno dei titoli più importanti e influenti della storia delle origini del videogioco, capace di lasciare un segno profondo in molte produzioni successive di ambientazione spaziale e fantascientifica.

E soprattutto Bruce partecipò direttamente a quella competizione del 1972.

Cinquantaquattro anni dopo, poter tornare a quei giorni attraverso i ricordi di qualcuno che li ha realmente vissuti significa avere l'opportunità, almeno per qualche pagina, di passare dal racconto della storia alla testimonianza diretta.

Ed è proprio da qui che voglio ricominciare.

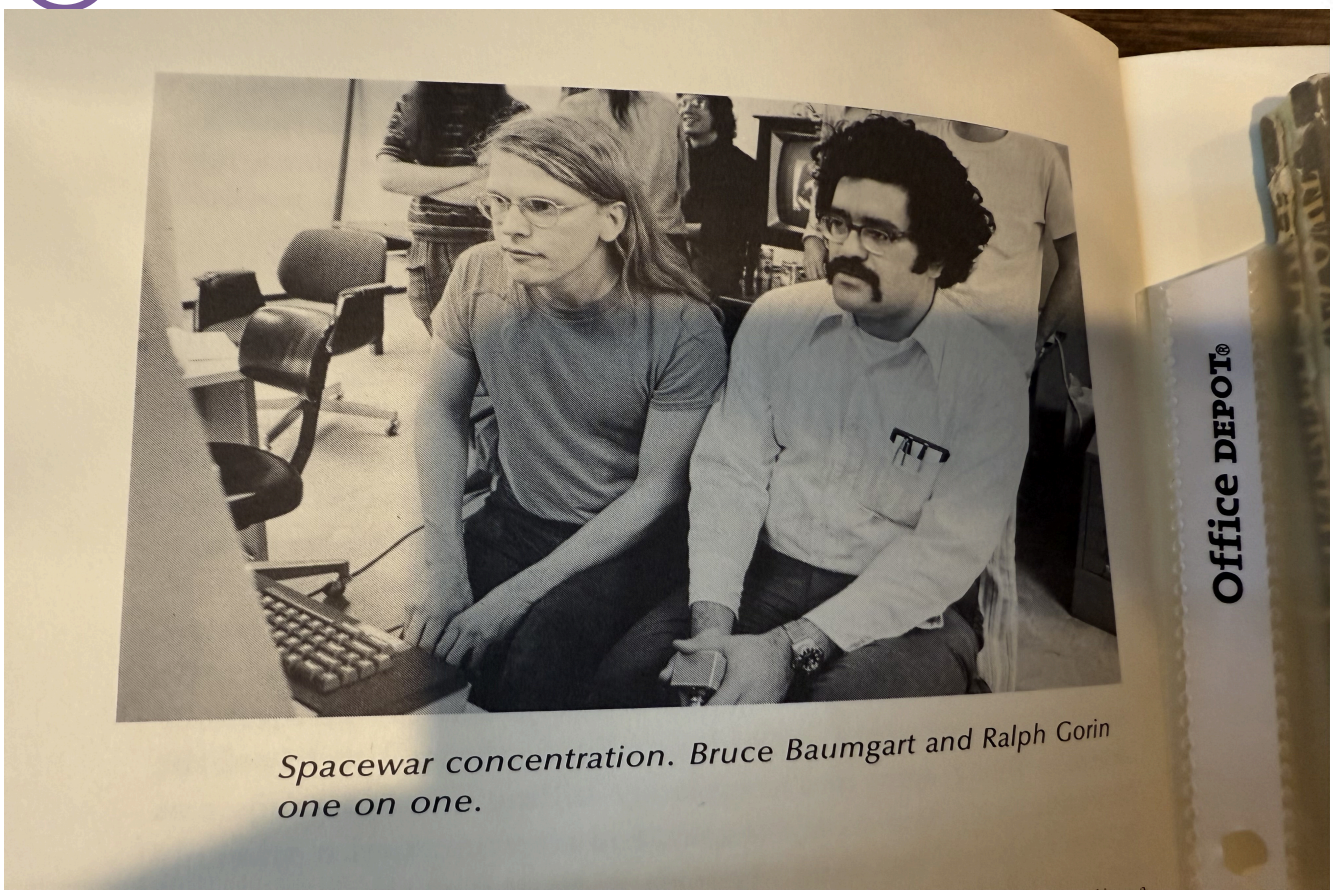
Bruce ammette candidamente che, nel 2024, ha ripreso a lavorare sulla computer grafica tridimensionale, creando un modello storico del **D.C. Power Lab** (l'edificio che ospitava il laboratorio SAIL), e ha iniziato seriamente a imparare a utilizzare una macchina Brother per il ricamo e il cucito.

F.O.: Bruce, innanzitutto, grazie davvero per aver accettato di rispondere alle mie domande.

Nel 1972 ero appena nato, ma è come se avessi vissuto attivamente quel periodo, respirando l'atmosfera di quei tempi memorabili, da bambino curioso e attraverso i racconti che ho potuto leggere nel corso degli anni. Tra questi ci sono i tuoi e quelli dei tuoi compagni di avventura di quel periodo.

Poiché mi piace dedicarmi alla ricerca storica e alla ricostruzione di fatti ed eventi che altrimenti rischierebbero di essere dimenticati con il passare del tempo, ho pensato di chiederti di raccontarmi aneddoti, episodi e ricordi che hai vissuto in prima persona.

Cominciamo quindi dall'inizio. Oggi sei in pensione ed



Bruce Baumgart (sin.) and Raplh Gorin (des.) (ritaglio del documento originale)

è passato moltissimo tempo da allora: come sei entrato inizialmente in contatto con Stanford e con lo Stanford Artificial Intelligence Laboratory? E che cosa ti ha portato lì?

BGB: Mi sono laureato a Harvard nel 1968. Il mio relatore di tesi, Ivan Sutherland, mi suggerì di presentare domanda a Stanford per proseguire gli studi.

Ad Harvard, durante l'estate, avevo lavorato con il professor Al Brenner del Dipartimento di Fisica, occupandomi della manutenzione e del miglioramento del PDP-1 al Dunbar Lab, accanto al Cambridge Electron Accelerator.

Così, quando arrivai al Dipartimento di Informatica di Stanford, nel settembre del 1968, fui assegnato ad assistere il professor Tony Hearn, fisico dello SLAC.

Nel gennaio del 1969, durante il mio secondo trimestre, Hearn e io ci trasferimmo entrambi al SAIL per lavorare su LISP e sui diagrammi di Feynman.

Allo SLAC, l'unico lavoro informatico interessante che svolsi fu aggiungere due o tre istruzioni macchina all'implementazione di LISP, in modo che potesse funzionare sul nuovo supercomputer dello SLAC, l'IBM 60/91.

Questo computer disponeva di un sistema unico nel suo genere di esecuzione anticipata e fuori ordine delle istruzioni, che non riusciva a richiamare il *garbage collector* quando l'istruzione CONS di LISP arrivava all'esaurimento dello spazio di memoria libero.

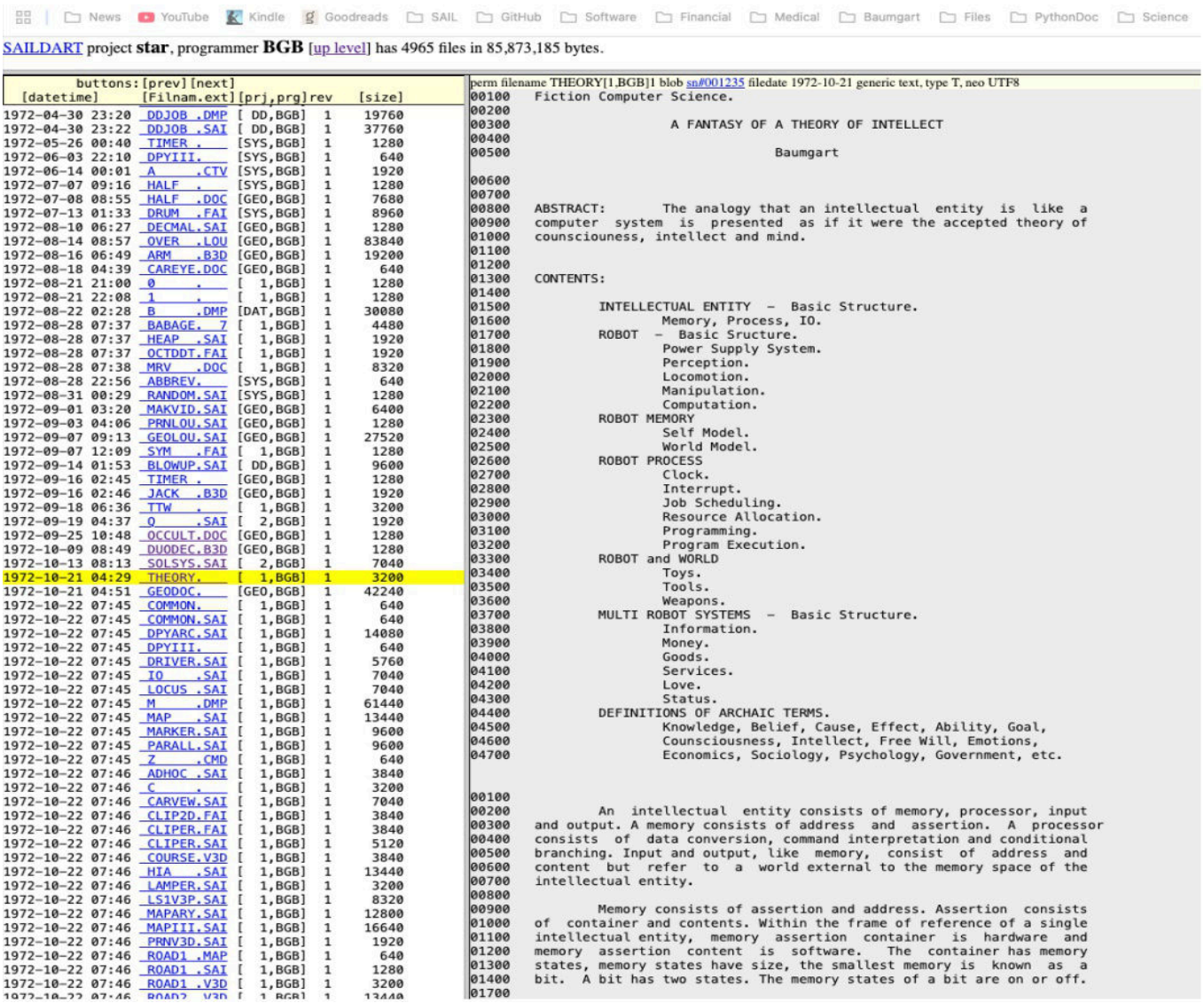
F.O.: A che cosa stavi lavorando esattamente al SAIL intorno al 1972?

BGB: Guarda i miei file risalenti al periodo intorno a mercoledì 18 ottobre 1972.

https://SAILDART.org/[star,BGB]

Cinque giorni prima compare un piccolo programma dimostrativo del Sistema Solare, mentre tre giorni dopo compare un file chiamato THEORY[1,BGB].

Inquietante: quella vecchia nota THEORY coincide con ciò che ho ripreso a fare nelle ultime settimane, ossia rappresentare le mie opinioni metafisiche sulla coscienza, sull'intelletto, sulla mente e sugli attuali strumenti di intelligenza artificiale del 2026.



THEORY - Intelletto, coscienza e mente

Gli altri file mostrano che stavo lavorando all'argomento della mia tesi, la modellazione geometrica per la visione artificiale, che sarebbe stata pubblicata alla fine del 1974.

F.O.: Come descriveresti il SAIL a qualcuno che non ha mai vissuto l'esperienza di un laboratorio informatico di quell'epoca?

BGB: Innanzitutto, semplicemente, «descrivere il SAIL»: ecco l'edificio del DC Power Lab, nella campagna californiana vicino alla Stanford University.

In secondo luogo, descrivere gli aspetti sociali del SAIL: la controcultura, gli hippie, i geek, l'ambiente tecnico e accademico, il centro di ricerca dell'ARPA durante la Guerra Fredda, insieme agli aspetti legati alla MFU, la Midpeninsula Free University.

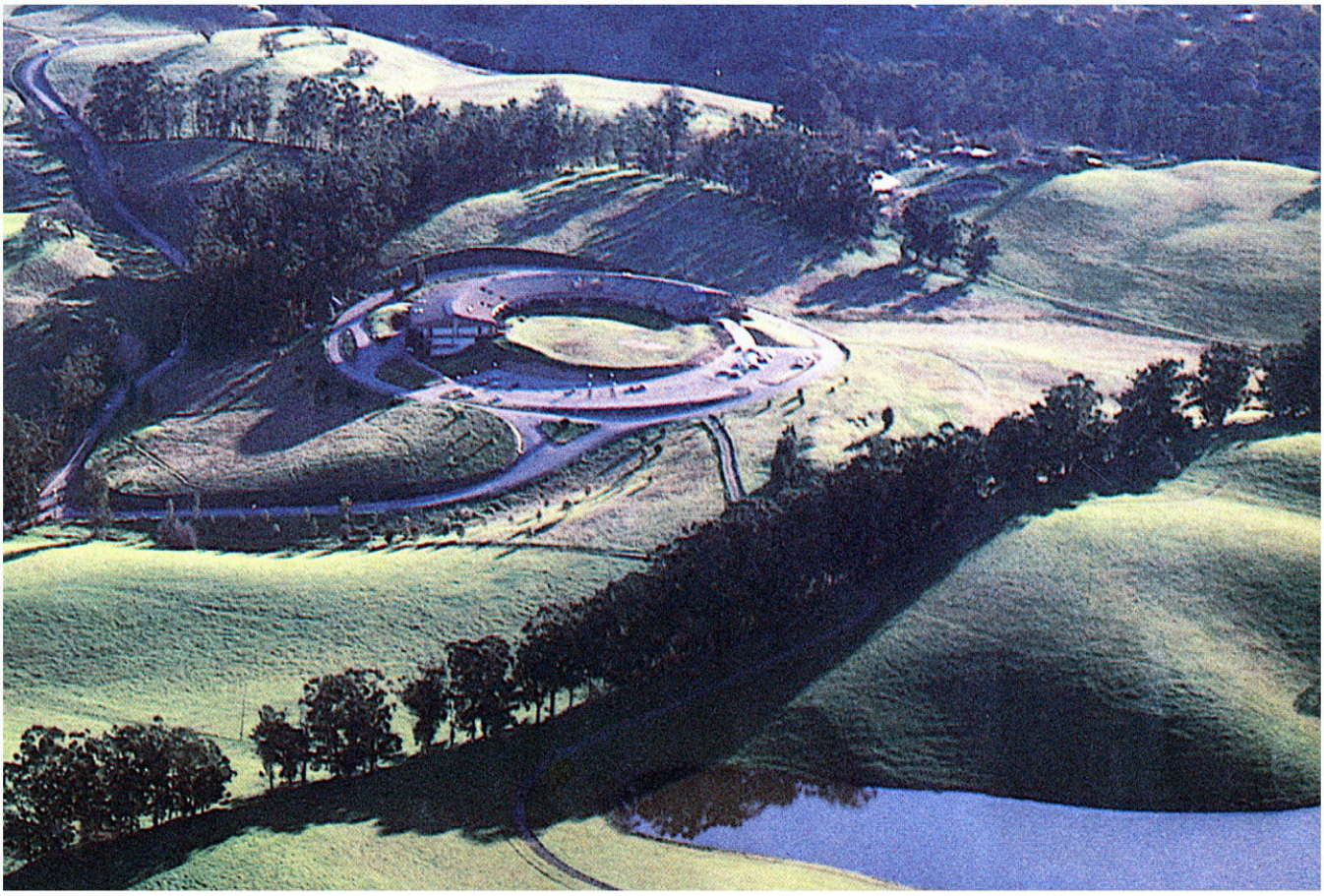
Infine, ignorare la premessa della domanda e descriverlo attraverso il confronto con gli altri laboratori informatici dell'epoca.

Le differenze più comuni comprendevano: time-sharing contro elaborazione batch, DEC contro IBM, interfaccia grafica contro schede perforate, archiviazione comunitaria dei file su disco online contro biblioteche comunitarie di nastri offline.

L'avvento del personal computer sarebbe arrivato subito dopo il 1972.

Erano disponibili penne ottiche, joystick e trackball (anche trackball capovolte, che in seguito sarebbero state chiamate mouse), oltre a tavolette e bacchette elettroniche.

Nessuno di questi strumenti prototipali per il posizionamento del testo venne adottato in un sistema generale di interfaccia grafica prima del 1972, con possibili eccezioni rivendicate da Xerox PARC, Engelbart allo SRI e



SAIL - DC Power Lab

dall'Università dello Utah.

F.O.: Esisteva una chiara distinzione tra professori, studenti, programmatori e tecnici hardware, oppure la comunità era più informale di quanto questi ruoli possano suggerire?

BGB: La comunità del SAIL era informale.

C'erano alcuni dipendenti tecnici di Stanford che non riuscivano a integrarsi e che se ne andarono; docenti e studenti erano più capaci di adattarsi rapidamente oppure di trasferirsi altrove.

Nella comunità hacker esiste una gerarchia sociale: i *Wizards* hanno un rango superiore ai *Newbies*.

Maltrattare i *Losers* e i *Bag Biters* era un'abitudine del MIT che, sfortunatamente, venne adottata anche da alcuni membri del SAIL.

F.O.: Com'era una notte tipica al SAIL?

BGB: Ho lavorato durante il turno notturno dal 1969 al 1974, praticamente dalle nove di sera alle nove del mattino.

Vorrei rispondere a questa domanda in modo approfondito, ma ho bisogno di esaminare SAILDART per ricostruire chi faceva che cosa durante quelle ore, in quegli anni.

F.O.: A parte Spacewar!, che cosa facevate concretamente per divertirvi nel laboratorio?

BGB: La ricerca era il divertimento del laboratorio!

Tra le altre attività ricreative, non legate al lavoro, che mi piacevano c'era pallavolo e il poker con piccole puntate, oltre al semplice passeggiare per l'edificio e i suoi dintorni, a volte per riflettere profondamente, altre volte per fare esercizio fisico.

Spacewar! venne eclissato dai chatbot: PARRY e ADVENT. Zork aveva invece origine al MIT.

Bisogna menzionare anche ELIZA e DOCTOR, oltre agli scacchi e al Go.

Persino FORTUNE COOKIE e "PORN", chiamato steganograficamente "PRNO", sono chatbot.

Codice sorgente di PRNO:

https://www.saildart.org/prno.mac[sw,reg]

F.O.: Al SAIL esisteva già una riconoscibile «cultura

hacker», anche se la parola aveva un significato piuttosto diverso da quello che possiede oggi?

BGB: Il SAIL e il laboratorio di intelligenza artificiale del MIT degli anni Sessanta e Settanta sono la cultura hacker. Stephen Levy ha scritto il libro Hackers.

Stewart Brand organizzò la prima Hackers Conference nel 1984, a Fort Cronkite, nella contea di Marin.

Il significato del termine «hacker» venne cambiato dal giornalista della CBS Dan Rather negli anni Ottanta.

F.O.: Quali persone del SAIL ti impressionarono maggiormente dal punto di vista tecnico, e perché?

BGB:

Hans Moravec

Dave Poole

Phil Petit

Andy Moorer

Vic Scheinman

Don Knuth

Lester Earnest

John McCarthy

Tom Gafford

Rispondere dettagliatamente richiederebbe più tempo.

F.O.: C'erano persone estremamente importanti all'interno del SAIL che oggi sono quasi assenti dalle ricostruzioni storiche dell'informatica?

BGB: Lester Earnest riteneva di non aver ricevuto un adeguato riconoscimento storico.

Ho una trentina di scatoloni contenenti i suoi documenti, da consegnare ai consueti archivi storici: Internet Archive, Stanford Green Library e Computer History Museum.

F.O.: Quanto erano competitivi tra loro i programmatori e i ricercatori, dal punto di vista intellettuale, tecnico o semplicemente per divertimento?

BGB: È difficile per me dirlo, perché ero troppo vicino all'élite interna.

La competizione per l'assegnazione dello spazio su disco è ben documentata e noi vincitori disponevamo ciascuno di un pacco disco utente UDP aggiuntivo da riempire.

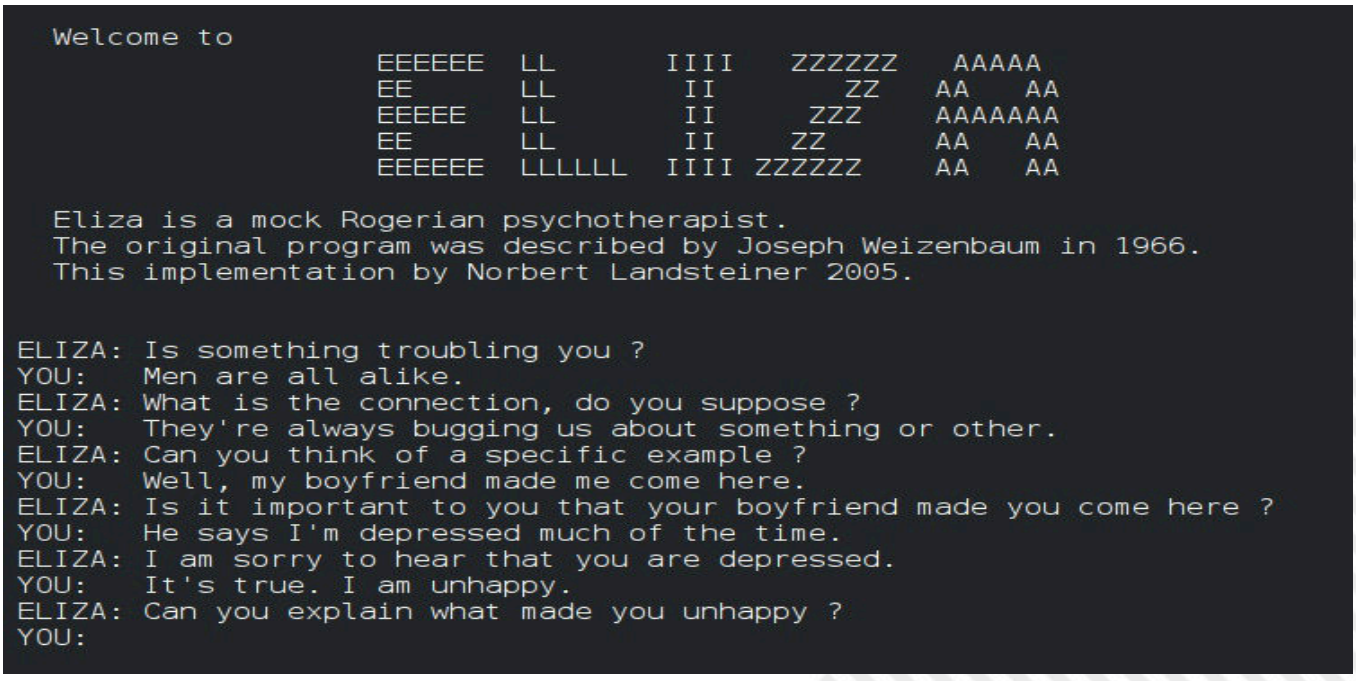
La competizione per il tempo di utilizzo del computer era estremamente dura per coloro che lavoravano durante gli orari regolari.

I battibecchi e l'assegnazione delle risorse compaiono in SAILDART.

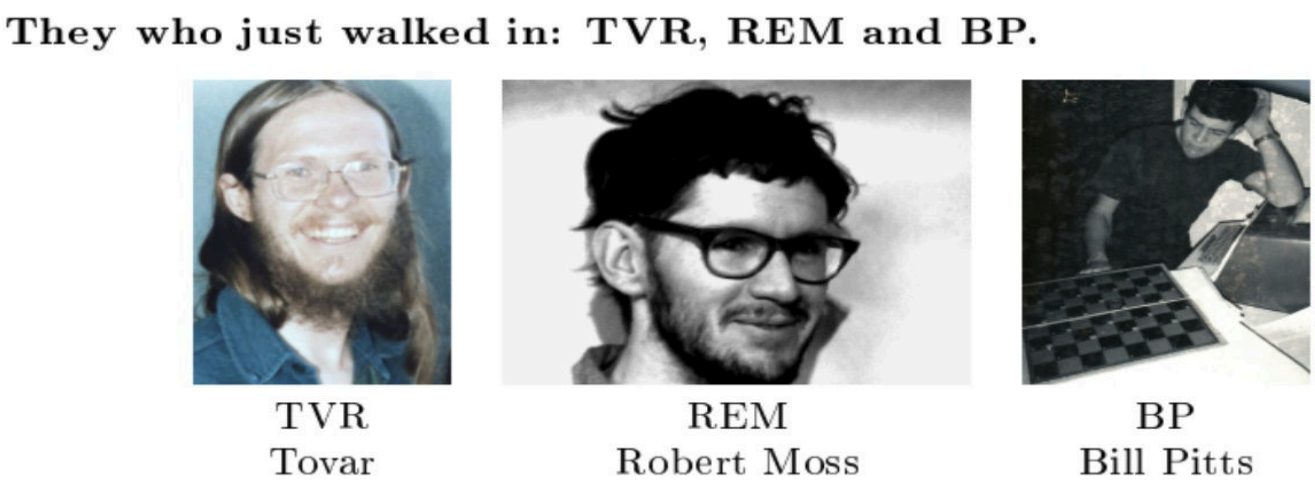
F.O.: Era normale modificare programmi scritti da altre persone?

BGB: Sì.

La manutenzione e le estensioni di PUB divennero un'attività aperta a tutti dopo che Larry Tesler si trasferì allo Xerox PARC.



Eliza - Il primo chatbot



Anche il lavoro sui caratteri tipografici per la XGP (Xerox Graphics Printer) richiese numerosi volontari.

Guarda tutti i creatori di font in SAILON-74.

- Collegamento Find-a-Font alla versione digitalizzata tramite scansione.

- Collegamento Find-a-Font alla versione originariamente digitale.

F.O.: Naturalmente conosciamo la collaborazione attorno a Spacewar!, ma più in generale come veniva considerata la paternità del software in quell'ambiente?

Nel caso specifico di Spacewar!, era normale che un giocatore interessato potesse leggere e modificare direttamente il codice del gioco?

BGB: Sì.

Guardando proprio adesso, nel 2026, l'elenco dei miei lavori del 1972, vedo che RHT aggiunse i danni parziali alla versione REG.

F.O.: Qual era il tuo rapporto con Ralph Gorin?

BGB: Eravamo colleghi al SAIL, avremmo avuto un rapporto più introspettivo successivamente.

F.O.: Che cosa ricordi di Slim Tovar e Robert E. Maas come membri della comunità del SAIL, al di là della loro partecipazione al torneo?

BGB: Elimina «Slim» da «Slim Tovar»: è semplicemente un aggettivo usato da Brand, non un nome.

Tovar è lo pseudonimo controculturale di una persona il cui nome anagrafico è John Mock.

Ulteriori informazioni biografiche più avanti.

F.O.: Se potessi accompagnare uno storico all'interno del SAIL durante una normale serata del 1972, che cosa pensi lo sorprenderebbe maggiormente?

BGB: Niente telefoni cellulari. Niente GPS. Niente schermi piatti.

Un solo televisore a colori collegato al computer.

Il prezzo di una Coca-Cola: quindici centesimi.

Il distributore automatico del Prancing Pony.

Prezzi del menu McDonald's negli anni Settanta:

- Hamburger: da 18 a 20 centesimi.
- Cheeseburger: 25 centesimi.
- Big Mac: 49 centesimi.
- Patatine fritte: 20 centesimi.
- Frappè: 25 centesimi.

Il Big Mac, che costava 50 centesimi (introdotto nel 1968), oggi costa 8 dollari a San Francisco.

Ma la vera, incredibile SORPRESA si chiama Legge di Moore.

Guarda Wikipedia:

https://en.wikipedia.org/wiki/Moore's_law

Passare da mille transistor nel 1970 a cinquanta miliardi di transistor nel 2020 è molto difficile da comprendere, anche se l'ho vissuto personalmente.

F.O.: Hai ricordato di aver incontrato Spacewar! sul PDP-1 di Harvard nel 1966, mentre lavoravi al Dunbar Lab, e che inizialmente il gioco non ti aveva particolarmente impressionato.

Che cosa ricordi di quella prima esperienza?

BGB: Il nastro di carta perforata di Spacewar!, ripiegato a fisarmonica, era scomodo da caricare.

La mappa stellare di sfondo, basata sul mondo reale, mi sembrava più interessante della fisica gravitazionale bidimensionale.

Di solito ero l'unica persona presente nell'edificio.

Spacewar è un gioco per due persone.

Notai che i siluri fotonici erano influenzati dalla gravità.

F.O.: E che cosa cambiò successivamente, rendendo Spacewar! così interessante per te durante gli anni a Stanford?

BGB: C'erano più persone con cui giocare.

Se viene giocato da soli, come un solitario, è noioso.

Presi in considerazione l'idea di scrivere un giocatore artificiale basato sull'intelligenza artificiale, per allenare gli esseri umani, ma avevo troppo altro lavoro urgente da svolgere per il dottorato.

F.O.: Lo Spacewar! originale nacque al MIT nel 1961–62 grazie al lavoro di Steve "Slug" Russell e di altri membri della comunità hacker del MIT.

Nel 1972, al SAIL veniva eseguito il programma SWR, che vediamo Ralph Gorin avviare nel racconto di Brand. Ricordi in che modo la versione di Stanford derivasse dallo Spacewar! originale del MIT?



Spacewar! sul DEC - PDP11 a Stanford'

Chi lo portò a Stanford, chi ne realizzò il porting e chi successivamente lo modificò o lo ampliò?

Quali furono, nello specifico, i contributi di Ralph Gorin? E, più in generale, in che modo Spacewar!, il suo codice sorgente e le sue varie versioni circolavano concretamente tra MIT, Harvard, Stanford e altri laboratori: attraverso nastri, DECUS, scambi diretti tra programmatori oppure con altri mezzi?

BGB: Le risposte a queste domande si trovano altrove... non fanno parte della mia esperienza diretta e personale.

F.O.: Quanto cambiò Spacewar! la possibilità di giocare in cinque rispetto alla tradizionale partita a due giocatori?

BGB: Divertimento per altre tre persone, utilizzando all'incirca la stessa quantità di risorse informatiche.

Stare tutti ammassati attorno a una console era ovviamente più sociale che giocare su cinque console separate.

F.O.: Dal resoconto del 1972 sappiamo che il programma offriva numerose opzioni configurabili.

I giocatori sperimentavano regolarmente configurazioni differenti?

BGB: Io non armeggiavo con le opzioni: accettavo quelle preferite dallo sfidante.

F.O.: Brand descrive la configurazione delle Olimpiadi come comprendente quattro mine spaziali, nessun danno parziale, punteggi visualizzati, un tubo lanciasiluri anziché due e posizioni iniziali casuali.

BGB: La competizione si svolse con DUE TUBI LANCIASILURI? Clic: Y.

La descrizione precedente non corrisponde al sacro testo di Rolling Stone, né al testo di Two Cybernetic Frontiers.

F.O.: Perché furono scelte proprio quelle impostazioni?

BGB: Ralph Gorin voleva impostare le posizioni iniziali su «Yes».

La partita standard comincia dalle posizioni standard.

F.O.: Brand descrive Ralph Gorin mentre configura

personalmente SWR prima delle partite.

Ricordi se quelle impostazioni furono scelte da Gorin, concordate tra i giocatori oppure facevano già parte di una configurazione comunemente utilizzata per le partite competitive?

BGB: Impostazioni scelte da Gorin. Nessuno obiettò.

Non era la configurazione standard predefinita.

F.O.: Quali controller utilizzaste esattamente durante le Olimpiadi di Spacewar?

Erano le scatole con quattro pulsanti di controllo descritte da Brand oppure furono utilizzati anche controller del tipo joystick?

BGB: Quattro pulsanti su una scatola, come descritto da Brand. Nessun joystick venne utilizzato per giocare a Spacewar.

F.O.: Ricordi chi progettò o costruì i controller utilizzati al SAIL e se derivassero dalle scatole di controllo originali di Spacewar!, sviluppate al MIT da Alan Kotok e Robert A. Saunders?

BGB: Le scatole utilizzate quella particolare sera furono probabilmente costruite da Ted Panofsky oppure da Joe Zingheim, i due tecnici che quell'anno si occupavano della manutenzione delle varie apparecchiature.

Diverse serie precedenti di scatole di controllo avevano piccoli pulsanti economici che tendevano a rompersi, perciò nel corso di quell'anno i pulsanti sostitutivi divennero progressivamente più grandi.

Penso che sia stata una forte nostalgia, più che l'orgoglio per un risultato ottenuto, a spingere Kotok e Saunders a rivendicare ciascuno l'invenzione e la realizzazione di questa parte banale di Spacewar.

F.O.: Quanto influiva il tipo di controller sull'esperienza di gioco di Spacewar! e sulle prestazioni di un giocatore esperto?

BGB: Un salto quantico: uno o due pulsanti rotti rappresentano uno svantaggio fatale.

Per quell'epoca e per noi vecchissimi veterani, meglio i pulsanti dei joystick.

Il joystick presenta però il vantaggio di evitare che si attivi accidentalmente l'iperspazio quando si intende semplicemente ruotare.

F.O.: I singoli giocatori avevano astronavi o posizioni sullo schermo preferite?

BGB: A me piace giocare con Pointy Fins.

Non ho fatto un'indagine tra gli altri giocatori.

F.O.: Esistevano tecniche non documentate, trucchi, particolarità o persino bug nella versione di Stanford che i giocatori esperti conoscevano e sfruttavano?

BGB: Quale documentazione esisteva, oltre al listato del codice sorgente? (ride)

F.O.: Spacewar! era evidentemente molto più di un semplice intrattenimento al SAIL: Les Earnest raccontò a Stewart Brand che il meccanismo utilizzato dal sistema time-sharing per gestire la grafica interattiva era effettivamente chiamato "Spacewar Mode".

Dal tuo punto di vista di programmatore, Spacewar! ebbe un ruolo pratico nello sviluppo o nella sperimentazione delle tecnologie del SAIL, per esempio nella grafica interattiva, negli schermi, nei dispositivi di input o nel software in tempo reale?

Se sì, ricordi qualche esempio specifico?

BGB: Sì. La modalità Spacewar è necessaria per la robotica e per i segnali periodici dell'orologio in tempo reale.

In effetti, il programma che visualizza il quadrante di un OROLOGIO è l'esempio di riferimento per mostrare come utilizzare la chiamata di sistema UUO di Spacewar.

F.O.: Oggi i giocatori e gli sviluppatori sono generalmente considerati due ruoli distinti.

Al SAIL, tuttavia, molti giocatori di Spacewar! erano anche programmatori capaci di comprendere e modificare il gioco stesso.

In che modo questo influenzava il tuo rapporto con Spacewar!?

Esisteva davvero una distinzione tra chi «giocava» e chi «sviluppava», oppure le due attività si sovrapponevano

naturalmente?

BGB: Quasi tutti al SAIL erano programmatori.

La maggior parte di noi non aveva il tempo libero aggiuntivo o il desiderio di modificare Spacewar.

F.O.: Quanto erano comuni le partite competitive di Spacewar! al SAIL prima del torneo organizzato da Stewart Brand?

BGB: Le partite competitive di Spacewar non erano una consuetudine.

Spacewar era semplicemente un divertimento o una dimostrazione grafica.

F.O.: I giocatori del SAIL avevano già una sorta di classifica informale di chi fossero i più forti?

BGB: Sì.

Non ci furono sorprese.

F.O.: Prima delle Olimpiadi, consideravi Ralph Gorin uno dei tuoi avversari più forti?

BGB: Dave Poole sarebbe stato l'avversario più forte, ma quella sera andò a ballare danze popolari.

F.O.: Tu e gli altri giocatori consideravate l'evento una competizione seria, uno scherzo, una festa oppure una combinazione di tutte e tre le cose?

BGB: Doveva essere una festa.

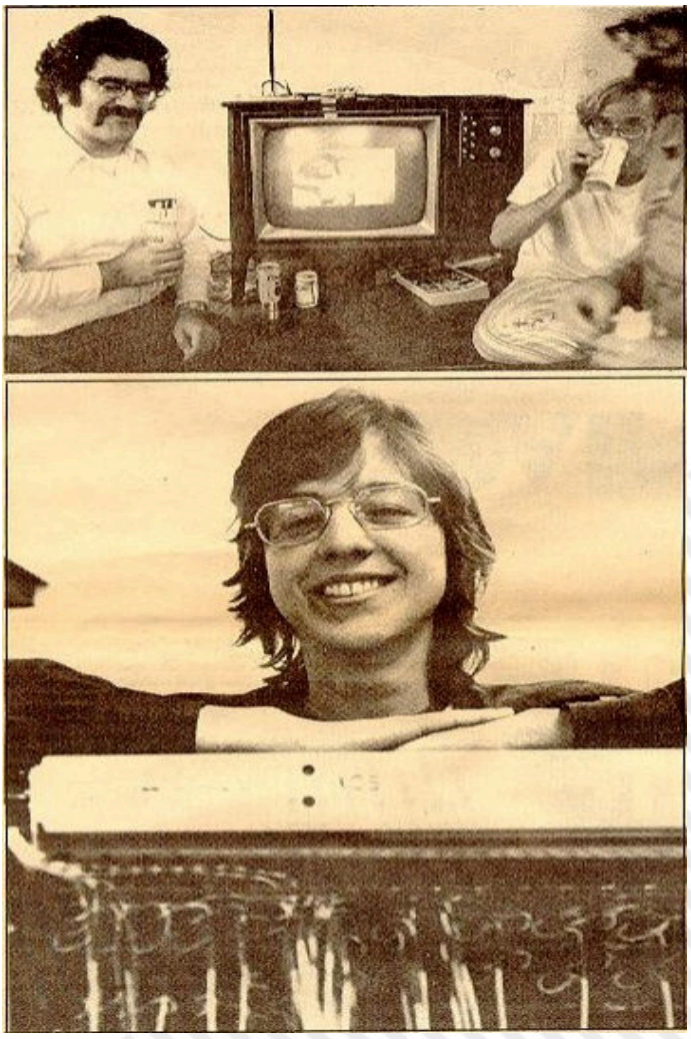
Spacewar non è uno scherzo e non si trattava di una competizione seria.

F.O.: Le coppie del torneo a squadre erano state stabilite in precedenza oppure si formarono quella sera?

BGB: Si formarono quella sera.

F.O.: Quanto era forte Ralph Gorin come giocatore di Spacewar!?





Copertina del numero della rivista Rolling Stone del 7 dicembre del 1972 e scatti dell'articolo

BGB: Forte quanto me.

Essenzialmente finivamo in parità ogni volta che avevamo l'occasione di giocare uno contro uno.

F.O.: Che cosa distingueva il suo stile di gioco e quanto era difficile affrontarlo?

BGB: Nessuno stile particolare: padroneggiava tutte le tattiche abituali.

F.O.: In una precedente intervista hai ricordato di esserti esercitato fino a riuscire a controllare contemporaneamente due astronavi in una partita a cinque giocatori, una con ciascuna mano, arrivando persino a posizionarle in modo da creare un fuoco incrociato contro un altro giocatore. Utilizzasti mai questa capacità durante le Olimpiadi?

BGB: No, non la utilizzai durante le Olimpiadi.

Durante le Olimpiadi ogni giocatore controllava soltanto una delle astronavi.

F.O.: E ricordi se ci fu uno scontro particolarmente significativo tra te e Ralph Gorin, oppure con un altro giocatore, durante il Free-for-All?

BGB: Non ricordo alcun dettaglio di quelle specifiche partite, né chi avesse quale astronave.

F.O.: I punteggi, i tabelloni degli incontri o i risultati furono registrati da qualche parte?

BGB: Sulla lavagna, nell'angolo superiore sinistro della sala degli schermi, visibile in questa fotografia (vedi l'immagine a destra).

F.O.: Potrebbe esistere ancora un tabellone del torneo, una stampa, un file informatico o qualche altro documento?

BGB: Forse, se Leibovitz o Rolling Stone conservano ogni singolo fotogramma di ciascun rullino fotografico che lei scattò quella sera.

Le Olimpiadi di Spacewar di mercoledì 18 ottobre 1972 ebbero luogo PRIMA che venisse scritto il primo nastro di backup DART affidabile giunto fino a noi, il 5 novembre 1972. Nell'archivio SAILDART ci sono file relativi al mese di ottobre 1972, ma nessuno di essi menziona le Olimpiadi di Spacewar. Posso però dedurre chi si trovasse nell'edificio il 18 e il 19 ottobre.

F.O.: Che cosa provasti concretamente nel momento in cui vincesti?

BGB: Cercai di «tornare al lavoro» dopo che i partecipanti alla festa lasciarono l'edificio, intorno alle due del mattino, ma non riuscivo a concentrarmi.

Così passeggiai per un po' all'esterno del laboratorio e poi tornai a casa prima dell'alba.

F.O.: Ti sembrò qualcosa di significativo oppure semplicemente la vittoria in una serata particolarmente divertente di Spacewar!?

BGB: La mia fotografia finì su Rolling Stone.

Fu significativo.

F.O.: Brand fu l'ideatore dell'evento, ma quale fu concretamente la sua presenza durante la competizione? Interagì molto con voi, fece domande e seguì attentamente



Bruce vittorioso!



Annie Leibovitz (des.) di Rolling Stone

le partite, oppure, una volta iniziato il torneo, rimase principalmente un osservatore?

BGB: Durante il torneo, Brand sembrava essere un osservatore che si occupava del proprio registratore, per registrare le chiacchiere dei giocatori.

I nomi dei giocatori degli incontri erano scritti sulla lavagna che si vede sullo sfondo della famosa fotografia di Bruce con la scatola dei pulsanti in mano.

L'uomo in piedi sullo sfondo, a destra, è RHT, Russell Taylor.

La lavagna serviva sia come calendario degli incontri sia come tabellone dei punteggi.

Ricordo di aver visto per la prima volta le lavagne bianche allo Xerox PARC.

F.O.: L'articolo di Brand comprende la celebre fotografia di Annie Leibovitz che ti ritrae come vincitore del Five-Man Free-for-All, mentre brandisci i controlli di Spacewar!. Che cosa ricordi del momento in cui venne scattata quella fotografia?

BGB: Non ricordo che Annie abbia detto qualcosa.

Era semplicemente una paparazza instancabile che continuava a scattare fotografie, far avanzare la pellicola e cambiare i caricatori.

Dopo la partita finale ero ancora seduto sulla mia sedia davanti allo schermo, consapevole di aver vinto.

Brand disse qualcosa come: «Guarda verso la macchina fotografica, alzati e mettiti lì, dietro lo schermo».

Avevo ancora i pulsanti in mano.

F.O.: La posa fu spontanea, suggerita da lei oppure preparata appositamente per l'articolo?

BGB: Spontanea.



Uno scorcio del gioco Starsector di David Baumgart



F.O.: Quando leggesti per la prima volta l'articolo di Brand su Rolling Stone, ti sembrò una rappresentazione accurata delle persone, dell'atmosfera e degli eventi che avevi vissuto?

BGB: Sì.

F.O.: Oggi le Olimpiadi Intergalattiche di Spacewar vengono spesso descritte come il primo torneo videoludico documentato e, retrospettivamente, persino come il primo evento di «eSport».

Che cosa pensi di queste definizioni?

BGB: La definizione di primo torneo videoludico documentato sembra accurata.

Tra le possibili obiezioni c'è il fatto che il Tris, gli scacchi e la dama giocati al computer erano stati documentati già qualche tempo prima del 1972.

Da bambino, visitando il Chicago Museum of Science and Industry, giocai a Tris contro una macchina attraverso uno schermo televisivo: il mio primo torneo di eSport?

Anche la definizione «eSport» è accurata.

Cercate la definizione di eSport:

«Gli eSport, abbreviazione di sport elettronici, consistono in competizioni organizzate di videogiochi tra giocatori professionisti o squadre».

F.O.: Ritieni che queste definizioni siano utili per descrivere ciò che accadde oppure rischiano di applicare categorie moderne a qualcosa che tu hai vissuto in modo completamente diverso?

BGB: Nessun problema.

Sono abbastanza utili, dato che mi tengono in vita a rispondere a domande di interviste storiche.

F.O.: Osservando i videogiochi e le competizioni videoludiche più di mezzo secolo dopo, riconosci qualcosa della cultura di Spacewar! e del SAIL nel mondo videoludico di oggi?

BGB: Sì, certamente, anche se la cattiva intelligenza artificiale e il cattivo rumore proveniente dalla Silicon Valley hanno dominato i media.

Mio figlio, David G. Baumgart, è uno sviluppatore indipendente di videogiochi.

Il suo progetto attuale assomiglia moltissimo a Spacewar. STARSECTOR

F.O.: Quando lavoravi al SAIL, avevi già la sensazione che il futuro dell'informatica sarebbe diventato sempre più interattivo, grafico e personale, oppure questa è in gran parte un'interpretazione che abbiamo costruito retrospettivamente, osservando ciò che è accaduto negli anni successivi?

BGB: Io sono uno di quelli che hanno intenzionalmente reso l'informatica interattiva, grafica e personale.

Alan Kay coniò la famosa frase «Il modo migliore per prevedere il futuro è inventarlo» durante una riunione del 1971 allo Xerox PARC, per sottolineare che il futuro della tecnologia è qualcosa che le persone possono costruire attivamente, anziché aspettare passivamente.

F.O.: Tra le persone che parteciparono alle Olimpiadi o che, più in generale, appartenevano alla comunità di Spacewar! al SAIL, chi ritieni sarebbe particolarmente importante intervistare oggi, per preservare ricordi o aspetti della storia che non sono ancora stati documentati adeguatamente?

BGB: La risposta richiede ulteriori riflessioni.

F.O.: Infine, esiste una persona, un evento, un dettaglio tecnico, un aneddoto oppure un aspetto di Spacewar!, del SAIL o delle Olimpiadi del 1972 che gli storici hanno trascurato, oppure qualche fatto o interpretazione della storia comunemente raccontata che vorresti correggere o chiarire?

In altre parole, c'è qualcosa che vorresti particolarmente lasciare agli archivi storici, mentre abbiamo ancora l'opportunità di ascoltarlo direttamente da qualcuno che era presente?

BGB: Mi viene in mente la celebre battuta di Winston Churchill: «La storia sarà benevola con me, perché intendo scriverla io stesso».

Una risposta completa richiede ulteriori riflessioni.

F.O.: UN ULTIMO MESSAGGIO!

Molti lettori sarebbero felici di ricevere un saluto personale da parte tua.

BGB:

Saluti,

→ Amici!

→ Romani!

→ Miei lettori cybernauti italiani!

—

Darò voce alla visione degli

storici Giochi Olimpici:

→ Sostituite la guerra vera con...

→ giochi sportivi!.....

Amen.

—

Benedizione:

→ → Pace

→ → Amore

→ → Felicità

→ e che la Forza sia con voi!

—

La voce di uno studente che

incita le masse:

Giocate con impegno! Giocate con impegno! Giocate la vostra partita!

—

BgB

F.O.: Davvero divertente! :-D

Grazie ancora, Bruce, per il tempo che ci hai dedicato e per il tuo contributo nel preservare una testimonianza diretta di questo straordinario periodo della storia dell'informatica e dei videogiochi.

Un caro saluto e a presto.

BGB: Benvenuti nelle mie raccolte di materiale sorgente sulla paleo-intelligenza artificiale, a disposizione di autori e archivi.

Le fotografie di Annie Leibovitz e di Bruce mentre festeggia la vittoria sono state rielaborate e colorate mediante l'utilizzo dell'AI. Tutti i diritti sulle immagini utilizzate in questo articolo, rimangono dei rispettivi proprietari.

Tutti i testi sono personali e non rielaborati con l'uso dell'AI. L'intervista è originale e arriva direttamente dalla "voce" di Bruce Baumgart.

VARESE RETROCOMPUTING: dalle origini al 27 Settembre 2026.

Cronaca di una giornata di amicizia e tanto sano retrocomputing

di Filippo Occulto



Manifesto di Varese Retrocomputing 2026

Sin dalle prime ore del mattino si respira l'atmosfera delle grandi occasioni, di quegli appuntamenti che per noi retrocomputeristi contano più di una prima alla Scala.

Una volta all'anno, gli spazi dell'Opera Hub di via Borghi, a Comerio, nell'area dell'ex Whirlpool, diventano il punto d'incontro di appassionati di informatica, sviluppatori, esperti di elettronica e smanettoni di ogni genere. Non occorrono grandi folle per rendere speciale un appuntamento del genere: bastano la voglia di ritrovarsi, qualche vecchio computer acceso e tante storie da raccontare.



Gianni "BBk" Zamperini (a sinistra) - Bruno Grampa, Gianni "BBK" (a destra)

Ripartiamo dall'inizio. La storia di Varese Retrocomputing comincia il 27 aprile 2003, quando viene organizzata la prima edizione nel parco di Villa Toeplitz, a Varese, in collaborazione con il MUel. A promuovere l'iniziativa è Gianni "BBK" Zamperini, figura centrale della manifestazione, scomparso nel 2014.

La sua memoria è stata celebrata nel corso dell'edizione 2024, in occasione della fondazione dell'omonima associazione. A ricordarlo è stato il nuovo direttivo, guidato da Bruno Grampa e composto da Gabriele "Gab" Petino, Amedeo Valoroso, Fabio Massa, Lorenzo Vivarelli, Eugenio Ciceri e Francesco Lecca.

L'associazione nasce con un obiettivo che va oltre l'organizzazione degli incontri annuali: raccogliere le varie anime che orbitano intorno al retrocomputing e arrivare un giorno a realizzare un museo dell'informatica nel quale le macchine non siano soltanto oggetti da esposizione, ma continuino a funzionare e a raccontare la propria storia.



Citofono-Space Invaders - Apple Macintosh replica - Altair 8800 mini



C64 Ultimate con Impossible Mission III



La serie di Levitator di Eugenio Ciceri

Prima del trasferimento a Comerio, le edizioni precedenti al 2025 si erano svolte nella cornice di Villa Truffini, a Tradate, sempre in provincia di Varese.

Personalmente, come socio e partecipante attivo alle iniziative di Varese Retrocomputing, ogni edizione rappresenta un'occasione per ritrovare vecchi amici, conoscere nuovi appassionati e confrontarmi su quella tecnologia che, a distanza di decenni, continua a suscitare il mio e altrui interesse. Appena arrivato, finisco immancabilmente coinvolto in discussioni di ogni genere, soprattutto quando si parla delle nuove produzioni destinate ai computer del passato. Perché il retrocomputing non è soltanto conservazione della memoria: è anche sviluppo, sperimentazione e voglia di continuare a creare.

Tra le prime macchine che attirano la mia attenzione c'è il Commodore 64 Ultimate di Lorenzo Vivarelli, sul quale gira Impossible Mission III, nuovo capitolo di una serie che ha segnato la storia del Commodore 64. Un'occasione per osservare da vicino quanto sia ancora viva la scena dello sviluppo videoludico per gli home computer degli anni Ottanta. Quest'anno, inoltre, la manifestazione è stata preceduta da un appuntamento particolare. Il sabato, negli stessi spazi di Opera Village, l'Associazione Hamlet ha organizzato una serata dedicata all'incontro tra cultura tecnologica, memoria del Novecento e sperimentazione sonora.

Protagonista il Collettivo Méliès, con "Il rumore del secolo", una suite storiografica per intonarumori, elettrofoni e altri strumenti non convenzionali. Un progetto che intreccia



Redazione di ZZAP



Atari 2600 Pac-man Edition



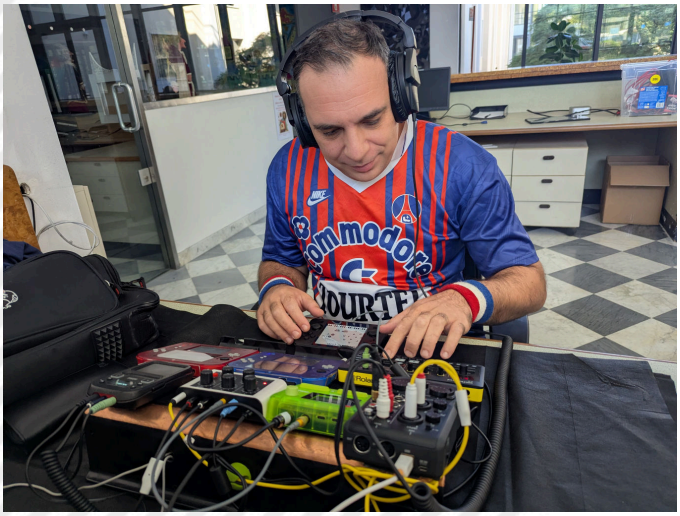
Retrocampus Minitel BBS



Pong su Tv BNC e Atari 2600

performance musicale, documentario di montaggio e narrazione, esplorando il rapporto tra tecnologia e suono. Non potevano mancare gli amici di ZZAP!, guidati da Paolo "Paolone" Besser, presenti con il loro stand e una selezione delle riviste autoprodotte dall'associazione Airons di Vigevano. Spazio anche all'editoria specializzata, con gli stand dedicati ai libri sulla storia del videogioco di Andrea Contato e alle esposizione hardware di Retroedicola.

Altra presenza eccellente della Manifestazione: Eugenio Ciceri con la sua serie di Levitator, ormai non limitata al solo Zx Spectrum, ma ad ogni riproduzione 8bit dell'epoca. Con una leggerissima predilezione per le macchine di casa Sinclair, svettano anche macchine Atari e Commodore. Nel mare magnum delle console fanno la loro bella mostra anche repro moderne di pezzi che sono un must, come l'Atari 2600 Pac-man edition. Immancabili i classici Pong-Style, NES, Sega in varie salse, Sony Playstation. Le immancabili apparizioni del Commodore VIC 20 con nuovi titoli direttamente dalla scena homebrew, ZX 80 firmato dalla mano di Federico Faggin, Amiga 500, 600, 1000 e 2000, Pet CBM 8032 e varie altre macchine minori. Per gli amanti del modding una versione di Space Invaders che gira direttamente su un citofono e una repro dell' Apple Machintosh del 1984, oltre



Michele "Mik" Lias con il suo stupendo Dj Set digitale



AlphaSyntauri con Music System





che una riproduzione di un Altair 8800 mini. Onnipresente Retrocampus con il suo ormai famoso terminale minitel di Francesco Sblendorio, dove gira una BBS completa.

Tornando alla giornata di Varese Retrocomputing, una delle sorprese più gradite è stata la partecipazione del pubblico, quest'anno particolarmente numeroso, complice anche una giornata di fine estate favorevole. Un risultato che sembra premiare il lavoro dell'associazione e l'impegno profuso nell'organizzazione.

A scandire i momenti della manifestazione non poteva mancare il DJ set dell'amico e socio Michele "Mik" Lias, presenza ormai abituale e protagonista dell'accompagnamento musicale negli eventi ai quali partecipa Varese Retrocomputing.

Tra gli espositori ritroviamo l'ormai immancabile collezione di EsoCop, presentata da Sergio Gervasini con la consueta collaborazione di Stefania Calcagno. Una presenza che negli anni è diventata uno dei riferimenti della manifestazione.

Altro appuntamento atteso è quello delle conferenze, che ogni anno richiamano nella sala al piano superiore appassionati, esperti e semplici curiosi. Sul palco si alternano nomi conosciuti della scena retroinformatica e nuovi relatori, offrendo occasioni di approfondimento su temi che spaziano dalla storia dell'informatica al retrogaming.

Tra le curiosità di questa edizione merita una menzione un alphaSyntauri, collegato a un computer Apple per produrre musica attraverso il sistema MusicSystem. Una testimonianza di quando informatica e sperimentazione musicale cominciavano a incontrarsi, aprendo possibilità che oggi diamo quasi per scontate.

Presente anche THE THINKER LAB, un laboratorio creativo dedicato ai giovani talenti, a conferma di come manifestazioni di questo genere possano rappresentare un punto d'incontro tra generazioni e competenze differenti.

Nel complesso, Varese Retrocomputing è riuscita ancora una volta a riunire espositori, collezionisti e realtà impegnate nella conservazione e nella valorizzazione della storia informatica.

Per le prossime edizioni mi piacerebbe vedere una partecipazione ancora maggiore di sviluppatori e creatori di nuovi videogiochi per le piattaforme a 8 e 16 bit. Sarebbe un modo per affiancare alla conservazione del passato anche quella scena contemporanea che continua a produrre software per macchine nate oltre quarant'anni fa.

La consueta scorpacciata di retrocomputing, per quest'anno, si è dunque conclusa. Ma il prossimo appuntamento è già all'orizzonte: dopo Halloween ci ritroveremo a Brusaporto Retrocomputing, che domenica 8 novembre celebrerà la sua diciannovesima edizione.

Per il momento è tutto.

Alla prossima!

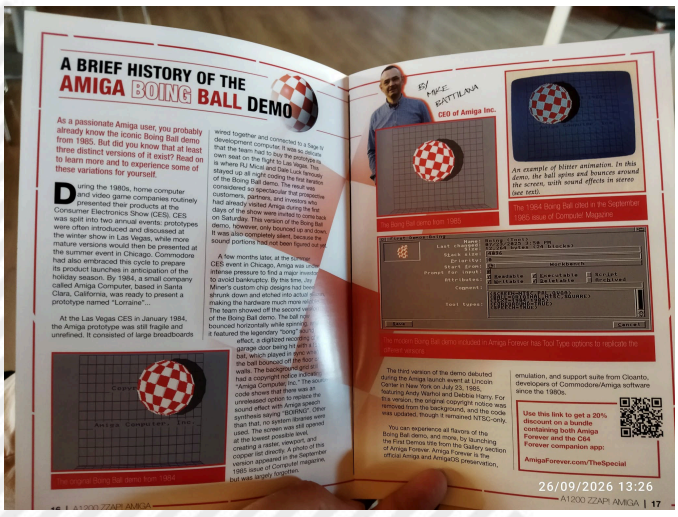
La conferenza di Michele Battilana a Pianeta Amiga del 26 e 27 Settembre

di Carlo Savorelli

Uno dei momenti più attesi dalla comunità Amiga alla manifestazione Pianeta Amiga tenutasi ad Empoli lo scorso 26 e 27 settembre era certamente la conferenza di Michele Console Battilana, fondatore di Cloanto e CEO di Amiga Corporation. C'è fermento nel mondo Amiga, come non capitava da anni. Le notizie sono corse veloci dopo la RetCon londinese dello scorso Giugno (https://retconfestival.co.uk/). Il 4 agosto, con un comunicato ufficiale

(https://amiga.com/news/20260804-amiga-update),
Amiga Corporation ha annunciato la stipula di un accordo di licenza con Commodore International, e la risoluzione dell'annosa controversia legale con Hyperion Entertainment BV. Questa duplice buona notizia ha segnato subito un cambio di passo: neanche due settimane dopo, all'evento "Kickstart 3.0 UK" di Derby(https://www.amigashow.com/), Amiga Corporation ha svelato il nuovo logo ufficiale di Amiga, basato sull'iconico logo del 1985 ma ridisegnato da zero riducendo la difformità degli angoli e unificando i serif per un look moderno e pulito. Infine il diciotto agosto, la Retro Games Ltd. annunciava sulla propria pagina Facebook che il "The A1200" è un prodotto Amiga ufficiale.
Contrariamente al "The A500 Mini" del 2022, il nuovo prodotto di Retro Games Ltd. è targato ufficialmente Amiga, e sfoggia il logo Amiga orgogliosamente sulla propria confezione... Come tutti i presenti ad Empoli hanno potuto apprezzare, durante l'unboxing di una delle primissime unità giunte in Europa (il The A1200 era stato svelato alla Gamescom di Colonia, ma ancora privo di scatola e inserti).

L'emozione di avere dopo tanti anni un prodotto Amiga ufficiale, completo di Workbench e distribuito nei negozi a meno di 200 euro, era palpabile nella sala. Purtroppo l'unità non è stata accesa, ma ne scriveremo più in dettaglio in occasione del lancio, ormai imminente. Sul versante del marketing, l'operazione sembra decisamente più curata rispetto a quella del The A500 Mini. L'uscita è fissata per dicembre, giusto in tempo per Natale, e ogni confezione include un booklet speciale modellato sulla storica rivista Zzap!. Curato da Chris Wilkins di Fusion Retro Books, il volumetto contiene un fumetto su Alien Breed (perfetto traino per la recente 35th Anniversary Collection) e diversi articoli di approfondimento: dalla storia della mitica Boing Ball fino a focus su Defender of the Crown — anch'esso alla



Il booklet contenuto all'interno della cofnezione del The A1200



Michele Battilana durante la sua presentazione

ribalta grazie al remake di Nordcurrent Labs — e sulla scena indipendente, come Richard Löwenstein con il suo Reshoot Proxima 3. Insomma, ce n'è davvero per tutti i gusti.

Se l'unboxing del The A1200 è stato il momento più scanzonato della conferenza, è vero che le domande del pubblico si sono concentrate maggiormente sul futuro di Amiga. Uno dei nodi principali è il doppio binario tra Amiga OS 4 (ancora di competenza di Hyperion) e la linea Amiga OS 3 basata su 68K. Si deve riflettere sul cambio di nome dell'OS per evitare la confusione tra le due numerazioni. Inoltre si deve spingere per una sincronizzazione degli ambienti: sarebbe ottimale che lo stesso software Amiga potesse essere fruito su dispositivi diversi garantendo la transizione più semplice.

Si è inevitabilmente parlato del rapporto con la nuova Commodore International di Perifractic, anche se l'umore generale in sala era di sollievo per il fatto che per la prima volta dopo un periodo lunghissimo, Amiga ha le gambe per camminare da sola. Non sono mancate osservazioni acute di Battilana sul mondo del "retro": c'è chi lo vede come una inesauribile gallina dalle uova d'oro, chi come una discesa a picco sulle montagne russe, i cui guadagni sono destinati ad estinguersi non appena verranno meno le generazioni legate al passato dalla nostalgia. In ogni caso, una volta di più, Amiga continuerà a far parte di questa storia.



Carlo con il The A1200







Nelle foto il The A1200, i componenti hardware della scatola e le viste laterali dello stesso (foto di Francesco Fiorentini)

RAZING CORE - Il Master System torna a sparare. E lo fa alla grande!!!

by Carlo Nithaiah Del Mar Pirazzini

Quarant'anni? E chi se ne importa. Il Sega Master System continua ostinatamente a dimostrare che una console non muore quando esce di produzione, ma soltanto quando smettiamo di usarla. E, a giudicare da quello che sta accadendo nella scena homebrew, quel momento sembra ancora decisamente lontano. Una delle dimostrazioni più convincenti arriva da Razing Core, nuovo shoot 'em up a scorrimento verticale realizzato da Bad Computer, pubblicato in versione completa l'8 settembre 2026 per Sega Master System.

Bad Computer non è nuovo alla macchina. Dopo progetti come Frontier Force, Astro Climber e Blast Arena, lo sviluppatore torna con un titolo decisamente più ambizioso, capace di mettere alla prova l'hardware Sega e, soprattutto, di ricordarci quanto potenziale fosse ancora nascosto all'interno di quella piccola console nera.

La premessa narrativa è quella classica che ogni appassionato di shoot 'em up potrebbe recitare anche nel sonno o nelle sue preghiere. Durante una missione nello spazio profondo, l'astronave Horizon incontra un misterioso oggetto alieno chiamato semplicemente "Core". La sfera prende il controllo dei sistemi della nave e la trascina attraverso l'iperspazio verso la Terra. Al ritorno, però, il pianeta è sotto attacco da parte di una gigantesca armata extraterrestre.

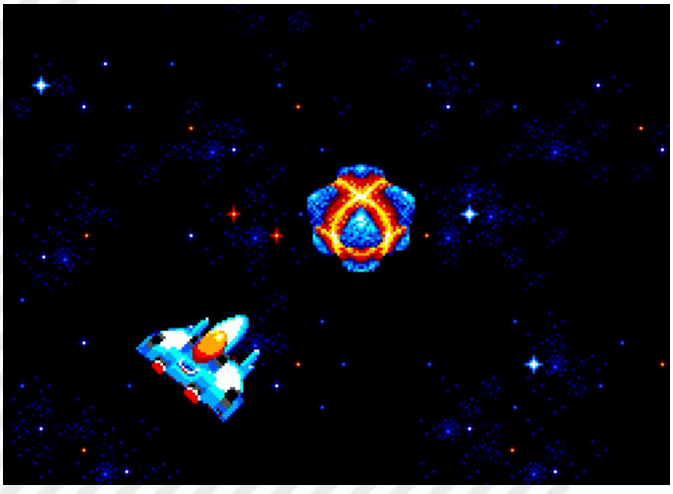
Non rimane quindi molto tempo per interrogarsi sulle intenzioni del Core: bisogna impugnare il controller e iniziare a sparare.Ed è qui che Razing Core comincia davvero a mostrare i muscoli.

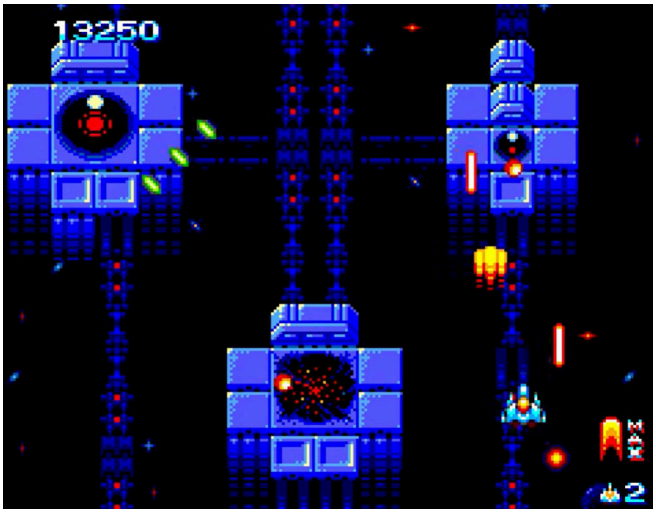
Il gioco propone sette stage ambientati attraverso il sistema solare, con scenari differenti, mid-boss e giganteschi avversari finali. La struttura richiama inevitabilmente la grande scuola degli sparatutto verticali degli anni Ottanta e Novanta, ma Bad Computer non si limita a riprodurne meccanicamente le caratteristiche. Introduce invece una serie di idee che rendono il sistema di combattimento sorprendentemente articolato considerando che tutto deve convivere con i due pulsanti del controller originale del Master System.

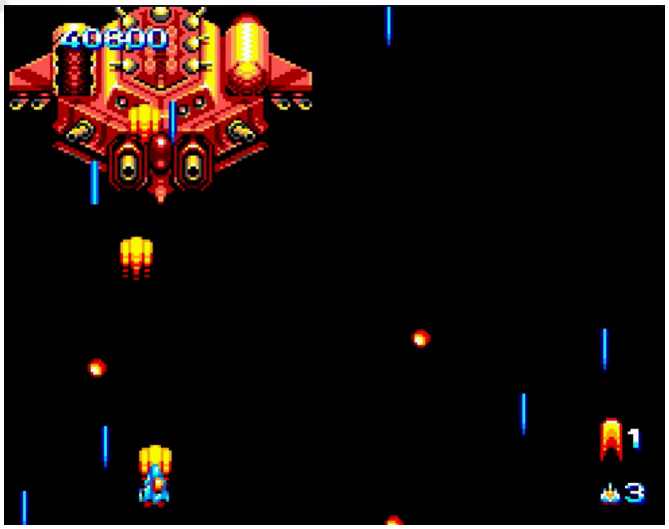
L'Horizon dispone innanzitutto di un fuoco rapido principale, attivabile semplicemente tenendo premuto il primo pulsante. Il secondo controlla invece l'attacco del Core: una sorta di arma rotante che colpisce gli avversari nelle vicinanze e può contemporaneamente offrire protezione dai proiettili nemici. Non è però possibile abusarne, perché dopo l'utilizzo necessita di un periodo di ricarica.

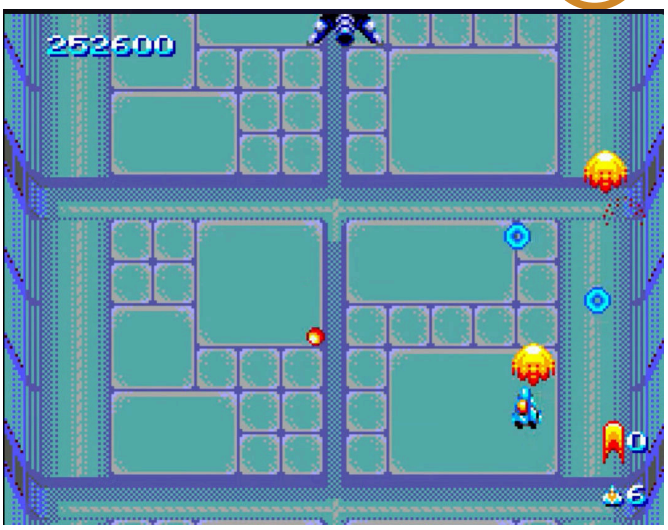
Premendo entrambi i pulsanti entra invece in gioco il Burst Charge, una delle meccaniche più interessanti del gioco. L'attacco può raggiungere cinque livelli di carica e, quando utilizzato, offre una devastante risorsa offensiva accompagnata da una temporanea invulnerabilità.

A queste possibilità si aggiungono le armi secondarie,









identificate dalle lettere A, B, C e D. Si passa da colpi ad ampio raggio a potenti fasci capaci di attraversare i nemici, mentre raccogliendo gli appositi potenziamenti è possibile aumentarne progressivamente l'efficacia fino a cinque livelli. Il risultato è un sistema immediato da comprendere ma sufficientemente profondo da spingere il giocatore a sperimentare. Ed è probabilmente uno degli aspetti meglio riusciti di Razing Core, sembra davvero un gioco concepito secondo la filosofia degli 8 bit, invece di un titolo moderno semplicemente travestito da produzione retrò.

Anche il livello di difficoltà dimostra una certa attenzione verso il giocatore contemporaneo. Sono disponibili tre modalità di difficoltà, accompagnate da continue infiniti e checkpoint in corrispondenza dei boss. Non significa che Razing Core sia una passeggiata, tutt'altro. Significa piuttosto che evita alcune delle punizioni più brutali tipiche degli shooter dell'epoca.

Morire non significa nemmeno perdere immediatamente tutti i potenziamenti accumulati. Il livello dell'arma viene ridotto, ma il giocatore mantiene una parte della propria capacità offensiva. Una scelta intelligente, perché permette alla partita di mantenere un buon ritmo senza eliminare completamente la tensione.



Naturalmente rimane il punteggio. Razing Core conserva orgogliosamente quella filosofia da sala giochi nella quale completare l'avventura rappresenta soltanto una parte dell'esperienza. Medaglie, zone nascoste, vite extra e bonus invitano a ripetere gli stage cercando traiettorie migliori e punteggi sempre più elevati.

Ma è probabilmente dal punto di vista tecnico che il gioco lascia l'impressione più forte.
Gli scenari sono colorati, dettagliati e soprattutto molto differenti tra loro. Sullo schermo si muovono contemporaneamente numerosi sprite e i boss occupano porzioni considerevoli dell'area di gioco. È il genere di situazione nella quale ci aspetteremmo di vedere il povero Master System implorare pietà tra rallentamenti e flickering.... E invece nulla... Razing Core riesce a mantenere l'azione sorprendentemente fluida.

Naturalmente non bisogna dimenticare che stiamo parlando di hardware nato nella prima metà degli anni Ottanta. Proprio per questo, vedere una tale quantità di elementi muoversi sullo schermo conservando una buona leggibilità dell'azione rappresenta forse uno degli aspetti più affascinanti dell'intera produzione.

Anche la componente sonora merita attenzione. Le musiche sono firmate da Lovepenguin, già autore della colonna sonora di Astro Climber, mentre le illustrazioni sono opera di Danmakuman. I brani accompagnano bene l'azione e, soprattutto, riescono a ottenere risultati notevoli dal chip sonoro della macchina. Le composizioni sono sufficientemente lunghe da evitare quella sensazione di

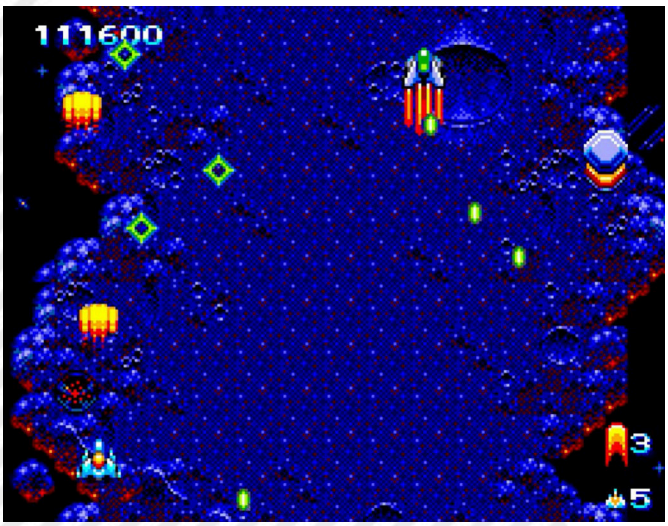


ripetitività che può diventare rapidamente fastidiosa in uno shoot 'em up.

Qualche imperfezione, naturalmente, rimane... Il Burst Charge richiede parecchio tempo per raggiungere la massima potenza e questo può spingere il giocatore a conservarlo troppo a lungo, trasformando un'interessante possibilità tattica in una risorsa utilizzata soltanto nelle situazioni più critiche. Alcuni nemici comuni, inoltre, sembrano possedere una resistenza leggermente eccessiva: quando lo schermo comincia a riempirsi, può diventare più conveniente evitarli piuttosto che cercare ostinatamente di eliminarli.

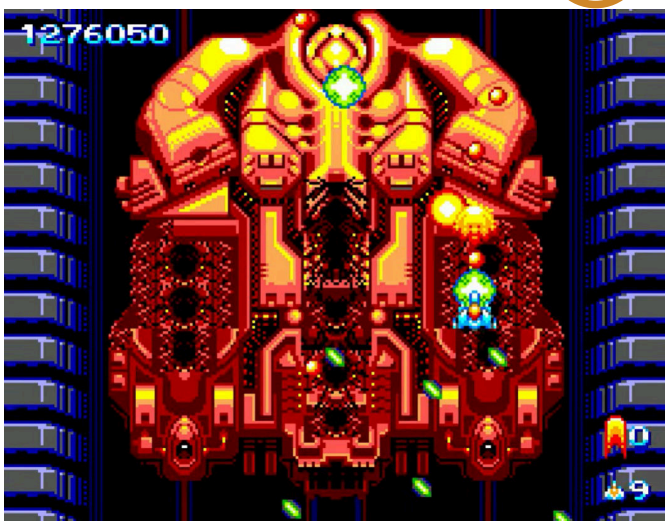
Si tratta comunque di osservazioni che incidono relativamente poco su un'esperienza complessivamente molto solida.

Particolarmente interessante è la volontà di Bad Computer di realizzare il progetto pensando innanzitutto all'hardware originale. Razing Core funziona su Sega Master System / Mark III ed è progettato per essere giocato a 60 Hz. È possibile utilizzarlo anche attraverso emulatori, sistemi FPGA e configurazioni Mega Drive compatibile mediante









flash cartridge.

La versione digitale viene distribuita come ROM, mentre è disponibile gratuitamente una demo contenente il primo livello. Una scelta che permette a chiunque di provarlo prima dell'acquisto. Bad Computer ha inoltre annunciato l'intenzione di realizzare una versione fisica nel 2027, prospettiva decisamente interessante per chi continua a utilizzare il proprio Master System originale.

Ed è forse proprio questo il punto più importante. Secondo noi, Razing Core non dovrebbe essere considerato soltanto una curiosità homebrew.

Titoli di questo tipo stanno lentamente cambiando il significato stesso del retrogaming. Per molti anni abbiamo considerato le vecchie console come sistemi appartenenti a un periodo storico concluso: macchine da preservare, collezionare ed emulare per continuare a giocare ai titoli pubblicati durante la loro vita commerciale. Oggi la situazione è differente.

Decenni di documentazione tecnica, reverse engineering, strumenti di sviluppo creati dalla comunità e conoscenza accumulata sull'hardware permettono agli sviluppatori indipendenti di affrontare queste macchine con una consapevolezza che spesso gli stessi programmatori dell'epoca non potevano avere.

Il risultato è affascinante: console teoricamente morte da decenni continuano a ricevere videogiochi nuovi.
Nel caso del Master System il fenomeno è ancora più interessante. La macchina Sega possiede alcuni ottimi shoot 'em up – impossibile non pensare a Power Strike o alla notevole conversione di R-Type – ma il genere non ha mai raggiunto sulla console la stessa importanza che avrebbe avuto successivamente su Mega Drive.

Razing Core sembra quasi voler recuperare parte di quella storia mancata. Guardandolo in movimento viene spontaneo chiedersi come sarebbe stato accolto un titolo simile se fosse apparso sugli scaffali nel periodo commerciale del Master System. Probabilmente avrebbe attirato parecchia attenzione. Ma forse è proprio questo il bello.
Razing Core non arriva con quarant'anni di ritardo. Arriva nel 2026, su Sega Master System. Ed è esattamente lì che vuole essere.

Più che un esercizio nostalgico, è la dimostrazione che preservare il passato videoludico non significa necessariamente congelarlo. Possiamo continuare a restaurare console, conservare cartucce e documentare software storico, ma possiamo anche utilizzare ciò che abbiamo imparato per creare qualcosa che allora non esisteva.

Il Master System, evidentemente, non ha ancora finito di raccontare la sua storia.
E se il futuro degli 8 bit passa attraverso produzioni come Razing Core, siamo più che felici di continuare ad ascoltarla. Capolavoro!

Giocabilità: 95%
Gameplay semplice ma divertente. Livelli ben strutturati e boss fight molto divertenti. Grande idea il CORE.

Longevità: 95%
Lo rigiocherete ancora ed ancora anche una volta terminato. E' una calamita.

NEW GAME

XIBALBA 64

Anno: 2026
Editore/Sviluppatore: Dominic Szablewski
Genere: FPS
Piattaforma: Nintendo 64
Sito web: https://modretro.com/products/xibalba-64

Nel 2014 Xibalba era nato come una piccola demo in JavaScript, progettata da Dominic Szablewski per mettere alla prova il suo motore di gioco. Dodici anni dopo, quella dimostrazione è diventata un titolo per Nintendo 64.
Pubblicato da ModRetro come gioco fisico per la console M64, ma compatibile anche con il Nintendo 64 originale, il progetto porta il giocatore in un mondo sospeso tra mitologia maya, tecnologia dimenticata e fantascienza da incubo. L'avventura si svolge nell'oltretomba di Xibalba, tra giungle, catacombe, basi tecnologiche e rovine misteriose. Il risultato è uno sparatutto in prima persona ispirato ai classici degli anni Novanta, ma con una personalità più enigmatica e inquietante. Non è un semplice porting della versione browser, Szablewski ha riscritto il gioco in C, aggiungendo livelli, armi, nemici, musica e modalità multiplayer.
La storia produttiva di Xibalba 64 è uno degli aspetti più interessanti del progetto. L'originale era stato realizzato come dimostrazione tecnica per il motore JavaScript Impact, in un periodo in cui giocare in 3D direttamente dal browser rappresentava ancora una notevole curiosità. Per trasformarlo in un prodotto completo, Szablewski lo ha adattato all'architettura del Nintendo 64. La fase iniziale del porting è stata relativamente rapida, ma il lavoro di espansione e ottimizzazione è durato diversi mesi. Il progetto è stato sviluppato utilizzando Libdragon, una libreria open source per la programmazione N64, e Tiny3D, un motore grafico capace di gestire trasformazioni, illuminazione e



rendering attraverso i coprocessori della console.
La scelta di pubblicarlo su cartuccia è particolarmente significativa. Secondo l'autore, Xibalba 64 è una delle pochissime nuove produzioni distribuite fisicamente per Nintendo 64 dopo la fine della vita commerciale originale della console. La trama è volutamente vaga. Il giocatore scende nelle profondità di un mondo ispirato a Xibalba, l'oltretomba della mitologia maya, attraversando templi in rovina, giungle, catacombe e strutture tecnologiche. La narrazione viene affidata a brevi testi introduttivi e alle ambientazioni. Non tutto viene spiegato e molti elementi restano avvolti nel mistero. Chi impartisce gli ordini? Che cosa rappresentano le creature? Perché rovine antiche e macchinari futuristici convivono nello stesso luogo? Questa scelta rafforza il senso di inquietudine. Il gioco non offre una mitologia ordinata e facilmente consultabile, ma un insieme di immagini, frasi e situazioni che sembrano appartenere a un sogno febbrile.
L'inglese talvolta incerto dei testi





contribuisce persino all'atmosfera. Alcune istruzioni possono risultare poco chiare, ma in un mondo popolato da dèi crudeli, mostri e tecnologie misteriose, anche la confusione linguistica sembra parte dell'esperienza.

Dal punto di vista del gameplay, Xibalba 64 guarda chiaramente ai primi sparatutto in prima persona. Il giocatore si muove rapidamente, raccoglie armi e affronta nemici in ambienti labirintici. Non esistono sistemi di copertura, ricariche elaborate o lunghe interruzioni narrative. La sopravvivenza dipende dal movimento, dalla precisione e dalla capacità di gestire le munizioni. Il design ricorda soprattutto i primi giochi del genere, ma non si limita a ricopiarne l'estetica. Le mappe alternano corridoi, stanze, passaggi nascosti e aree più aperte, invitando a esplorare ogni angolo alla ricerca di armi, segreti e vie alternative.

Il ritmo è generalmente rapido, anche se alcune sezioni possono risultare più lente quando il giocatore deve orientarsi o individuare l'accesso alla stanza successiva. L'assenza di una mappa non è necessariamente un difetto. Perdersi contribuisce alla sensazione di trovarsi in un luogo ostile e incomprensibile.

Armi e nemici

L'arsenale cresce progressivamente e offre armi sempre più potenti. Il giocatore deve scegliere quando utilizzare il proprio equipaggiamento, perché le munizioni non sono infinite e alcune situazioni richiedono una gestione attenta delle risorse. I nemici appartengono a categorie diverse e comprendono creature organiche, entità tecnologiche e avversari legati alla mitologia dell'ambientazione. Il loro comportamento non è sempre estremamente sofisticato, ma la varietà contribuisce a mantenere vivo il ritmo. Come nei classici del genere, eliminare tutto ciò che si muove non è sempre obbligatorio, ma raramente il giocatore resiste alla tentazione. In Xibalba 64 esplorare in silenzio è possibile; continuare a farlo dopo aver visto il primo nemico è molto meno probabile. Visivamente, il gioco utilizza una grafica low-poly che richiama l'estetica del Nintendo 64 e degli sparatutto degli anni Novanta. Le texture sono limitate, le forme geometriche essenziali e gli ambienti costruiti con poligoni relativamente semplici. Tuttavia, il risultato è efficace. Il basso dettaglio non appare come una mancanza, ma come una scelta coerente con l'hardware e con l'atmosfera. La giungla, le rovine

GIUDIZIO FINALE

» Giocabilità 88%

Trama misteriosa ma affascinante, gameplay solido. Forse troppo lineare nella struttura dei livelli.

» Longevità 85%

La trama inquieta ma affascina e il 17 mondi si esplorano volentieri e con grande curiosità.





e i corridoi tecnologici hanno un'identità visiva riconoscibile e l'illuminazione contribuisce a creare un senso di minaccia. Il limite più noto del Nintendo 64 riguarda la memoria texture, estremamente ridotta. La console dispone di appena 4 KB di texture memory nella pipeline grafica, con texture singole molto piccole. Per superare il problema, Szablewski ha ottimizzato l'ordine di rendering, raggruppando gli elementi con texture comuni e limitando il numero di caricamenti. Attraverso una serie di ottimizzazioni, il gioco riesce a mantenere una velocità stabile di 60 fotogrammi al secondo nella modalità singolo giocatore. È un risultato notevole per un progetto homebrew N64. La modalità a quattro giocatori non mantiene sempre i 60 fps, ma resta comunque fluida. Il gioco comprende diverse modalità multiplayer locali. La campagna principale può essere affrontata in cooperativa a due giocatori con schermo diviso. Questa modalità modifica sensibilmente l'esperienza: esplorare l'oltretomba insieme a un amico è più semplice in alcuni momenti, ma aumenta anche il caos durante gli scontri. È inoltre presente una modalità Bug Hunt per quattro giocatori, nella quale i partecipanti devono collaborare contro le creature del gioco. Infine, non poteva mancare il deathmatch, con combattimenti tutti contro tutti fino all'ultimo sopravvissuto. Le opzioni permettono di personalizzare le partite senza complicarle eccessivamente. Il risultato è particolarmente adatto alle serate in compagnia: poche regole, armi pesanti e quattro persone pronte a discutere su chi abbia sparato per primo.

La colonna sonora è composta da Andreas Lösch, già autore delle musiche dell'originale. I brani utilizzano atmosfere ambientali e inquietanti, più vicine all'horror psicologico che al semplice accompagnamento d'azione. L'audio è stato uno degli aspetti tecnicamente più impegnativi. Lo spazio della cartuccia è limitato e circa 31 MB dei 32 MB disponibili sono occupati da



musica ed effetti sonori. Per la riproduzione viene utilizzato il formato VADPCM, decodificato dal coprocessore RSP. L'autore ha inoltre sperimentato con il codec Opus e con un decoder accelerato, ma durante il gameplay risultava ancora troppo pesante per l'hardware. Una caratteristica curiosa è il lettore musicale integrato, che può essere sbloccato durante la campagna. È un piccolo extra che valorizza la colonna sonora e rappresenta un omaggio ai lettori bonus presenti in alcuni giochi dell'epoca.

La campagna comprende 17 livelli e numerosi segreti. Sono presenti difficoltà aggiuntive, trucchi e modalità sbloccabili, elementi che aumentano la rigiocabilità oltre il semplice completamento della storia. La struttura dei livelli incoraggia l'esplorazione. Chi si limita a seguire il percorso principale può arrivare rapidamente alla fine, ma chi cerca stanze nascoste e oggetti bonus trova ulteriori motivi per ripetere le missioni. Il gioco è quindi più ricco di quanto la sua struttura essenziale possa suggerire. Non punta sulla durata artificiale, ma sulla scoperta graduale dei suoi ambienti e delle sue modalità.

Xibalba 64 viene distribuito in una Collector's Edition che comprende cartuccia, confezione in stile rétro e manuale stampato. È compatibile con console Nintendo 64 PAL e NTSC e con il ModRetro M64.

La presenza del manuale è particolarmente apprezzabile. Non si tratta soltanto di un foglio di istruzioni, ma di un elemento che contribuisce all'esperienza complessiva. In un periodo dominato dalle versioni digitali, aprire una confezione, inserire una cartuccia e leggere un manuale cartaceo restituisce una sensazione ormai rara. Xibalba 64 è molto più di una curiosità per collezionisti. Il gioco non cerca di rivoluzionare il genere. Preferisce muoversi rapidamente, sparare, esplorare e lasciare che siano le immagini, la musica e i testi criptici a costruire l'esperienza. La mancanza di una mappa può disorientare, ma contribuisce alla sensazione di trovarsi davvero in un luogo sconosciuto.

Il comparto tecnico è il suo punto di forza più evidente. Riuscire a mantenere 60 fotogrammi al secondo in singolo giocatore, gestire texture limitate, ambienti ampi, musica compressa e modalità multiplayer su hardware N64 è un risultato notevole. Un FPS rétro inquietante, tecnico e pieno di personalità. Xibalba 64 dimostra che il Nintendo 64 può ancora ospitare nuove avventure e che, nell'oltretomba maya, anche perdersi è probabilmente parte del rituale.

di **Carlo Nithaiah Del Mar Pirazzini**

ROGUECRAFT DX

Anno: 2026
Editore/Sviluppatore: Thalamus Digital Publishing, Badger Punch Games
Genere: Roguelike
Piattaforma: Windows
Sito web: https://thalamusdigital.itch.io/roguecraft-dx-pc



Ci sono roguelike che richiedono decine di ore, manuali più lunghi di un'enciclopedia e un inventario organizzato meglio di un supermercato. Poi c'è Roguecraft DX, che preferisce una filosofia più semplice: entrare in un dungeon, trovare la chiave, picchiare qualche mostro, bere pozioni dall'identità discutibile e morire con dignità. Sviluppato da Badger Punch Games e pubblicato da Thalamus Digital, Roguecraft DX nasce da un progetto Commodore 64 realizzato per una competizione di programmazione da 4 KB. Da quell'esperimento vide luce prima Rogue64, poi trasformato in Roguecraft per Amiga e successivamente ampliato nelle versioni DX per Evercade, Amiga, Mega65 e Windows. L'edizione PC, pubblicata su Steam il 10 settembre 2026, rappresenta la versione più completa del progetto.

L'obiettivo è attraversare le profondità di Mordecoom, un livello alla volta, cercando la chiave che permette di scendere al piano successivo. Il dungeon viene generato proceduralmente a ogni partita: stanze, percorsi, oggetti e nemici cambiano continuamente. Il movimento e i combattimenti sono a turni. Il sistema è accessibile in pochi minuti, ma richiede più tempo per essere padroneggiato. Combattere frontalmente non è sempre la scelta migliore: spesso conviene attirare un mostro sulle punte, condurlo attraverso il fuoco o sfruttare l'ambiente per evitare lo scontro diretto. La profondità non nasce da decine di statistiche, ma dalla lettura della stanza e dalla capacità di prevedere il movimento dei nemici. Il gioco non è complicato, ma può punire con estrema rapidità una decisione presa troppo in fretta. La scelta del personaggio determina anche il livello di difficoltà. Il Guerriero è la scelta più semplice. Ha più salute, maggiore resistenza e può aumentare forza e punti vita più degli altri personaggi. È ideale per imparare le regole e affrontare i primi dungeon

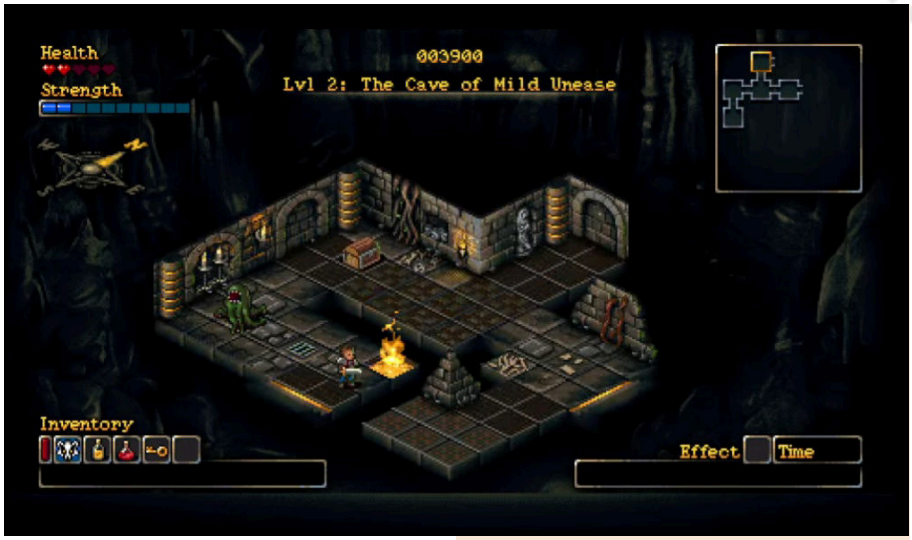




senza essere eliminati dal primo pollo di passaggio. Il Ladro occupa una posizione intermedia. È più fragile del Guerriero, ma può teletrasportarsi alle spalle dei nemici e ottenere un potente bonus per gli attacchi furtivi. Usarlo bene richiede una maggiore attenzione al posizionamento, ma permette di risolvere molti scontri con eleganza. Il Mago è il personaggio più difficile. Possiede meno salute e forza, ma può lanciare attacchi a distanza. Questa abilità consente di eliminare i nemici senza entrare nel loro raggio d'azione, purché il giocatore riesca a mantenere le distanze. Durante l'esplorazione si trovano pozioni dai colori e dagli effetti inizialmente sconosciuti. Possono curare, aumentare le statistiche, garantire protezione, permettere il teletrasporto o liberare una stanza con un potente attacco elettrico.

La particolarità è che gli effetti cambiano da una partita all'altra. Una pozione rossa identificata durante una discesa manterrà lo stesso effetto fino alla morte del personaggio, ma potrebbe comportarsi in modo completamente diverso nella partita successiva. Questo sistema crea una piccola componente sperimentale. Bere una pozione sconosciuta può salvarci oppure peggiorare una situazione già pericolosa. Quando la salute è bassa, però, la curiosità diventa rapidamente una forma di pessimismo. Oltre alle pozioni sono presenti potenziamenti per salute e forza, oro e altri oggetti. La gestione dell'inventario rimane volutamente semplice, in linea con la filosofia generale del gioco. Poche regole, ma applicate in modo chiaro.

Uno degli elementi più curiosi di Roguecraft DX è il famiglio. Durante una partita è possibile trovare una creatura chiusa in gabbia, liberarla e conservarla nell'inventario. Il famiglio può essere evocato per combattere accanto al protagonista o recuperare oggetti difficili da raggiungere. La sua presenza, tuttavia, dura soltanto alcuni



turni. Bisogna quindi scegliere il momento giusto per utilizzarlo. Evocarlo troppo presto significa sprecarne l'aiuto, aspettare troppo può voler dire morire prima di poterlo chiamare.

E sì, tra i possibili compagni c'è anche un tasso. È probabilmente il tipo di creatura che ci si aspetta di trovare in un dungeon lovecraftiano dopo aver già incontrato mostri tentacolari, pozioni sospette e polli armati di una personalità spropositata.



La campagna principale della versione DX comprende 15 livelli, cinque in più rispetto all'edizione originale. Il primo è The Wilderness, un'area all'aperto che precede la discesa vera e propria nel dungeon. Sono stati inoltre aggiunti nuovi nemici, nuove pozioni, più animazioni e ambientazioni graficamente più dettagliate.

L'edizione Windows introduce anche due modalità inedite. C'è l'Endless



Dungeon, una modalità senza un numero prestabilito di piani. Più si scende, più la difficoltà aumenta, fino a quando il dungeon decide che il giocatore ha avuto abbastanza speranze. La seconda è Chicken Run, una modalità nella quale si controlla un pollo invece di un Guerriero, un Ladro o un Mago. Il pollo ha valori di salute e forza molto bassi, quindi bisogna cercare uova curative e potenziamenti temporanei. È una modalità particolare e più difficile, ma perfettamente coerente con l'umorismo del gioco. La campagna standard richiede circa 30-40 minuti per una partita, in base all'abilità del giocatore. È una durata ideale per una sessione breve, un viaggio in treno o una pausa durante la quale si dovrebbe lavorare ma si preferisce affrontare un'altra stanza piena di mostri.

La grafica utilizza pixel art isometrica e riesce a dare al dungeon un aspetto riconoscibile. Le pareti, le stanze, le trappole e gli oggetti hanno un buon livello di dettaglio, mentre i mostri sono animati con sufficiente personalità. L'atmosfera lovecraftiana viene alleggerita da elementi più bizzarri e umoristici. Le creature tentacolari convivono con polli aggressivi e con la presenza surreale del tasso. Il risultato evita di diventare troppo cupo e mantiene un tono da avventura fantasy rétro.

La mappa automatica è utile per orientarsi e permette di visualizzare la struttura dei livelli. Il comparto sonoro richiama inoltre la tradizione Amiga, completando l'identità retro del progetto. Arriviamo agli errori che erano presenti anche nelle versioni precedenti. Il principale difetto riguarda i movimenti diagonali. La visuale isometrica non sempre rende intuitiva la direzione dei comandi e può capitare di premere un input pensando di muoversi verso un lato, finendo invece per attaccare nella direzione sbagliata.In un gioco a turni, anche un errore apparentemente piccolo può avere conseguenze importanti. Un



movimento sbagliato concede infatti un turno ai nemici, che possono approfittarne per attaccare.Anche l'assenza di una progressione permanente e di salvataggi tradizionali può dividere il pubblico. La struttura segue il modello classico del roguelike, quando il personaggio muore, la partita ricomincia. La durata ridotta delle run rende il limite meno pesante, ma chi preferisce conservare ogni progresso potrebbe trovarlo frustrante.

Arrivando al sodo della recensione questa versione DX per Windows è più completa e raffinata e diciamo che aggiunge due nuove modalità estreme ma che mettono "pepe" all'avventura globale. È facile da imparare, difficile da padroneggiare e abbastanza ironico da far accettare persino l'ennesima morte causata da una scelta apparentemente innocua.

di **Carlo Nithaiah Del Mar Pirazzini**

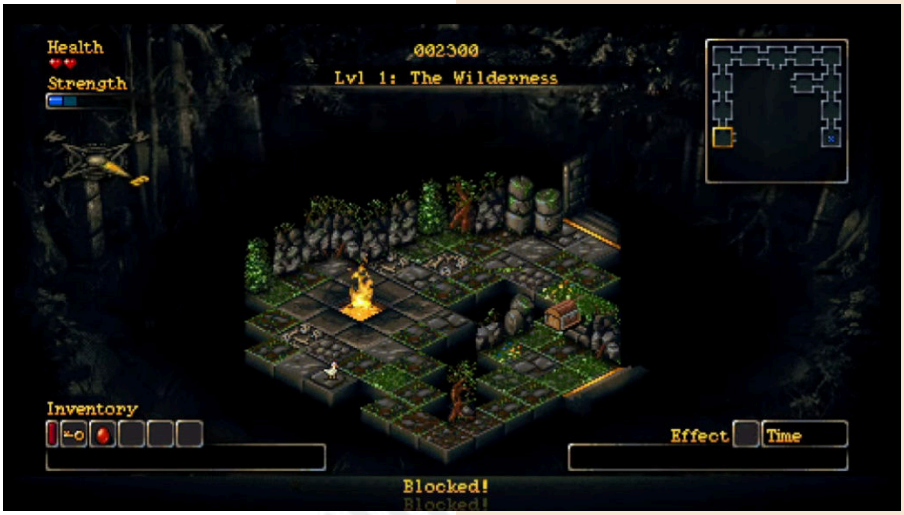
GIUDIZIO FINALE

» Giocabilità 90%

Semplice da apprendere e con un ottimo level design nella versione "campagna classica". Interessante l'Endless Dungeon e folle la campagna col pollo da guerra.

» Longevità 90%

E' un gioco difficile come tutti i roguelike e forse non adatto a chi cerca il classico gioco d'avventura o gioco di ruolo. Resta un titolo molto longevo anche dopo averlo portato a termine una volta.



L'ABBAYE DES MORTS

Anno: 2026
Editore/Sviluppatore: Shrek/MB Maniax
Genere: Action/Platform
Piattaforma: ZX Spectrum NEXT
Sito web: https://shrek128.itch.io/labbaye-des-morts

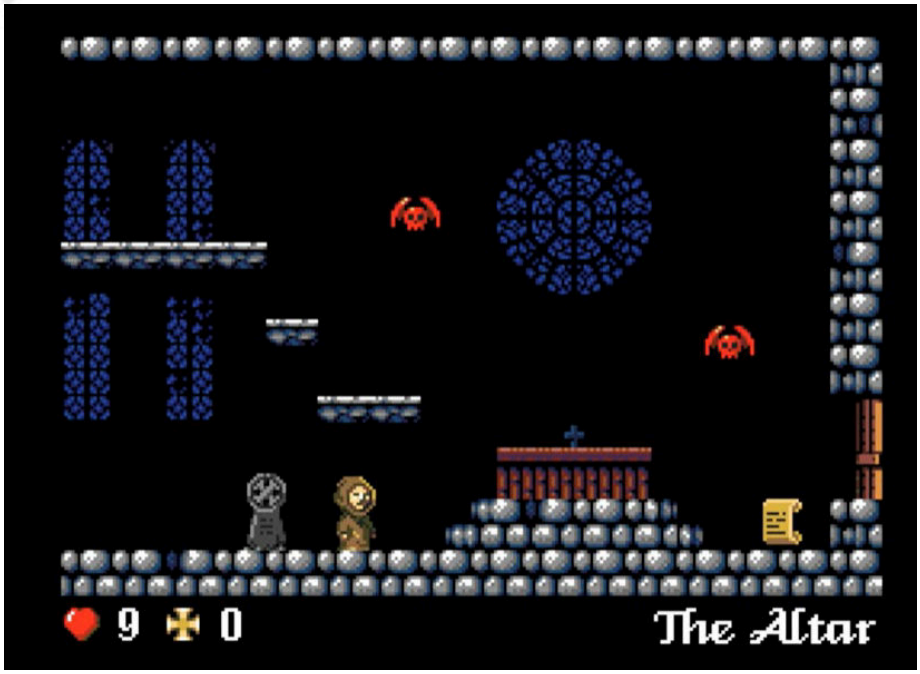
Esistono luoghi nei quali sarebbe meglio non entrare. Cantine umide, catacombe dimenticate, certi bagni degli autogrill e, naturalmente, l'abbazia di L'Abbaye des Morts. Il piccolo classico creato da Locomalito nel 2010 torna sullo ZX Spectrum Next grazie a Shrek/MB Maniax, in una conversione realizzata per il ByteFest 2026 che sembra quasi chiudere un cerchio: un gioco moderno nato per imitare lo Spectrum finisce finalmente su uno Spectrum… dopato!!!



La storia ci porta nel XIII secolo. Jean Raymond, cataro in fuga dai crociati, trova rifugio in una vecchia chiesa abbandonata. Ottima idea, Jean. Peccato che sotto il pavimento si nascondano mostri, trappole e qualcosa decisamente meno accomodante di un parroco irritato. Per uscirne vivo dovremo esplorare 23 schermate interconnesse, recuperare dodici croci catare, interpretare gli indizi lasciati sulle pergamene e soprattutto imparare quando saltare, abbassarsi o scappare con grande dignità. Armi? Nessuna. La fede sarà anche potente, ma contro una fiammata serve soprattutto un buon tempismo.

La struttura rimane quella deliziosamente rétro dell'originale con la sua esplorazione non lineare, le piattaforme, gli enigmi ambientali e quel pizzico di backtracking che trasforma ogni stanza in un luogo da memorizzare. È un'avventura breve,





GIUDIZIO FINALE

» Giocabilità 85%
Gameplay solido e godibilissimo.

» Longevità 85%
Non molto lungo ma sempre divertente… anche qui su NEXT.

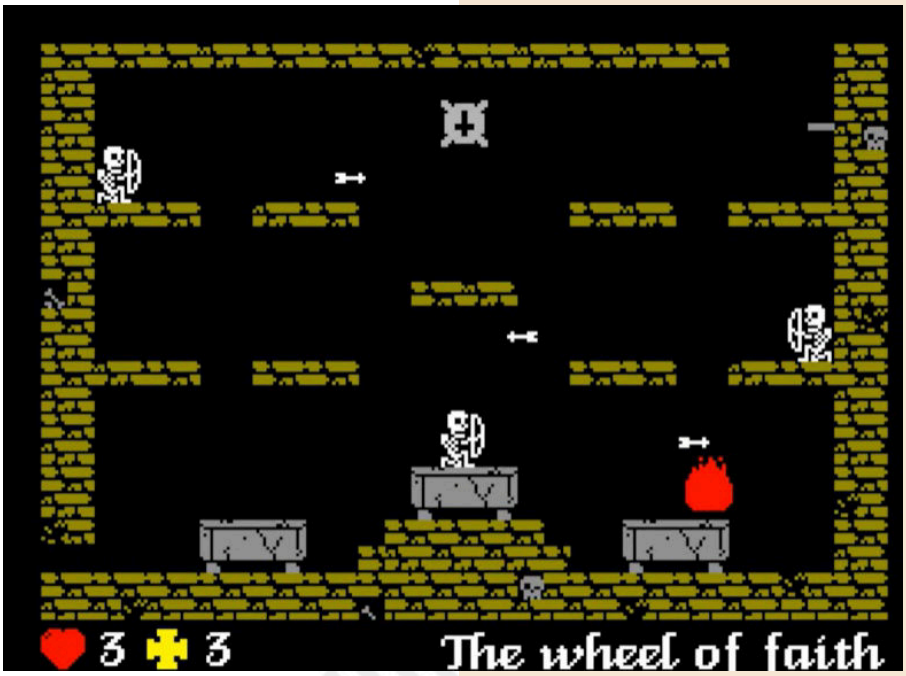
circa mezz'ora una volta capiti i suoi segreti, ma intensa e costruita senza schermate inutili, caratteristica apprezzata anche nelle recensioni delle precedenti incarnazioni.

Sul Next il gioco acquista però una marcia in più. La risoluzione resta fedelmente a 256×192, ma entrano in scena Layer 2, sprite hardware, sonoro adattato e soprattutto due set grafici selezionabili al volo: il ruvido look 8-bit oppure una veste 16-bit più ricca e ombreggiata. È come restaurare un'abbazia medievale lasciando al visitatore l'interruttore per scegliere se guardarla nel 1984 o con gli occhiali nuovi.

Oscuro, essenziale e ancora sorprendentemente affascinante, L'Abbaye des Morts dimostra perché, dopo sedici anni e conversioni per numerosi sistemi, continui a risorgere. In fondo, con un titolo simile, restare morto sarebbe stato decisamente fuori tema.

di **Carlo Nithaiah Del Mar Pirazzini**





ORPHEUS TO HELL AND BACK

Anno: 2025
Editore/Sviluppatore: Studio Loading
Genere: ARPG
Piattaforma: Game Boy
Sito web: https://studioloading.itch.io/orpheus

Nelle profondità dell'Ade, dove le ombre si muovono tra fiamme eterne e creature infernali sorvegliano ogni sentiero, un solo eroe osa sfidare il regno dei morti. Il suo nome è Orfeo e la sua arma non è una spada, ma una lira capace di piegare il comportamento dei nemici e trasformare il pericolo in uno strumento.

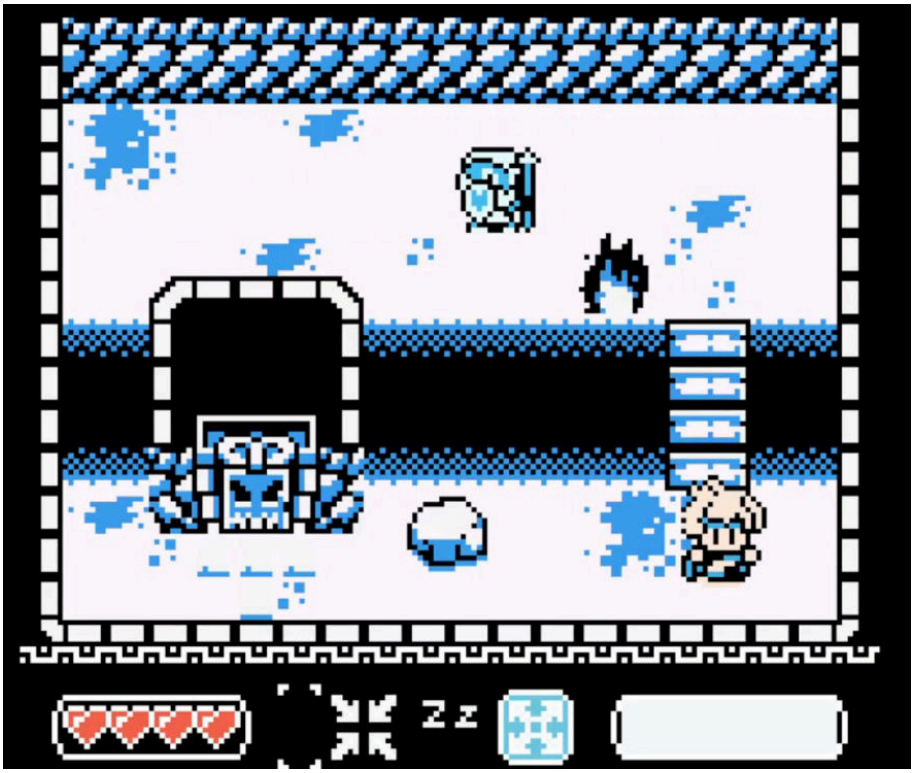
Orpheus: To Hell and Back porta questa celebre discesa agli inferi sul piccolo schermo del Game Boy, trasformandola in un'avventura compatta fatta di enigmi, trappole e creature da manipolare. Con una grafica essenziale, un'atmosfera oscura e una struttura che richiama i classici portatili, il gioco propone un viaggio breve ma ricco di personalità, nel quale ogni melodia può rappresentare la salvezza oppure l'ultimo errore commesso prima di finire tra le fiamme dell'oltretomba. Il protagonista è Orfeo, impegnato a salvare la propria amata Euridice dalle profondità dell'Ade. La storia viene raccontata attraverso poche schermate di testo e lascia presto spazio all'esplorazione di una serie di ambienti pieni di nemici, ostacoli, trappole e rompicapi. L'idea più interessante riguarda la lira del nostro eroe. Il protagonista può utilizzare tre melodie differenti: una serve a stordire i nemici, una li attira verso di sé e un'altra li respinge. Gli avversari, compresi i boss, seguono percorsi prestabiliti e il giocatore deve sfruttare le melodie per spingerli verso il fuoco, dentro una trappola o oltre un precipizio. Il sistema è semplice da comprendere, ma richiede un po' di pratica. I comandi sono ridotti all'essenziale, come da tradizione Game Boy. Il pad direzionale serve per muoversi, mentre i pulsanti A e B permettono di selezionare e utilizzare le abilità. Il problema è che il gioco spiega poco il funzionamento della lira, lasciando al giocatore il compito di sperimentare.

Una volta compreso il tutto, i livelli offrono situazioni divertenti e originali. I nemici diventano strumenti da utilizzare per risolvere gli enigmi e ogni schermata richiede di osservare i movimenti, scegliere il momento giusto e dirigere le creature verso il pericolo. È un approccio più vicino al









puzzle game che all'azione tradizionale. L'esperienza, però, non è sempre precisa. Il rilevamento delle collisioni è piuttosto severo e Orfeo sembra occupare molto spazio rispetto alle dimensioni ridotte delle schermate. È quindi facile toccare accidentalmente un nemico o finire in una trappola, soprattutto durante le fasi più concitate.

Le battaglie contro i boss risentono ancora di più di questo problema. Le situazioni diventano caotiche e, in alcuni momenti, il giocatore sa perfettamente cosa dovrebbe fare, ma viene penalizzato da un movimento leggermente troppo ambizioso o da una collisione poco chiara. Questi difetti possono generare una certa frustrazione, anche perché la difficoltà non deriva sempre dalla complessità dei rompicapi. A volte sembra piuttosto il risultato di controlli rigidi e di una gestione degli spazi non del tutto precisa. D'altra parte, questa imperfezione contribuisce anche al fascino rétro del gioco. Orpheus sembra davvero un titolo uscito da un'epoca in cui il giocatore doveva imparare tutto attraverso tentativi, errori e qualche inevitabile imprecazione.La durata è contenuta. L'avventura può essere completata in circa un paio d'ore, anche considerando i diversi fallimenti e i tentativi necessari per superare alcune sezioni. Per alcuni giocatori sarà un limite; per altri, una scelta appropriata. Orpheus: To Hell and Back non vuole essere un viaggio enorme nell'oltretomba, ma una piccola avventura da affrontare in brevi sessioni. Il gioco è stato pubblicato in formato digitale e fisico, una scelta apprezzabile per chi ama collezionare produzioni indipendenti su cartuccia.

di **Ingrid Poggiali**

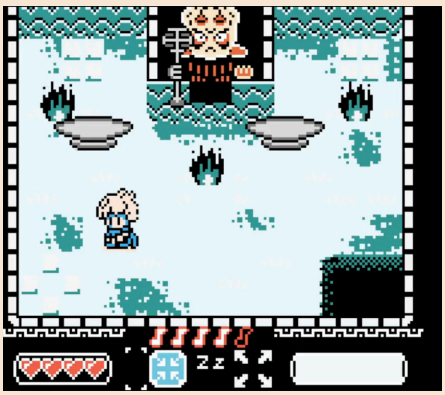
GIUDIZIO FINALE

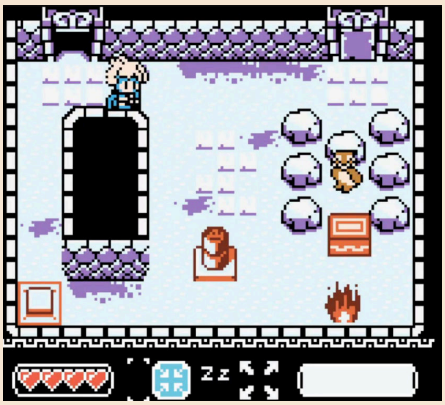
» Giocabilità 75%

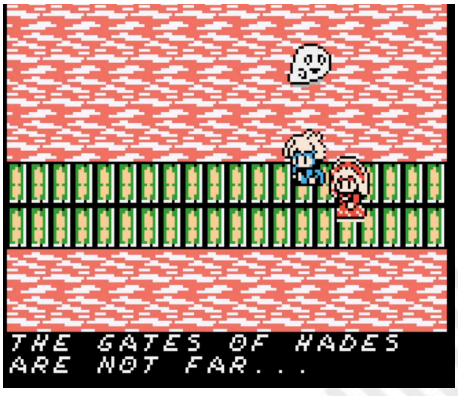
Il sistema di gioco risente di alcune imperfezioni nella collisione tra sprite e forse le meccaniche andavano spiegate in modo più "preciso".

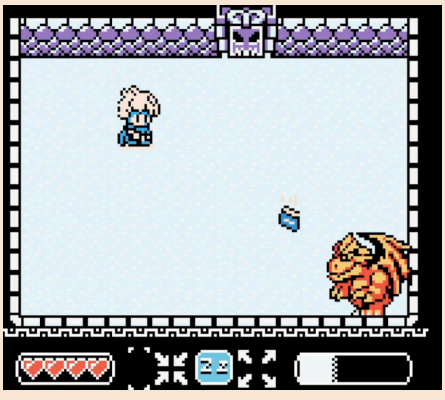
» Longevità 80%

Non lungo da portare a termine ma piuttosto soddisfacente una volta appreso l'utilizzo della lira.









KABOOMANIA

Anno: 2026
Editore/Sviluppatore: Fancy Factory
Genere: Maze Game
Piattaforma: Amiga OCS/ECS/AGA, PC, XBOX, PS5, PS4, Switch, macOS
Sito web: https://www.fancy-factory.com/kaboomania/

Oy Bimbi! Prendete Bomberman, il vostro Amiga 1200 (o un emulatore), aggiungete mille livelli, una quantità sospetta di esplosivi e una colonna di Chris Hulsbeck, ottenete… Kaboomania. Il nuovo titolo di Fancy Factory che è un maze bomber game dal cuore decisamente retrò, nato per Amiga e successivamente portato su Pc e console moderne.



L'idea è semplice e sicuramente l'avrete già vista da qualche parte (nel titolo di Hudson soft???). Muoversi in labirinti pieni di blocchi, nemici e bonus, piazzare dinamite e cercare di non trovarsi dalla parte sbagliata dell'esplosione. In altre parole, il vero nemico non è sempre il mostro davanti a noi, ma spesso il giocatore che cinque secondi prima ha pensato: "Tranquillo, questa bomba la controllo".

Il gioco nasce e cresce su Amiga e questa origine si nota subito. La grafica è una bella e colorata pixel art con una risoluzione autenticamente ed autarchicamente retrò: 128 colori nella versione AGA e 32 su macchine ECS/OCS. Il risultato è un bel tripudio di colori e fondali vivaci, nemici bislacchi e stravaganti, bonus di tutti i tipi e ambientazioni che ricordano i bei tempi gloriosi del bombarolo su Amiga nei primi anni Novanta.

Il gioco è disponibile su A500, A1200, CD32 e in formato WHDLoad, oltre che su PC e console moderne. La versione AGA gira su sistema PAL e, per una maggiore fluidità nelle mappe più grandi e nel multiplayer a quattro giocatori, consigliamo almeno 2 MB di Fast aggiuntiva e un processore 030. Il sistema di gioco è quello immediato dei giochi di Bomberman. Il protagonsita, Scott Whiskers, deve attraversare labirinti distruggendo blocchi e nemici con la dinamite. Le esplosioni si propagano in quattro direzioni e possono aprire nuovi passaggi, eliminare avversari e rivelare potenziamenti. Naturalmente, la dinamite non distingue tra amici, nemici e chi l'ha appena piazzata. Il rischio di rimanere intrappolati nella propria esplosione è sempre presente, soprattutto quando aumentano il raggio d'azione e il numero di bombe disponibili. All'inizio le mappe sembrano semplici, ma presto richiedono una certa attenzione. Bisogna capire dove posizionare gli ordigni, prevedere i tempi dell'esplosione e lasciare sempre una via di fuga.

Devo ammettere che leggere 1000 livelli impressiona non poco. Tutto questo "ben di Dio di longevità" è distribuito in cinque mondi tematici. E' chiaro che il gioco non si esaurisce dopo un breve serata. Tutti i quadri sono realizzati a mano e presentano ambientazioni differenti, con nemici dotati di comportamenti diversi,





ostacoli, bonus e situazioni più elaborate. C'è un po' di ripetitività a dire il vero, ma è quasi impercettibile. La campagna, però, segue una struttura molto tradizionale. Non ci sono sistemi complessi di progressione, alberi delle abilità o meccaniche inserite per forza in un gioco che non ne ha bisogno. Kaboomania preferisce concentrarsi sulla sua idea centrale, ovvero far piazza pulita e proseguire al prossimo labirinto. Da una parte rende l'esperienza immediata e coerente; dall'altra può diventare ripetitiva durante sessioni molto lunghe.

Senza ombra di dubbio la modalità Multiplayer è la più divertente. E' un multiplayer locale, che permette un massimo di quattro giocatori su schermo. Qui il gioco smette di essere semplicemente "un gioco" ma diventa una macchina per produrre accuse, risate e sospensioni di amicizie, amori e rapporti umani.

Altro pezzo forte è il coinvolgimento di Hulsbeck nella colonna sonora, vero pezzo forte con ben 14 tracce di alto livello.

Concludendo Kaboomania non cerca di nascondere la propria natura. È un gioco colorato, rumoroso e orgogliosamente rétro, costruito intorno a una formula che conosciamo da decenni: piazzare bombe, distruggere blocchi, eliminare nemici e cercare di non morire a causa della propria brillante strategia. Il gioco funziona soprattutto quando viene affrontato in brevi sessioni o in compagnia. La campagna offre una quantità enorme di livelli, ma il multiplayer locale è probabilmente il punto in cui la formula dà il meglio di sé. Qui l'azione è immediata, le partite sono comprensibili da subito e ogni esplosione può generare una reazione esagerata da parte di chi è appena stato eliminato.

L'assenza di grandi innovazioni può essere vista come un difetto, ma anche come una scelta precisa. Kaboomania non vuole trasformare il maze bomber in un gioco di ruolo, in un roguelike o in una piattaforma piena di sistemi superflui. Vuole essere un gioco Amiga colorato e divertente, con un'enorme quantità di livelli e una colonna sonora firmata da uno dei compositori più importanti della scena.

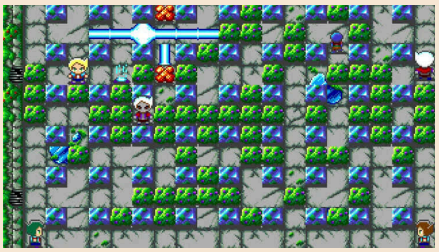
di **Carlo Nithaiah Del Mar Pirazzini**

GIUDIZIO FINALE

» Giocabilità 85%
Un maze bomber come Bomberman ma con 1000 livelli e una caterva di bonus... libidine!

» Longevità 80%
Sicuramente non innovativo... ma chi se ne frega!









MAPPY

Anno: 2026
Editore/Sviluppatore: charger0277
Genere: Platform
Piattaforma: Commodore 64 (REU)
Sito web: https://charger0277.itch.io/mappy



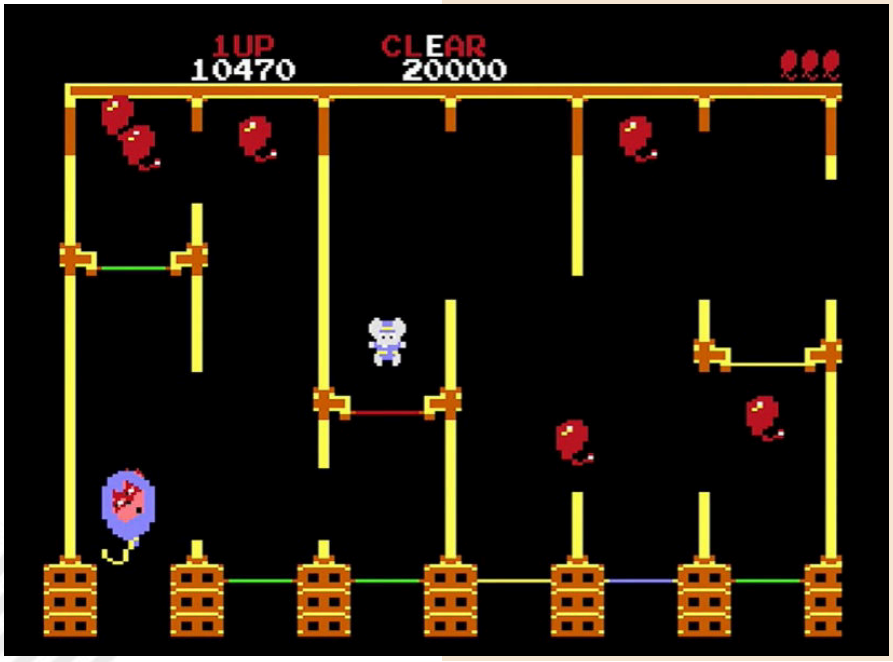
Nel 1983 Namco decise di riscrivere le leggi della natura: i gatti diventavano ladri professionisti e un topo indossava il distintivo della polizia. Nasceva così Mappy, uno degli arcade più irresistibili dell'epoca, dove la storica rivalità tra felini e roditori si trasformava in un inseguimento degno di un cartone animato impazzito!

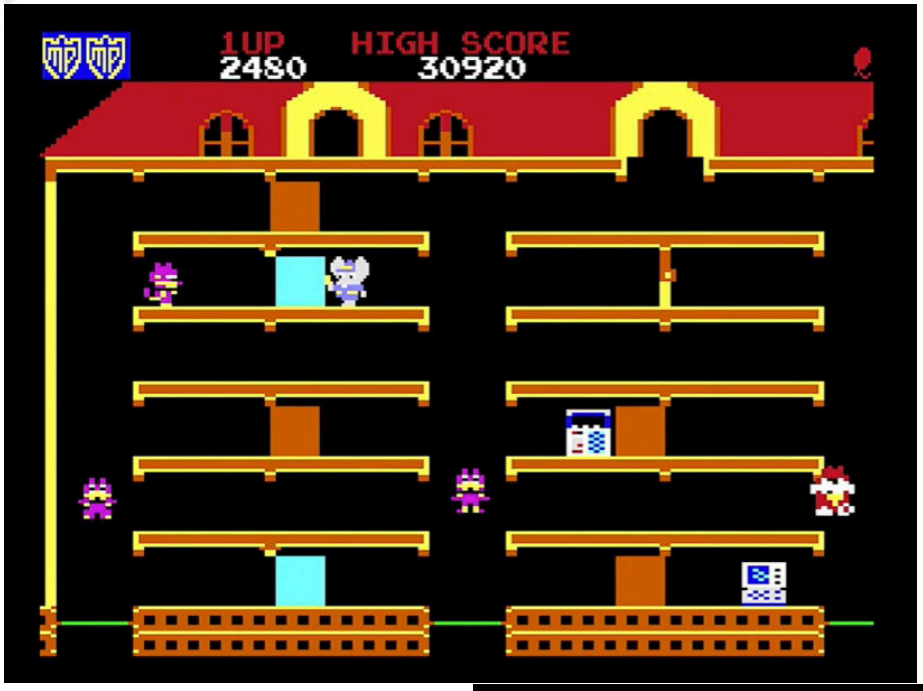
Oggi, grazie al nostro connazionale charger0277, questa deliziosa follia approda sul Commodore 64 con una conversione che promette di riportare l'esperienza originale sul nostro amato biscottone.

Il protagonista è Mappy, un coraggioso topo poliziotto incaricato di recuperare la refurtiva nascosta nella villa del perfido gatto Goro e della sua banda di Meowkies. Peccato che i felini non abbiano alcuna intenzione di collaborare e considerino il nostro eroe più un antipasto che un rappresentante della legge!
La soluzione? Correre, saltare sui trampolini e sbattere porte in faccia ai gatti. SBAM! BOING! MIAOOO! Altro che Tom e Jerry: qui il topo rischia la pelliccia a ogni piano!

Il gameplay conserva tutta la frenesia dell'arcade Namco, con sedici abitazioni, livelli bonus, palloncini da scoppiare e porte speciali capaci di spedire i mici a fare un viaggio non programmato. Attenzione però ai trampolini: dopo troppi rimbalzi cedono, trasformando Mappy in una sottiletta al formaggio!







GIUDIZIO FINALE

» Giocabilità 90%
C'è tutto l'arcade e tutto il suo sistema di controllo frenetico e folle. Giocabilissimo e godibilissimo con un reparto tecnico d'eccellenza.

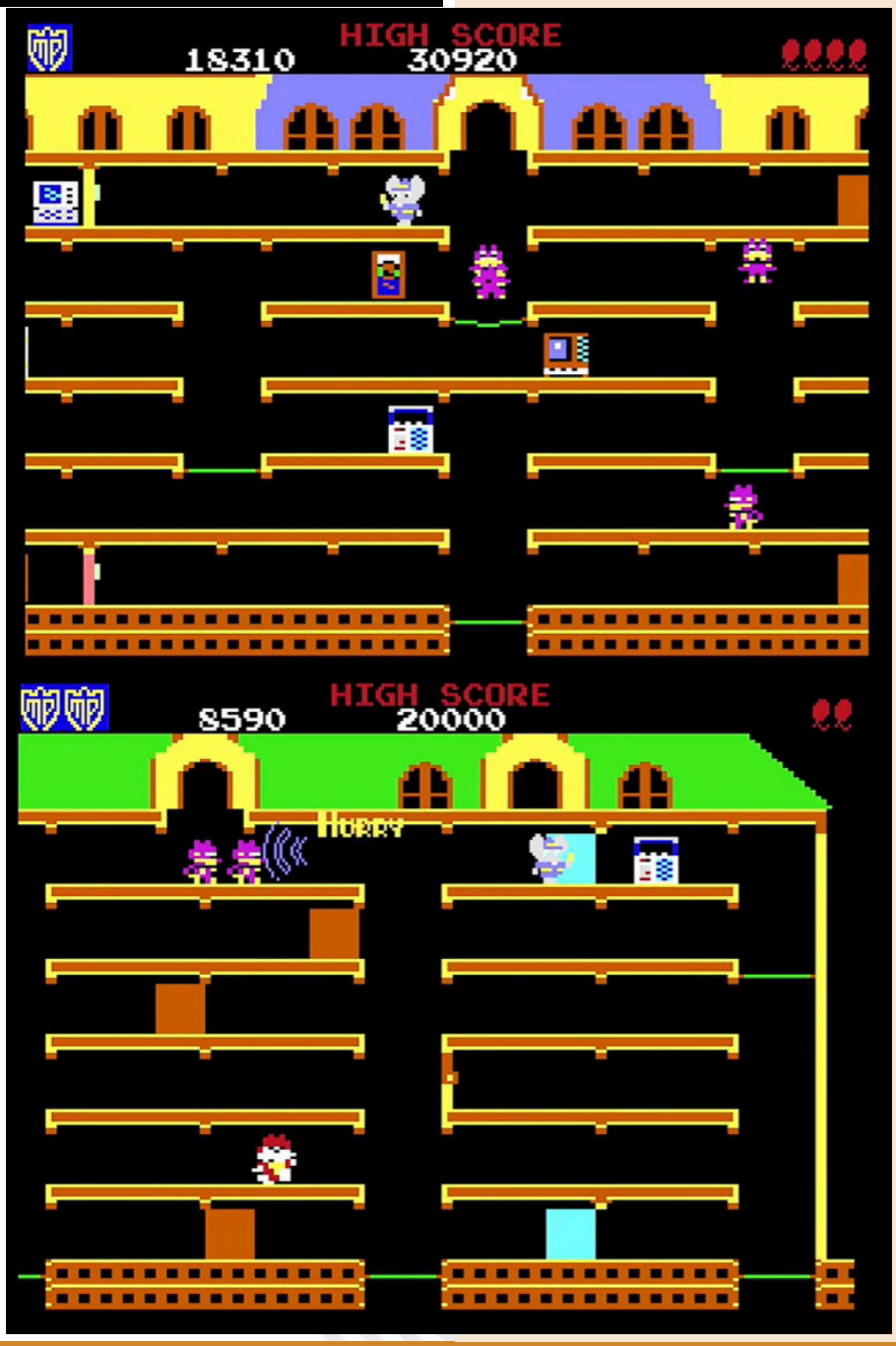
» Longevità 85%
E' un titolo della "vecchia" generazione con tutte le sue spietatezze schierate in campo.

Tecnicamente il lavoro è notevole con uno scrolling fluido, animazioni fedeli e una ricostruzione delle dinamiche originali verificata attraverso oltre mille registrazioni dell'arcade. Il tutto gira a 50 o 60 fotogrammi al secondo, secondo il sistema utilizzato. Spettacolare!!!

C'è però un piccolo prezzo da pagare a tutto questo libidinoso ben di Dio!!! Serve una REU da almeno 1 MB, reale o emulata. Il vecchio C64, evidentemente, ha deciso di concedersi un ampliamento di memoria prima di affrontare l'invasione felina.

Mappy rimane un concentrato di divertimento immediato, riflessi e sana cattiveria arcade. Una conversione gratuita che restituisce al Commodore 64 un classico irresistibile.

di **Carlo Nithaiah Del Mar Pirazzini**



WORMS
THE DIRECTOR'S CUT 1.5

Anno: 2026
Editore/Sviluppatore: Andy Davidson
Genere: Battle game (espansione di Worms: The Director's Cut)
Piattaforma: Amiga AGA
Sito web: https://worm.net/

Alcuni videogiochi invecchiano con eleganza e altri che non si possono guardare per quanto sono "orridi"... poi c'è Worms... che invece ci fa godere perché non invecchia mai anzi continua a farci lanciare granate addosso come se il tempo non fosse mai passato, e lo fa ancora con grande classe ed eleganza. A quasi trent'anni dalla sua pubblicazione su Amiga, il celebre gioco strategico a turni è tornato a far parlare di sé grazie a una nuova espansione gratuita, sviluppata con il coinvolgimento dell'autore originale Andy Davidson e della comunità Amiga.

Il pacchetto, conosciuto durante lo sviluppo come Worms DC 1.5 e pubblicato per celebrare il trentesimo anniversario della serie, aggiunge oltre 40 nuovi livelli, più di 50 pacchetti vocali, 10 terreni personalizzati, nuove squadre e numerosi contenuti creati da gruppi della demo scene. Il tutto è disponibile gratuitamente dal sito ufficiale Worm.net. L'aspetto più interessante del progetto non è soltanto la quantità dei contenuti, ma il modo in cui sono stati realizzati. Andy Davidson ha coinvolto diversi gruppi della scena Amiga, fornendo loro indicazioni e livelli di esempio da utilizzare come base. Ogni gruppo ha potuto quindi progettare le proprie mappe, contribuendo con idee, stile grafico e una buona dose di cattiveria strategica.Il risultato è una raccolta che conserva un forte carattere comunitario. Non siamo davanti a un'espansione prodotta da un grande team commerciale, ma a una sorta di album collettivo dedicato a Worms e all'Amiga. Ogni livello riflette una diversa sensibilità, proprio come accadeva con le demo, le riviste e il software amatoriale dell'epoca. Tra i contenuti è presente anche un livello ispirato a Ruff 'N Tumble, realizzato con il permesso dei detentori dei diritti. Era stato progettato anche un omaggio alla serie televisiva Red Dwarf, ma il contenuto non è stato incluso a causa di problemi legati alle autorizzazioni.

Le nuove mappe offrono ambientazioni diverse e situazioni pensate per mettere alla prova sia i giocatori esperti sia quelli che vogliono semplicemente tornare a giocare con gli amici. Questa aggiunta aumenta notevolmente la longevità del gioco, soprattutto perché ogni scenario può essere affrontato con squadre, armi e strategie differenti. Un altro elemento importante è rappresentato dagli oltre 50 nuovi sample pack vocali. In Worms, le voci dei vermi sono parte integrante dell'identità del gioco... diciamocelo subito... un attacco non è completo senza un'imprecazione esagerata, una protesta ridicola o un'esultanza fuori luogo.

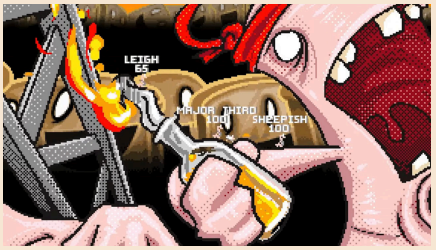
L'espansione include pacchetti ispirati a Worms 2, Worms Armageddon e Worms World Party, oltre a set caratterizzati da accenti e personalità differenti. Sono presenti, tra gli altri, voci americane, australiane, francesi, tedesche, italiane, russe, spagnole, svedesi e scozzesi. Non mancano inoltre pacchetti più particolari, come Drill Sergeant, Double-Oh-Seven, Formula One, Soul Man, Stooges, Kidz e Cyberworms.

Questi contenuti non modificano le regole di gioco, ma cambiano il tono delle partite. Un verme che parla come un sergente trasforma un semplice turno in un'operazione militare. Un verme con una voce improbabile, invece, può far ridere anche dopo essere stato centrato da una pecora esplosiva.

Tra i contenuti più curiosi compare

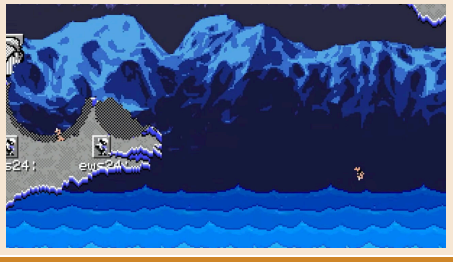


un livello realizzato in collaborazione con Jeff Minter, fondatore di Llamasoft e figura storica della scena videoludica indipendente. L'omaggio è particolarmente appropriato. Minter ha sempre rappresentato un approccio personale, sperimentale e spesso psichedelico allo sviluppo, mentre Worms è uno dei giochi più riconoscibili della cultura Amiga. L'incontro tra i due mondi aggiunge un elemento celebrativo e collega diverse generazioni della scena indipendente.



Il progetto include inoltre un album remix composto da nove brani ispirati al tema originale di Worms, scritto da Bjørn Lynne. La musica è uno degli elementi più riconoscibili del gioco e le nuove versioni contribuiscono a trasformare l'espansione in una celebrazione completa, non limitata ai soli livelli.

Ciliegina sulla torta è l'editor di livelli online dove è possibile realizzare mappe, terreni personalizzati,





GIUDIZIO FINALE

» Giocabilità 95%

Semplice e distruttivo e ancora più divertente. Nessuna texture moderna, ma tantissimo da provare e riprovare e con una marea di missioni in più… e poi l'editor dei livelli è una chicca!

» Longevità 95%

L'espansione aggiunge contenuti senza modificare la struttura del gioco. Il campo di battaglia resta bidimensionale, il turno continua a essere regolato da vento e traiettorie e ogni attacco può produrre risultati completamente diversi da quelli previsti. La fisica non è sempre realistica e proprio per questo è divertente… e poi… vi ho già detto dell'eccellente editor vero?



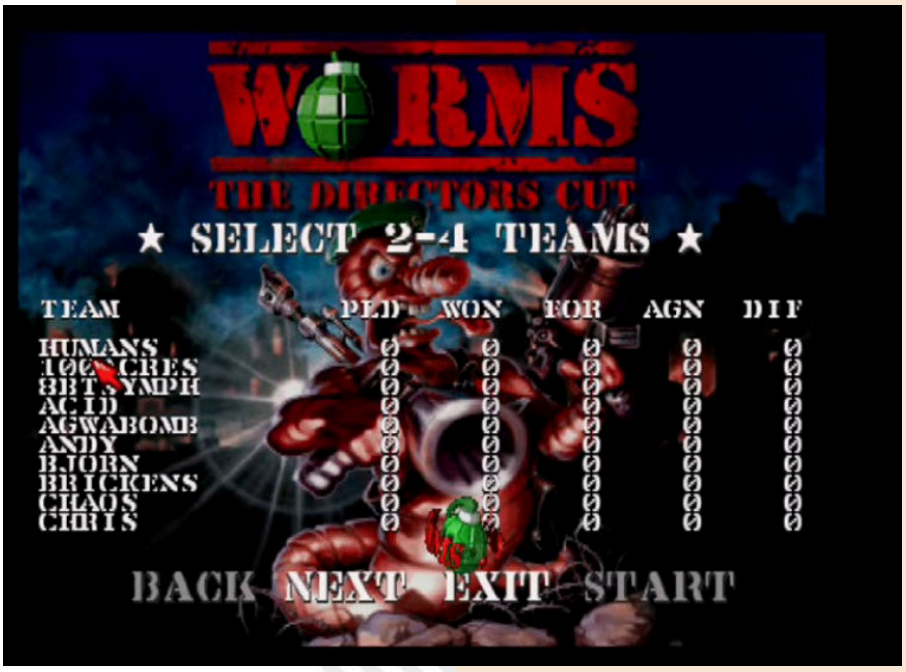
mountain set e nuove squadre. L'editor rende più semplice progettare nuovi scenari senza dover ricorrere a programmi complessi o modificare manualmente i file del gioco. Gli utenti possono creare un campo di battaglia equilibrato oppure progettare una mappa completamente assurda, con piattaforme impossibili, corridoi stretti e posizioni di partenza pensate per mettere subito tutti nei guai. Per un gioco basato sulla distruzione del paesaggio, questa possibilità è particolarmente importante. Il terreno non è un semplice elemento decorativo, ma una parte integrante delle regole. La comunità può quindi continuare a produrre nuove situazioni e prolungare la vita del gioco ben oltre il contenuto ufficiale.

L'espansione può essere scaricata gratuitamente dal sito Worm.net. Il pacchetto viene distribuito in formato ZIP o LhA e deve essere scompattato nella cartella WORMSDC dell'installazione di Worms: The Director's Cut. Sono disponibili anche download separati per mappe, terreni, mountain set e sample pack vocali. Gli utenti possono quindi installare l'intero pacchetto oppure scegliere soltanto i contenuti preferiti.

È importante ricordare che l'espansione è compatibile esclusivamente con Worms: The Director's Cut per Amiga AGA. Non funziona direttamente con le versioni per PC, console o altri sistemi. È possibile utilizzare hardware originale, FPGA o emulatori compatibili, ma serve comunque la versione specifica del gioco.Questa limitazione può sembrare severa, ma è coerente con la natura del progetto. L'obiettivo non è realizzare un remake moderno, bensì ampliare una versione precisa della storia di Worms mantenendone grafica, controlli, fisica e identità.

Worms: The Director's Cut 1.5 è un omaggio intelligente alla versione Amiga e alla comunità che continua a sostenerla. Gratuita, ricca e piena di contenuti, questa espansione dimostra che, quando si parla di vermi armati, il concetto di pensionamento è evidentemente esploso insieme alla pecora… Booom!

di **Carlo Nithaiah Del Mar Pirazzini**



TETRIS BASIC AMSTRAD

Anno: 2026
Editore/Sviluppatore: Altanerus Dog
Genere: Puzzle Game
Piattaforma: Amstrad CPC
Sito web: https://altanerus-dog.itch.io/tetris-basic-amstrad

Tetris-Basic-Amstrad di altanerus dog è un piccolo progetto homebrew per Amstrad CPC. L'idea è semplice e curiosa, ricreare Tetris interamente in Locomotive BASIC 1.0, senza utilizzare routine in linguaggio macchina, proprio come se fosse uno dei programmi pubblicati negli anni Ottanta sui manuali o sulle riviste dedicate all'Amstrad CPC. Il risultato occupa appena 5 KB e, pur con evidenti limitazioni, è completamente funzionale.



Il giocatore deve completare 99 linee disponendo correttamente i tetramini, utilizzando una configurazione di comandi essenziale: O e P per muoversi lateralmente, Q per ruotare i pezzi e la barra spaziatrice per accelerarne la caduta.

Il principale elemento caratterizzante è la lentezza. L'uso di una matrice bidimensionale e la quantità di calcoli richiesti dal BASIC provocano un certo ritardo nella risposta ai comandi, soprattutto nella prima versione. Non si tratta quindi di un Tetris frenetico, bisogna giocare con calma, prevedere gli spostamenti e accettare i tempi di elaborazione della macchina.

Questo può apparire scomodo, ma rappresenta anche il senso dell'operazione: osservare direttamente i compromessi e i limiti della programmazione dell'epoca.

La versione 2 migliora proprio gli aspetti più problematici, offrendo maggiore velocità, controlli più efficaci e una gestione più precisa della caduta rapida. Nonostante la sua semplicità, il gioco conserva una certa componente ipnotica. Una volta iniziata la partita, l'obiettivo delle 99 linee invita a proseguire, anche soltanto per vedere fin dove può arrivare un







GIUDIZIO FINALE

» Giocabilità 80%
E' lento e semplice ma è sempre Tetris!

» Longevità 80%
Un curioso esperimento di programmazione che risulta piacevole da giocare ogni tanto.





programma BASIC così compatto.

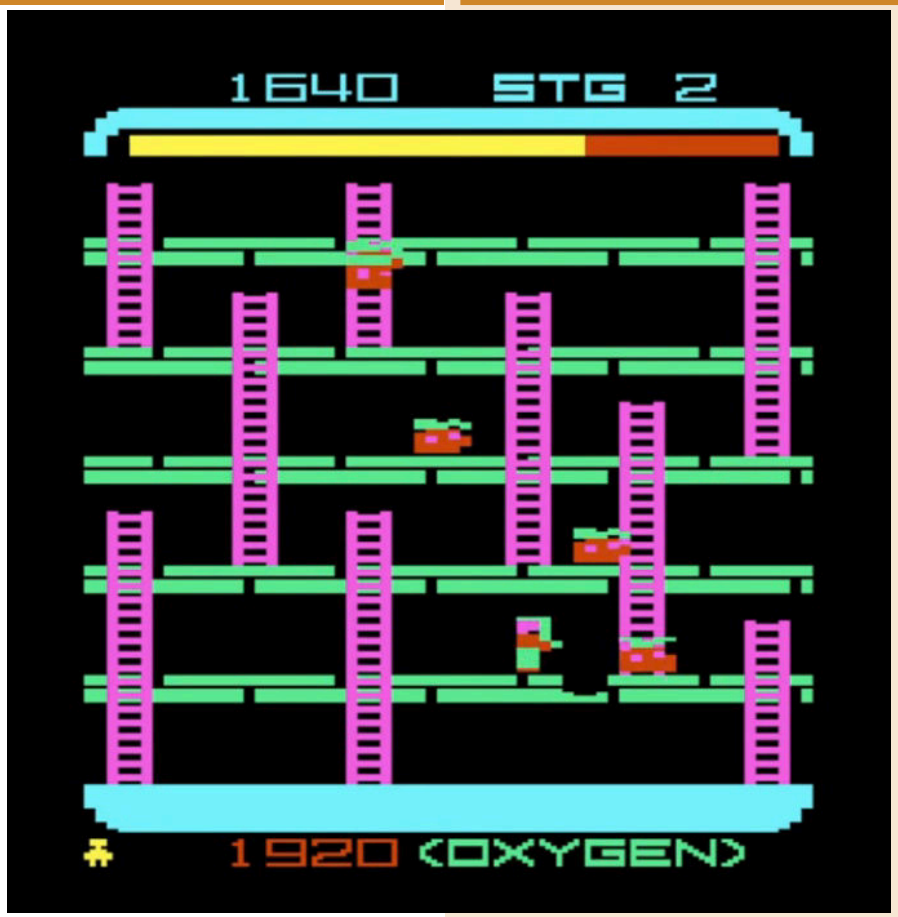
Tetris-Basic-Amstrad non è una conversione tecnicamente ambiziosa, né vuole competere con le versioni commerciali del celebre puzzle game. È piuttosto una piacevole capsula del tempo, un omaggio ai listati da copiare a mano e ai giochi distribuiti tramite riviste. Imperfetto ma sincero, offre agli appassionati di Amstrad CPC un'esperienza interessante tanto per il gioco quanto per il valore didattico e nostalgico.

di **Giampaolo Moraschi**



SPACE PANIC DX

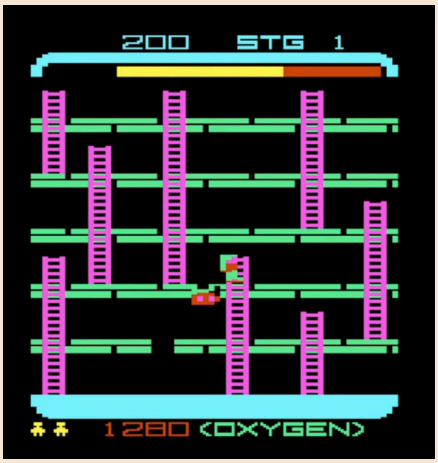
Anno: 2026
Editore/Sviluppatore: Denial
Genere: Platform
Piattaforma: Commodore VIC-20
Sito web: https://sleepingelephant.com/ipw-web/bulletin/bb/viewtopic.php?t=11628

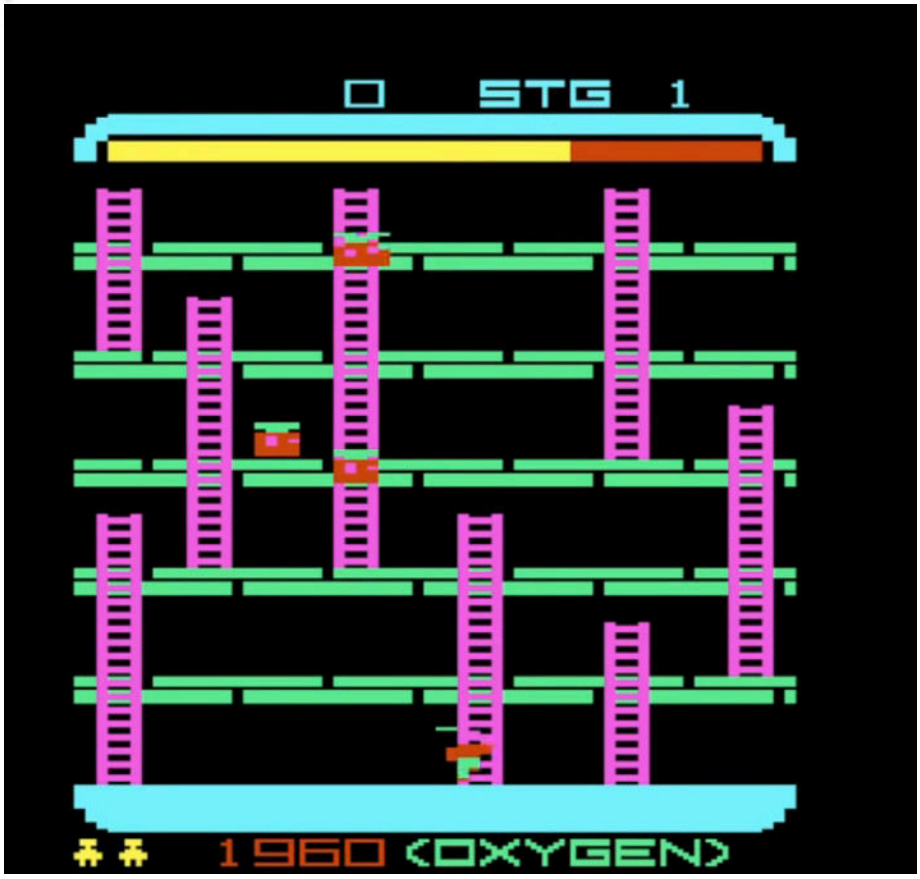


I vecchi arcade continuano a esercitare un fascino particolare perché, dietro una grafica essenziale, nascondono spesso idee di gioco ancora efficaci. Space Panic DX, realizzato da Denial per Commodore VIC-20, recupera uno dei titoli più interessanti della prima generazione dei coin-op, Space Panic, pubblicato da Universal nel 1980 e considerato uno dei precursori dei platform a schermata fissa.

Il giocatore controlla un astronauta impegnato a muoversi fra piattaforme e scale, mentre creature aliene lo inseguono. La caratteristica più originale è l'assenza del salto, per eliminare i nemici bisogna scavare buche nelle piattaforme, attirare gli alieni al loro interno e poi colpirli richiudendo la trappola con la pala. I nemici più resistenti richiedono spesso due buche sovrapposte, rendendo ogni eliminazione una piccola operazione strategica.

L'idea contribuì a definire alcune meccaniche poi sviluppate da giochi come Donkey Kong e Lode Runner. Anche l'ossigeno, che diminuisce durante la partita, impedisce di procedere con troppa cautela e mantiene costante la pressione. Questa versione per VIC-20 ripropone fedelmente la struttura, senza aggiungere elementi estranei o trasformare il gioco in un platform moderno. I comandi sono semplici: il pulsante di fuoco permette di scavare, mentre fuoco più direzione verso l'alto a riempire la buca e colpire il nemico intrappolato. La semplicità dei controlli non significa però semplicità. Per scavare bisogna trovarsi nella posizione corretta, mentre per eliminare un alieno occorrono tempismo e una buona conoscenza dei suoi movimenti. Una buca aperta troppo presto può rivelarsi inutile; una preparata nel punto giusto può invece consentire di neutralizzare più avversari con una sola azione. Dal punto di vista tecnico, il progetto sfrutta con intelligenza le risorse del VIC-20. La grafica è necessariamente spartana, ma piattaforme, scale, astronauta e alieni risultano facilmente riconoscibili. In un gioco basato soprattutto sulla lettura dello schermo, questa chiarezza è più importante del dettaglio grafico. La versione è indicata come

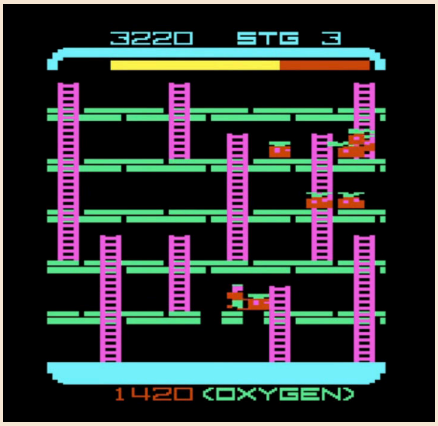




GIUDIZIO FINALE

» Giocabilità 70%
Risente un po' nella struttura arcaica del gioco e richiede impegno per essere padroneggiato.

» Longevità 70%
Difficile e non adatto ai casual gamers. Decisamente vecchio stile per curva di difficoltà.

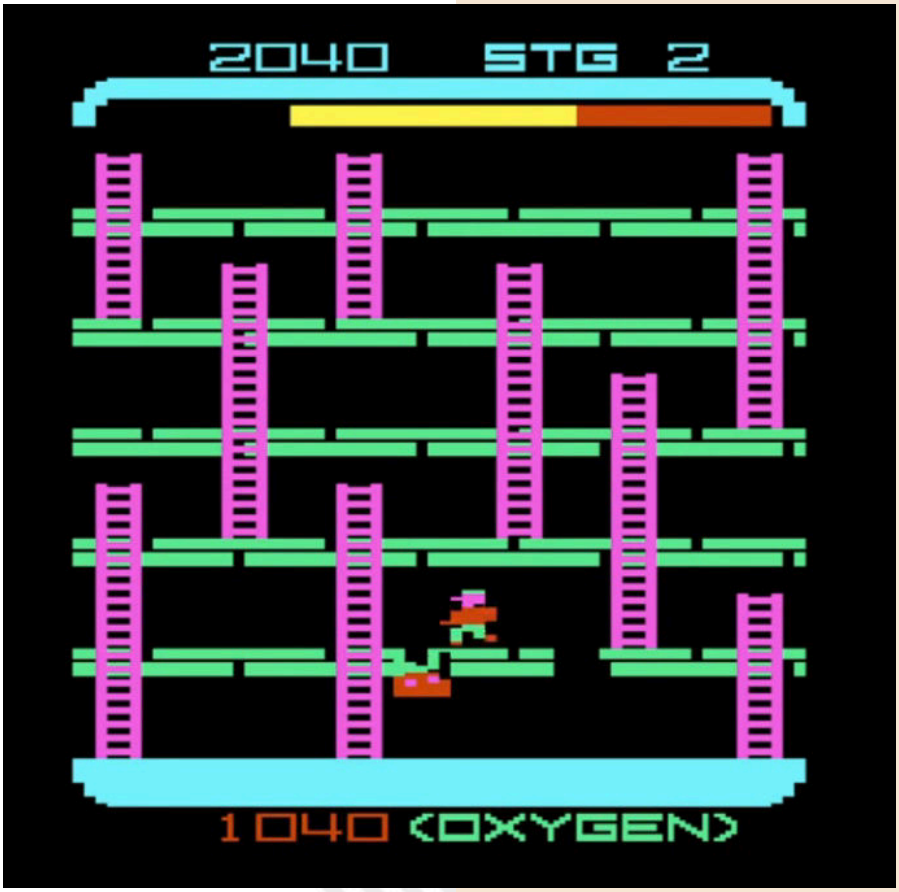


conversione PAL e richiede 35K di memoria.

Il fascino di Space Panic DX nasce dal suo ritmo particolare. Non è un gioco veloce nel senso moderno, ma riesce a creare tensione perché ogni operazione richiede tempo. Questa impostazione genera un'esperienza fatta di brevi tentativi e ripetuti miglioramenti. Le prime partite servono a capire le distanze, il comportamento dei nemici e il momento giusto per agire. In seguito il giocatore comincia a pianificare le trappole, cercando di sfruttare le cadute e le eliminazioni multiple per ottenere punteggi migliori.La difficoltà può apparire severa, soprattutto a chi è abituato ai platform contemporanei. Tuttavia, è una difficoltà coerente con la filosofia arcade originale. Di sicuro il gioco non accompagna il giocatore, ma gli chiede di imparare attraverso l'esperienza.

Non è un gioco destinato a chi cerca grafica spettacolare o una progressione moderna, ma rappresenta un'ottima proposta per gli appassionati di retrogaming e homebrew. Denial dimostra che anche un'idea del 1980 può funzionare su un computer domestico dalle risorse limitate, quando viene adattata con attenzione e rispetto.

di **Giampaolo Moraschi**



SUPER MARIO BROS

Anno: 2026
Editore/Sviluppatore: LackofTrack
Genere: Platform
Piattaforma: Sega Master System
Sito web: https://www.smspower.org/Homebrew/SuperMarioBrosSmsPort-SMS



Qualcuno avverta Alex Kidd: c'è un idraulico italiano abusivo nel Master System. E non è venuto ad aggiustare i tubi.

Quarantuno anni dopo l'originale, Super Mario Bros. compie infatti uno dei sacrilegi più affascinanti della storia degli 8 bit... gira sulla console dell'acerrima rivale SEGA. Il responsabile è LackofTrack, che aveva presentato il progetto alla SMS Power! Coding Competition 2026 come semplice proof-of-concept limitato al World 1-1. Evidentemente si è fatto prendere la mano, perché il 31 agosto è arrivata la versione 1.00 con il gioco completo.

E attenzione... non stiamo parlando di un platform travestito da Mario con due baffi appiccicati sopra. Il progetto utilizza come riferimento le disassembly del codice originale NES, con l'obiettivo di riprodurne il comportamento il più fedelmente possibile. Risultato? Mario corre, salta, prende funghi, schiaccia Goomba, entra nei tubi e continua a scoprire che la principessa è nell'ennesimo castello sbagliato. Insomma, tutto regolarmente frustrante come nel 1985... con tanto di warp zone al seguito!

Ma LackofTrack non si è accontentato. Oltre alla modalità grafica fedele all'originale troviamo una versione Enhanced, ispirata a Super Mario All-Stars, che sfrutta la palette del Master System per offrire fondali e sprite molto più ricchi. Il risultato è curioso. Un gioco NES che gira su SEGA cercando di sembrare la versione Super Nintendo. A questo punto manca soltanto Sonic che faccia causa per occupazione abusiva di console. Anche il sonoro beneficia del trattamento speciale grazie al







GIUDIZIO FINALE

» Giocabilità 95%
Super Mario Bros su SMS con due versioni in una. Mamma mia se si incacchierà la Nintendo!?!!?!

» Longevità 90%
C'è tutto il classico più una versione "enhanced" come quella vista di Super Mario All Stars.

supporto FM YM2413, con arrangiamenti ispirati alla versione All-Stars. Rimangono qualche rallentamento quando lo schermo si affolla e un po' di flickering, ma sono peccati veniali che quasi contribuiscono all'autenticità 8-bit.

Il risultato è tecnicamente impressionante e soprattutto tremendamente divertente da vedere in azione. Super Mario Bros. SMS Port non riscrive la storia della guerra SEGA-Nintendo: semplicemente aspetta che siano passati quarant'anni, entra dalla porta sul retro e mette Mario sul Master System.

SEGA does what Nintendon't? Stavolta, tecnicamente… fa proprio Nintendo.

di **Roberto Del Mar Pirazzini**







BUBBLE FIGHT

Anno: 2026
Editore/Sviluppatore: SiRioKD
Genere: Arcade
Piattaforma: Sega Master System, Sega SG-1000 e SC-3000
Sito web: https://siriokd.itch.io/bubble-fight



Bubble Fight nasce da un'idea tanto semplice quanto irresistibile... portare sul SEGA SG-1000 lo spirito di Balloon Fight, uno degli arcade più immediati e divertenti degli anni Ottanta. SiRioKD non nasconde affatto la fonte d'ispirazione e realizza una reinterpretazione fedele, racchiusa in appena 32 KB e destinata anche allo SC-3000.

Il principio è quello che conosciamo. Il nostro protagonista vola sostenuto dai palloncini e dobbiamo sfruttare continue spinte verso l'alto, inerzia e cambi di direzione per conquistare una posizione favorevole sugli avversari. Colpirli dall'alto significa far scoppiare i loro palloncini; finire sotto di loro significa invece prepararsi a perdere.

Ed è proprio nella particolare fisica del volo che Bubble Fight trova la sua forza. I comandi si comprendono immediatamente, ma controllare davvero traiettorie e velocità richiede pratica. All'inizio si svolazza premendo furiosamente il pulsante, poi si comincia a sfruttare cadute, accelerazioni e improvvisi cambi di direzione. Quella che sembrava una semplice baraonda di sprite diventa così una piccola battaglia di posizione.

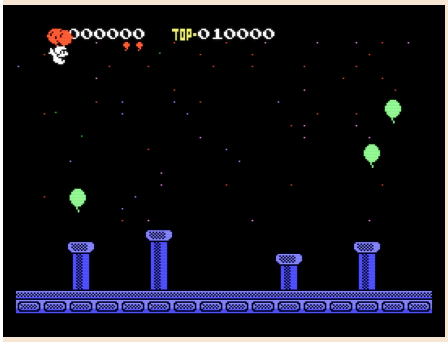
Tecnicamente il gioco mantiene volutamente un aspetto essenziale, perfettamente coerente con l'hardware SEGA dei primi anni Ottanta. Sprite, scenari e colori non cercano effetti spettacolari, privilegiando invece pulizia e leggibilità dell'azione. Tutto deve essere compreso in una frazione











di secondo e Bubble Fight riesce bene nell'intento.

Naturalmente la struttura è semplice e inevitabilmente ripetitiva, ma fa parte del suo DNA arcade. Non servono tutorial, progressioni complicate o mille modalità: si prende il controller e si gioca. Bubble Fight è Balloon Fight! Non fa nulla di nuovo in più. È piuttosto una curiosa operazione di archeologia videoludica: come sarebbe stato questo classico se fosse arrivato sul SG-1000 quarant'anni fa? Probabilmente proprio così. E dopo l'ennesimo palloncino scoppiato, premere nuovamente Start viene ancora terribilmente naturale.



di **Marta Rossmann**

GIUDIZIO FINALE

» Giocabilità 80%
Semplice e divertente. Si adatta perfettamente anche su SC3000

» Longevità 80%
Dopo le prime partite ci si prendere la mano e il divertimento c'è tutto.

LEMMINGS RELOADED

Anno: 2026
Editore/Sviluppatore: LunaticLab
Genere: Puzzle Game
Piattaforma: Commodore 64, Commodore 64 Ultimate
Sito web: https://lunaticlab.itch.io/lemmings-reloaded



Nel panorama dell'homebrew per Commodore 64, Lemmings Reloaded è uno di quei progetti super attesi. Dopo diversi mesi di sviluppo, numerosi trailer e notizie, ecco che LunaticLab pubblica il pacchetto il 15 agosto 2026, un tributo non commerciale alla serie ideata da DMA Design e, in particolare, alla versione C64 del 1994. Ma non si limita a riproporre quel vecchio adattamento, ne amplia l'esperienza con nuovi livelli, una presentazione più ricca e una serie di funzioni che lo trasformano in una sorta di edizione definitiva per la macchina Commodore.

Il principio di gioco è quello ben noto. I lemming sono sempre quelli che abbiamo amato ed odiato in tutti questi anni, escono da un ingresso e avanzano senza fermarsi, mentre il giocatore deve assegnare loro abilità specifiche (come scavare, costruire, bloccare o arrampicarsi) per raggiungere l'uscita evitando trappole e precipizi. Non si controlla direttamente ogni creatura, ma si dirige l'intera marcia attraverso interventi rapidi e ragionati. È una formula che unisce puzzle game e strategia in tempo reale, e che conserva intatta la propria efficacia anche a distanza di decenni.

La vera sorpresa è la quantità di contenuti. Lemmings Reloaded propone 123 livelli, tre in più rispetto alla versione Amiga originale, combinando scenari classici, conversioni provenienti da altri sistemi e stage completamente nuovi. La progressione mantiene l'articolazione tradizionale in difficoltà crescenti, ma introduce anche un sistema opzionale di valutazione a stelle. Completare un livello non è più l'unico obiettivo, perché diventa importante salvare il maggior numero possibile di lemming e utilizzare le abilità con precisione. Questa aggiunta è particolarmente azzeccata. I livelli più semplici acquistano una seconda vita, mentre quelli già conosciuti possono essere rigiocati per migliorare sempre di più e aumentare il proprio valore. Il sistema







GIUDIZIO FINALE

» Giocabilità 95%

Una meraviglia con 123 livelli di gioco, nuovi livelli, sistema di punteggio che spinge a migliorarsi sempre di più e una colonna sonora ben fatta.

» Longevità 90%

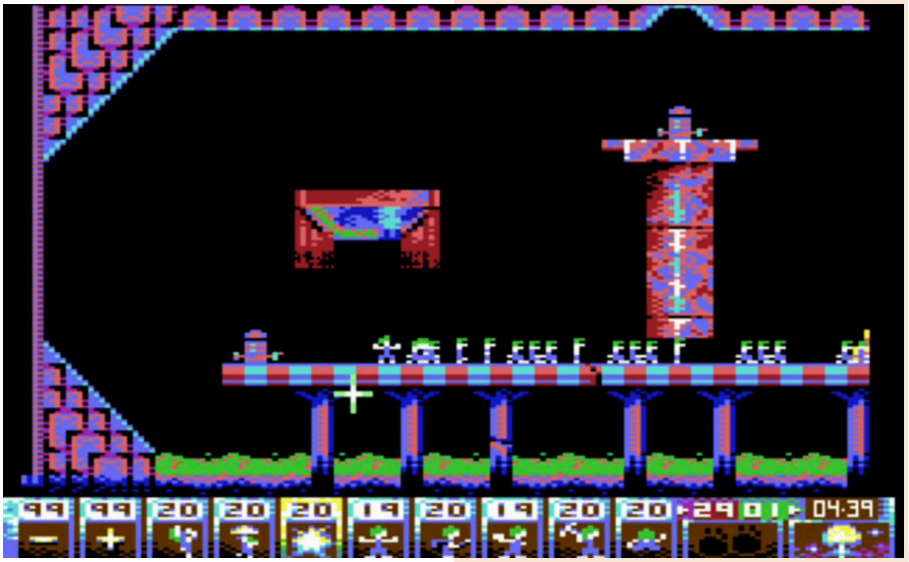
Il mouse è praticamente indispensabile per godersi appieno l'esperienza, mentre la compatibilità con alcuni dispositivi può richiedere attenzione. Una volta superati questi aspetti, però, emergono una conversione ricchissima, un sistema di gioco ancora irresistibile e una realizzazione tecnica davvero notevole.

di valutazione non altera la struttura originale, ma offre una motivazione concreta per tornare su scenari apparentemente risolti.

Dal punto di vista tecnico, il progetto è notevole. Gli sviluppatori hanno scelto una distribuzione esclusivamente su cartuccia, nei formati EasyFlash e GMod2, con salvataggio dei progressi direttamente nella memoria del supporto. Sono inoltre previsti il supporto per Commodore 64 reale, C64 Ultimate, VICE e altri ambienti compatibili. Il gioco richiede però una macchina PAL e il mouse è fortemente consigliato, perché rappresenta il sistema di controllo più naturale per selezionare rapidamente i lemming.

La grafica cerca di avvicinarsi alla versione Amiga più di quanto facesse il vecchio porting ufficiale. Gli sfondi sono più dettagliati, gli elementi del livello risultano leggibili e i piccoli protagonisti sono gestiti tramite sprite, con un lavoro di multiplexing che spinge il C64 vicino ai suoi limiti. Nei momenti con molti lemming sullo schermo possono comparire rallentamenti, flickering i e una risposta meno fluida, ma si tratta di compromessi comprensibili considerando la quantità di elementi animati.

Uno degli aspetti più discussi riguarda la colonna sonora. Il gioco include trenta tracce SID composte da Nordischsound, per oltre un'ora di musica, sbloccabili durante la progressione e ascoltabili anche attraverso un jukebox integrato. Le musiche non riproducono semplicemente il materiale originale, scelta che potrebbe deludere chi cerca una conversione filologica dell'edizione Amiga. D'altra parte, la varietà della

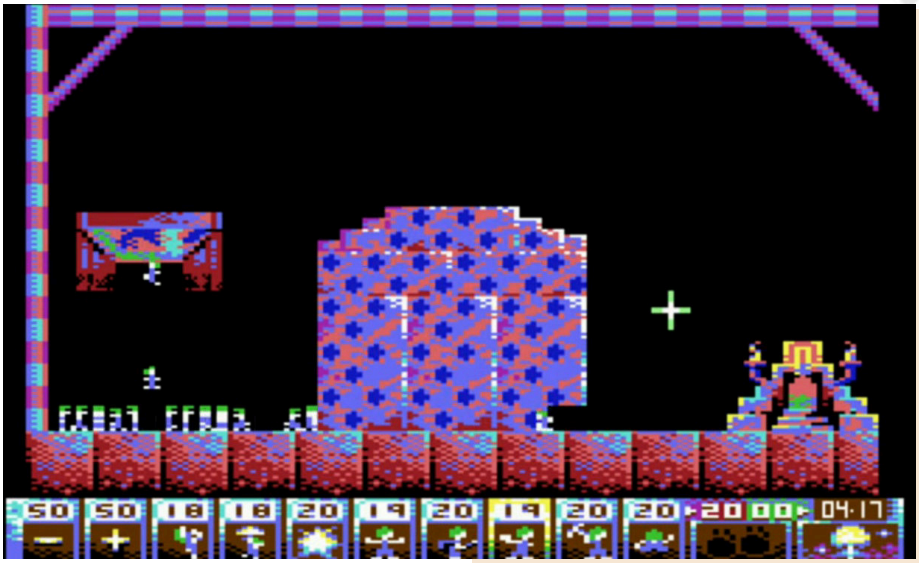




nuova colonna sonora contribuisce a dare personalità al progetto e sfrutta bene le caratteristiche del SID. Alcuni brani risultano lontani dall'atmosfera tradizionale di Lemmings, ma nel complesso accompagnano efficacemente l'azione e aggiungono valore alla modalità jukebox.

Il principale limite di Lemmings Reloaded non è il gioco in sé, ma la piattaforma su cui viene eseguito. Il C64 deve gestire molti sprite, collisioni, animazioni e operazioni di selezione contemporaneamente, e in alcune situazioni la fluidità non è paragonabile a quella delle versioni Amiga o PC. Anche il mouse può comportarsi in modo non perfettamente uniforme su determinati adattatori o sistemi moderni.

Sono tuttavia difetti secondari rispetto all'ambizione del progetto. Il risultato supera nettamente il vecchio porting C64 e dimostra quanto sia ancora possibile ottenere da una macchina del 1982 attraverso una programmazione accurata e una gestione intelligente della memoria. La presenza di 123 livelli, del salvataggio, della selezione degli stage, dellepasswordclassicheedelle



valutazioni a stelle rende l'esperienza completa e duratura.

Signori miei, questo Lemmings Reloaded è una delle produzioni homebrew più impressionanti apparse recentemente su Commodore 64. Non è soltanto un omaggio nostalgico, ma una rielaborazione concreta e generosa, capace di offrire più contenuti della versione Amiga originale e di correggere molte delle debolezze del porting ufficiale.



di **Carlo Nithaiah Del Mar Pirazzini**



ROGÜL

Anno: 2026
Editore/Sviluppatore: Retro Blitter Team
Genere: Roguelike/GDR
Piattaforma: Atari ST/Ste
Sito web: https://www.atarimania.com/games/atari-st-games-rogul-45634



Nelle profondità di Amalotolos, dieci dimensioni marcite dall'oscurità attendono il guerriero abbastanza folle da attraversarle. Tra rovine dimenticate, corridoi infestati e creature nate dall'incubo, ogni passo può essere l'ultimo: i mostri osservano, le ombre si stringono attorno alla preda e persino la terra sembra pronta a tradire chi la calpesta. Qui non esistono eroi invulnerabili, né seconde possibilità. La morte è definitiva, il bottino è incerto e ogni battaglia può trasformare una vittoria in una condanna. Solo chi saprà dominare la paura, sfruttare ogni angolo del labirinto e accettare il prezzo della sconfitta potrà raggiungere il tiranno delle dimensioni demoniache..

Rogül nasce originariamente su Atari 8-bit per opera di Wojtek Bociański e arriva successivamente su Atari ST e STE grazie al Retro Blitter Team. Il progetto non è una semplice conversione realizzata in fretta, ma una ricostruzione sviluppata con un nuovo motore e una presentazione pensata per sfruttare le caratteristiche delle macchine Atari a 16 bit. La storia è essenziale, ma offre una motivazione chiara. Non ci sono lunghe sequenze narrative o dialoghi elaborati. Rogül preferisce raccontare la propria storia attraverso l'ambientazione tra rovine, passaggi stretti, creature ostili e stanze che sembrano progettate per mettere alla prova la resistenza psicologica del giocatore.

La struttura è quella tipica del roguelike. I mondi vengono generati proceduralmente, quindi ogni partita presenta mappe, nemici e oggetti disposti in modo differente. Non è possibile affidarsi completamente alla memoria: una strada sicura in una partita può trasformarsi in una trappola in quella successiva. L'obiettivo di ogni dimensione è esplorare, combattere, raccogliere esperienza e trovare il modo di proseguire. La difficoltà aumenta gradualmente e i nemici diventano sempre più aggressivi e resistenti. Il gioco non accompagna il giocatore con tutorial invadenti. Dopo pochi istanti, bisogna imparare







GIUDIZIO FINALE

» Giocabilità 94%

Un mix tra gdr classico e roguelike dove ogni mossa conta e il gameplay è linea super funzionale. Audio e grafica curati e adattati al meglio.

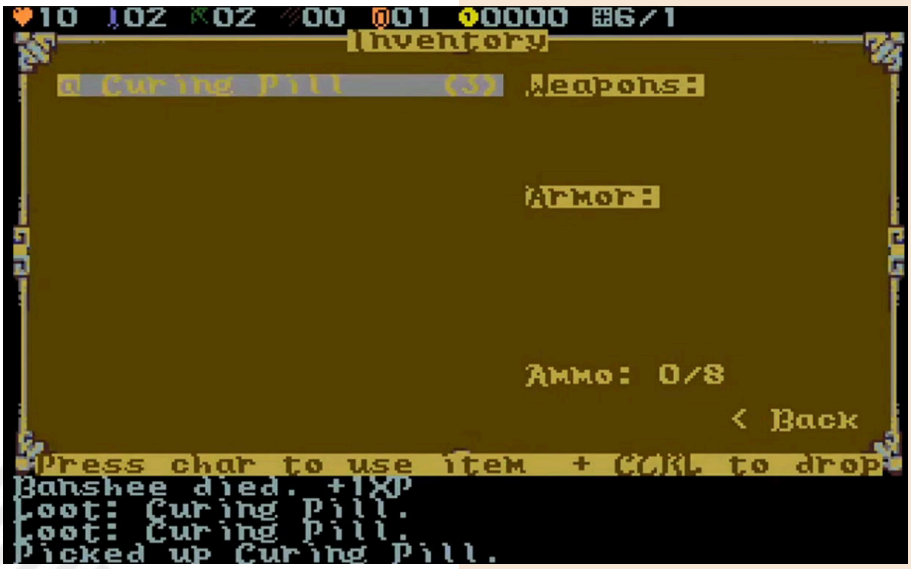
» Longevità 90%

E' spietato! Mettetevelo bene in testa… Ma è gratificante da esplorare e giocare e vi incollerà allo schermo.

direttamente sul campo, osservare i movimenti dei nemici, capire quali oggetti utilizzare, valutare le distanze e scegliere quando combattere e quando invece evitare lo scontro. Una scelta può sembrare severa, ma si adatta perfettamente al genere. Rogül vuole che ogni scoperta sia conquistata, non semplicemente consegnata attraverso una finestra di istruzioni.

Gli scontri si svolgono a turni e la posizione sul terreno ha un'importanza decisiva. Ogni movimento può influenzare il combattimento successivo, perché muri, corridoi e ostacoli modificano le possibilità del protagonista e dei nemici. Non basta quindi premere il pulsante d'attacco. Occorre osservare l'ambiente, capire quali avversari si trovano nelle vicinanze e decidere se avanzare, arretrare o sfruttare un punto favorevole.Le armi disponibili comprendono lame, asce, spade e altri strumenti per gli attacchi ravvicinati. Sono presenti anche archi e balestre, ma le armi a distanza devono essere trovate durante l'esplorazione e non sono sempre disponibili. Questa limitazione rende ogni ritrovamento importante. Il sistema è semplice, ma non banale. La profondità nasce dall'interazione tra terreno, posizionamento, equipaggiamento e caratteristiche degli avversari. I nemici sconfitti forniscono esperienza, permettendo al protagonista di salire di livello e migliorare le proprie possibilità di sopravvivenza. Il problema è che la morte è permanente… quando il guerriero cade, la partita ricomincia dall'inizio. È una scelta crudele, ma indispensabile per mantenere la tensione. Ogni combattimento può diventare importante e ogni errore lascia una lezione. Una partita fallita non è necessariamente tempo perso: può insegnare a riconoscere un nemico, scegliere meglio l'equipaggiamento o evitare un combattimento inutile. Naturalmente, dopo essere morti per la quinta volta a causa dello stesso avversario, la filosofia lascia spazio a un commento poco elegante rivolto allo schermo (fidatevi ne sono volati parecchi in redazione di commenti poco eleganti). È parte del rito roguelike. La generazione procedurale rafforza questo sistema. Non potendo memorizzare completamente la mappa, il giocatore deve sviluppare una vera capacità di adattamento. Ogni partita diventa un nuovo tentativo di attraversare l'oscurità, portando con sé soltanto l'esperienza accumulata. Durante l'esplorazione è possibile raccogliere pozioni, pergamene, armi e altri oggetti utili. Le pergamene possono offrire effetti differenti, tra cui il teletrasporto, mentre le pozioni possono recuperare salute o aiutare a superare una situazione pericolosa.

Sono presenti anche altari misteriosi, capaci di offrire ricompense o conseguenze imprevedibili. Avvicinarsi significa spesso scegliere se rischiare oppure proseguire senza sapere cosa ci si è lasciati alle spalle. Gli altari rappresentano bene la filosofia di Rogül: il gioco premia la curiosità, ma non





garantisce che la curiosità venga ricompensata. A volte il misterioso dono degli dèi è proprio ciò che mancava per completare la giornata… un nuovo problema. L'oro raccolto può essere utilizzato presso i negozi, dove acquistare oggetti e prepararsi per le dimensioni successive. La gestione dell'inventario diventa così fondamentale.

Questa versione Atari ST/STE mostra una notevole attenzione all'atmosfera. Gli ambienti sono oscuri, le creature sono minacciose e le dieci dimensioni hanno una personalità visiva riconoscibile. Il lavoro grafico del Retro Blitter Team non si limita ad aumentare la risoluzione dell'originale. La presentazione è stata ricostruita per adattarsi all'hardware Atari a 16 bit, con una direzione artistica che valorizza rovine, corridoi e stanze infestate. La visuale dall'alto è funzionale alla struttura del gioco. Permette di leggere il campo di battaglia, osservare i nemici e valutare le possibili vie di fuga. L'ambiente resta sempre parte del combattimento e non un semplice sfondo. Non servono effetti moderni per creare tensione. Una stanza stretta, due nemici ai lati e una barra della salute quasi vuota sono sufficienti per trasformare il salotto in una cripta. Il nuovo motore è stato sviluppato specificamente per Atari ST e STE. La versione STE può sfruttare l'audio DMA, mentre l'Atari ST utilizza il chip Yamaha 2149F per la musica compatibile con l'hardware originale.La musica Yamaha è attribuita ad Adrien "AD" Dangha, mentre grafica, effetti sonori e musica WAV sono curati da Krzysztof "Hospes" Jedowski. Questa attenzione alle differenti configurazioni dimostra la cura tecnica del progetto. L'obiettivo non è semplicemente far partire il gioco, ma creare un'esperienza adatta alle diverse caratteristiche delle macchine. La versione STE sembra offrire il comparto sonoro più ricco, ma anche l'Atari ST mantiene una resa coerente e riconoscibile. In entrambi i casi, il risultato contribuisce a dare un'identità precisa all'avventura.

Alla fine della fiera… Rogul è una delle proposte più interessanti per Atari ST e STE. Parte da una formula classica, ma la sviluppa con sufficiente personalità da non sembrare un semplice esercizio nostalgico. E' Un roguelike oscuro, impegnativo e sorprendentemente moderno nella struttura. Rogül è un must che dimostra come l'Atari ST non ha ancora esaurito le proprie possibilità. Tra i suoi "circuiti" viscere pulsano ancora mostri, labirinti e abbastanza oscurità da far rimpiangere di non aver scelto un gioco di carte. Il gioco del mese!

di **Carlo Nithaiah Del Mar Pirazzini**



FINAL WHISTLE

Anno: 2026
Editore/Sviluppatore: Magic Studio EU
Genere: Sportivo
Piattaforma: Xbox, PC Windows
Sito web: https://magicstudioeu.itch.io/final-whistle

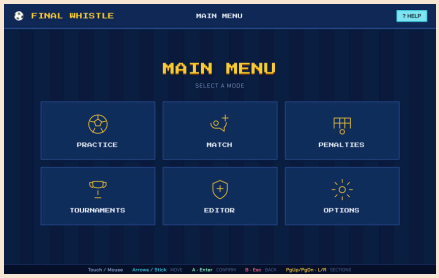


Il gioco più amato dagli italiani! Il calcio! Certo negli ultimi anni non siamo più presenti ai mondiali, ma questo non porta via nulla al nostro amore per questo sport.

Final Whistle di Magic Studio è un tributo al calcio e a quel meraviglio periodo dei giochi di calcio top down degli anni '90. Mi riferisco agli anni gloriosi di Kick Off e Sensible Soccer... campi inquadrati dall'altro, palla dinamica che si staccava dai piedi e tiri ad effetto incredibili. Bene se siete cresciuti con Kick Off, Sensible Soccer o Micropros Soccer, in questo titolo ritroverete subito quell'atmosfera e la sentirete inserita con rispetto e senza orpelli moderni inutili.

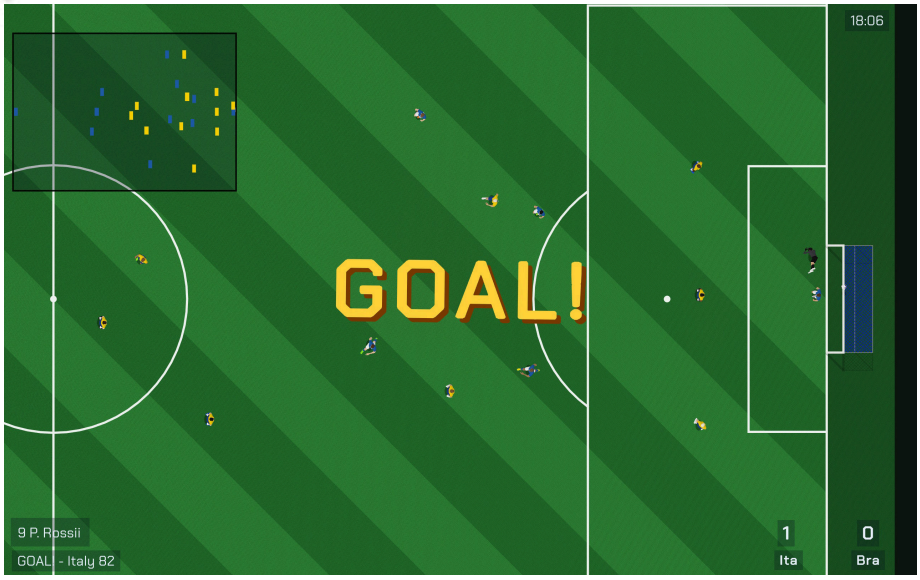
Il cuore del gioco è la fisica del pallone e il rapporto tra giocatore e palla. A differenza di tanti arcade moderni, il pallone non resta "incollato" ai piedi. Ogni tocco ha inerzia, ogni contrasto può cambiare traiettoria e ogni tiro richiedere lettura del gioco, della porta, angolo e forza. La possibilità di "piegare" il tiro dopo aver colpito la palla è un dettaglio che fa ancora la differenza e permette di correggere in corsa la partita, come nei migliori titoli della tradizione Amiga e PC DOS.

Questo approccio sposta sicuramente l'attenzione dalla semplice velocità di esecuzione alla lettura della situazione. Bisogna sapere quando passare, quando cercare il cross o quando tirare di prima. Il risultato è un calcio che premia la precisione e la pianificazione più del puro riflesso, pur mantenendo un ritmo elevato e competitivo.



Il gioco nasce per Windows e Xbox, con un'impostazione che privilegia il gioco locale e le partite rapide. Non è presente nessuna carriera





GIUDIZIO FINALE

» Giocabilità 85%
Fisica del pallone coerente con la tradizione top down, tiri correggibili dopo il colpo e un'impostazione semplice ma pulita e senza fronzoli.

» Longevità 78%
Difficile ma gratificante perché richiede pratica per essere "maneggiato". Carriera manageriale assente se cercate qualcosa di più "longevo".

manageriale complessa. L'obiettivo e di ricreare le sensazioni delle partite vere e non costruire un simulatore di club.

L'estetica è funzionale al gameplay con visuale d'alto classica, sprite leggibili e un campo sempre ben definito. Nessuna pixel art nostalgica ne iperrealismo, ma una grafica che serve alla giocabilità. Stesso discorso per l'audio che segue la stessa logica con effetti chiari, niente distrazioni e tutto finalizzato a far capire cosa sta succedendo in campo.

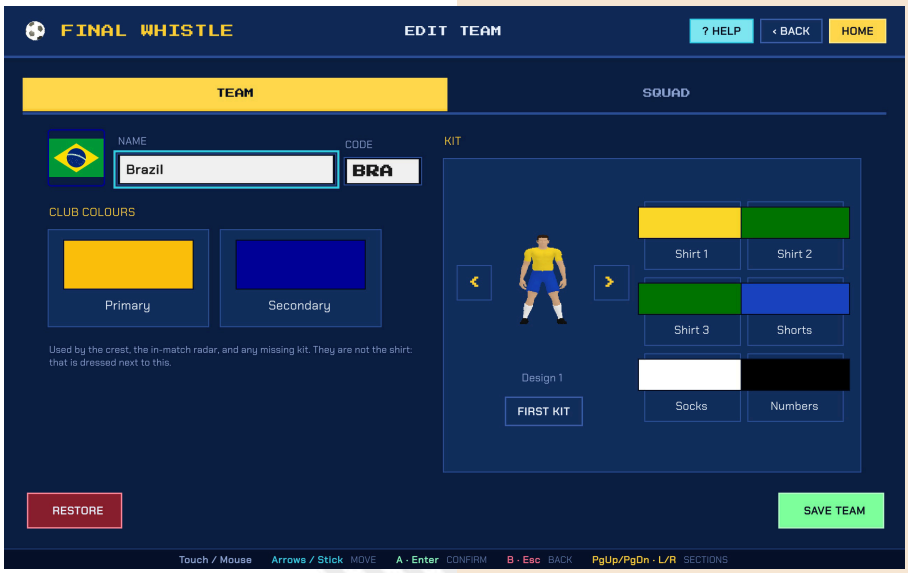
E' un titolo nato appositamente per chi viene dai classici già citati. Si adatta a sessioni brevi, ha una bella serie di campionati internazionali ed è super settabile in ogni menù. Chi preferisce i manageriali sportivi invece deve guardare oltre. Se si cercano tattiche, trasferimenti o altro questo non è il titolo che fa per voi.

Concludendo possiamo dire che Final Whistle è un titolo di nicchia ma coerente. Prende l'idea di calcio "vista dall'alto" e la applica con rispetto puntando sul controllo, la fisica del pallone e la leggibilità. Non è Fifa e non è un manager... ma, grazie al cielo, è un titolo per chi vuole sentire che ogni tocco di palla conta e che segnare richiede una lettura e non solo riflessi.

di **Giampaolo Moraschi**







PITFALL II

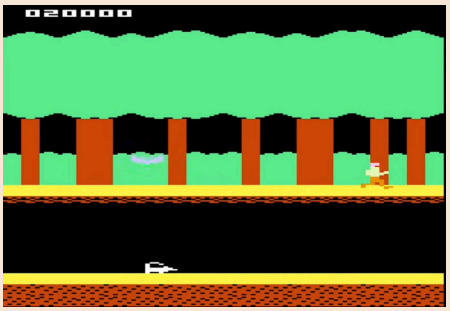
Anno: 2026
Editore/Sviluppatore: Gekka
Genere: Platform
Piattaforma: Commodore VIC 20
Sito web: https://www.sleepingelephant.com/ipw-web/bulletin/bb/

Ah, Pitfall II... un must su Atari 2600... appunto! Prendete l'originale per Atari 2600 (un capolavoro assoluto, con tanto di chip audio integrato nella cartuccia) e provate a stiparlo dentro un VIC-20 espanso. Il risultato è la dimostrazione vivente che «se vuoi, puoi... ma magari anche no».



Diciamocelo... Pitfall Harry qui sembra un simpatico omino stilizzato fatto con i fiammiferi, ma, ehi, la giungla c'è tutta! I serpenti, gli scorpioni e i pipistrelli sono un po' «sgranati», ma hanno il loro fascino squisitamente retrò. Bisogna socchiudere un po' gli occhi e metterci la fantasia, esattamente come facevamo da ragazzini davanti alla TV a tubo catodico. E sapete cosa? Funziona!

Il gioco mantiene l'innovativa struttura dell'originale: niente Game Over e vite infinite! Bellissimo, no? Beh, c'è un piccolo «dettaglio». Ogni volta che sfiorate un pipistrello o vi fate la barba con la coda di uno scorpione, la musica si trasforma in un pianto funebre e Pitfall Harry viene risucchiato

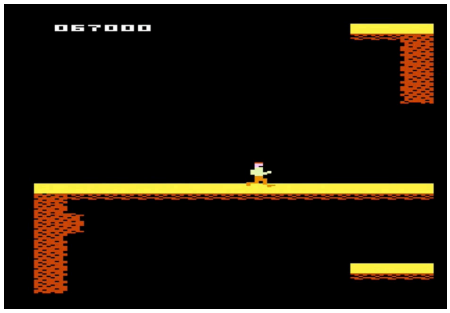


Anche la leggendaria marcetta di Pitfall II c'è! Ovviamente il chip sonoro del VIC-20 non può competere con quello utilizzato dalla versione Atari 2600, quindi la musica sembra uscire da un carillon elettronico leggermente stonato. Però conserva quel ritmo allegro che vi farà canticchiare comunque mentre cercate disperatamente di non cadere nei burroni.

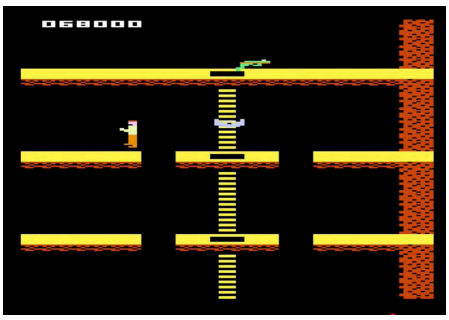




all'indietro fino all'ultimo checkpoint.



In pratica, progredire nel gioco è un po' come provare a salire una scala mobile al contrario indossando i pattini da ghiaccio. I salti richiedono una precisione da neurochirurgo e, quando, dopo mezz'ora di sofferenze, vi ritroverete nuovamente al punto di partenza per colpa di un coccodrillo distratto, il desiderio di lanciare il joystick fuori dalla finestra sarà fortissimo. Soprattutto per me, che non sono famosa per la mia pazienza!



Ma, ehi, quando finalmente riuscirete a superare quel maledetto passaggio, la soddisfazione sarà talmente grande che vi sentirete pronti per una laurea ad honorem in ingegneria dei videogiochi!



Insomma, Pitfall II su VIC-20 è un piccolo miracolo di programmazione, capace di farvi perdere la pazienza più e più volte, ma al quale è davvero impossibile voler male!

di **Chiara "Kika" Carrara**

GIUDIZIO FINALE

» Giocabilità 70%

La risposta dei comandi richiede una devozione quasi religiosa e una pazienza da monaco buddista. L'assenza del Game Over è fantastica, ma il continuo «fai un passo avanti e tre indietro» fa sembrare Harry un pendolare perennemente in ritardo. Cadi, vieni risucchiato al checkpoint e il gioco sembra dirti: «Dai, ritenta e sii più saggio». Punitivo, certo, ma ci si casca ogni volta.

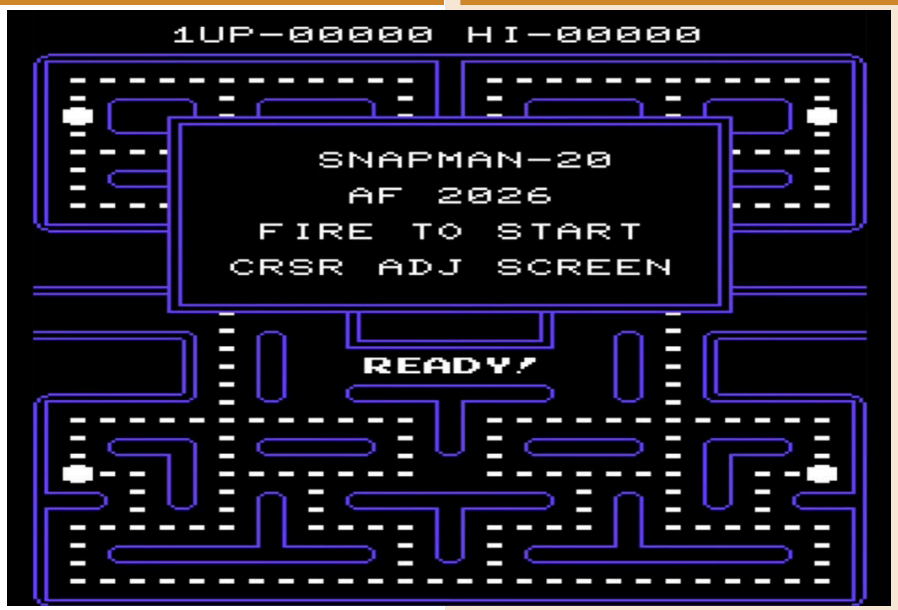
» Longevità 80%

La mappa è gigantesca, ma è soprattutto il continuo «avanti e indietro» a farla sembrare infinita. Tra un «Ok, l'ultimo tentativo e spengo» e un «Stasera lo finisco», vi ritroverete a invecchiare davanti al monitor. Un ottimo investimento: comprate un gioco oggi e vi durerà fino alla pensione.

SNAPMAN-20

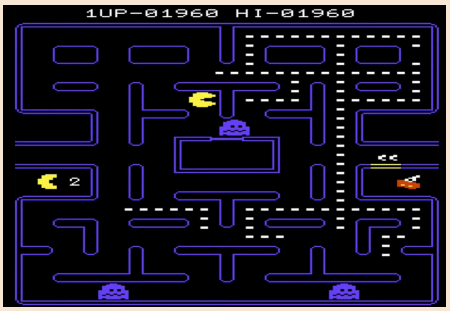
Anno: 2026
Editore/Sviluppatore: Beamrider
Genere: Maze game
Piattaforma: Commodore VIC 20
Sito web: https://sleepingelephant.com/ipw-web/bulletin/bb/viewtopic.php?p=127447#p127447

Correva l'anno 1983 e la Pac-Man Mania mieteva vittime in ogni sala giochi del pianeta. Come portarsi quell'ossessione a casa senza spendere un patrimonio? La risposta fu Snapman, la versione «fatta in casa» pubblicata dal Gruppo Editoriale Jackson per VIC-20. Un clone sfacciato, certo, ma nato in un'epoca fantastica in cui bastava cambiare un paio di lettere al titolo e ridisegnare qualche pixel per regalare un sogno a migliaia di ragazzini.



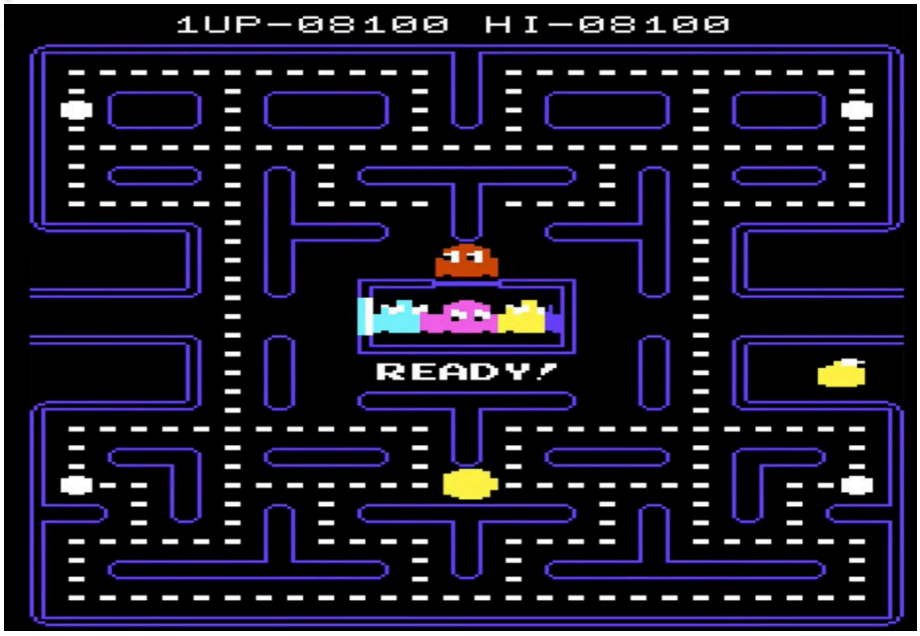
Sul VIC-20 le pretese dovevano necessariamente essere ridimensionate e Snapman spoglia l'arcade originale di tutto il superfluo per adattarlo al piccolo microcomputer Commodore. Al posto della classica pallina gialla troviamo un simpatico sprite composto da pixel squadrati che, tutto sommato, non è poi così male. E i quattro mitici fantasmini? Rimpiazzati da simboletti colorati che vagano per il labirinto seguendo un ritmo tutto loro. Bisogna metterci un pizzico di fantasia, ma, ehi... nel 1983 ci sembrava davvero di avere la sala giochi direttamente in camera!

E poi c'è il sonoro, un piccolo gioiellino di fedeltà retrò! Il classico «waka-waka» da sala giochi viene ricreato con sorprendente efficacia per le capacità sonore del VIC-20, con ogni pallino inghiottito accompagnato da un «pop» ritmato e soddisfacente. Quando poi ingurgitate una super-pillola e parte l'inseguimento, il cambio di tono e l'immancabile segnale sonoro della cattura, quando venite presi, vi riportano dritti nell'atmosfera dei primi anni Ottanta.



Per quanto riguarda il gameplay, la regola è quella solita e sacrosanta: ripulisci lo schermo, abbuffati di pillole speciali per trasformare i cacciatori in prede ed evita il contatto fisico come se si trattasse di un parente





invadente a Natale. Facile a dirsi, un po' meno a farsi!

I comandi richiedono una precisione da neurochirurgo: svoltare l'angolo con mezzo microsecondo di ritardo significa piombare dritti tra le braccia del «fantasma» di turno, con lo stesso entusiasmo di chi sta andando a pagare le tasse. A me è capitato proprio mentre lo provavo durante un mitico gameplay sul mio canale YouTube... che umiliazione!

I nemici, poi, sono degli imprevedibili geni del male. Rimanere incastrati in un vicolo cieco nell'esatto millisecondo in cui svanisce l'effetto della super-pillola è un grande classico della commedia tragica. In quel secondo di terrore puro, mentre vedete la vostra vittima trasformarsi nuovamente nel vostro carnefice, la tentazione di sfilare la cassetta dal registratore, lanciarla dalla finestra e fingersi svenuti è fortissima. Soprattutto per me, che ho la pazienza di un gatto davanti a un cetriolo!

Però, quando, tra un attacco di panico e l'altro, riuscirete a inghiottire l'ultimo pallino allo stremo delle forze, la soddisfazione sarà talmente grande che vi verrà voglia di premiare il VIC-20 con un bel bacio sulla scocca!

Veniamo quindi al dunque... Snapman è senza dubbio un piccolo classico che dimostra come non servano grafiche pompate per farvi battere il cuore a mille. È semplice, immediato, spietato al punto giusto e conserva ancora oggi quel fascino ingenuo e irresistibile dei primi videogiochi casalinghi.

Preparate il joystick, perché, una volta inserita la cassetta, sarà molto più difficile smettere di giocare che iniziare!

di **Chiara "Kika" Carrara**

GIUDIZIO FINALE

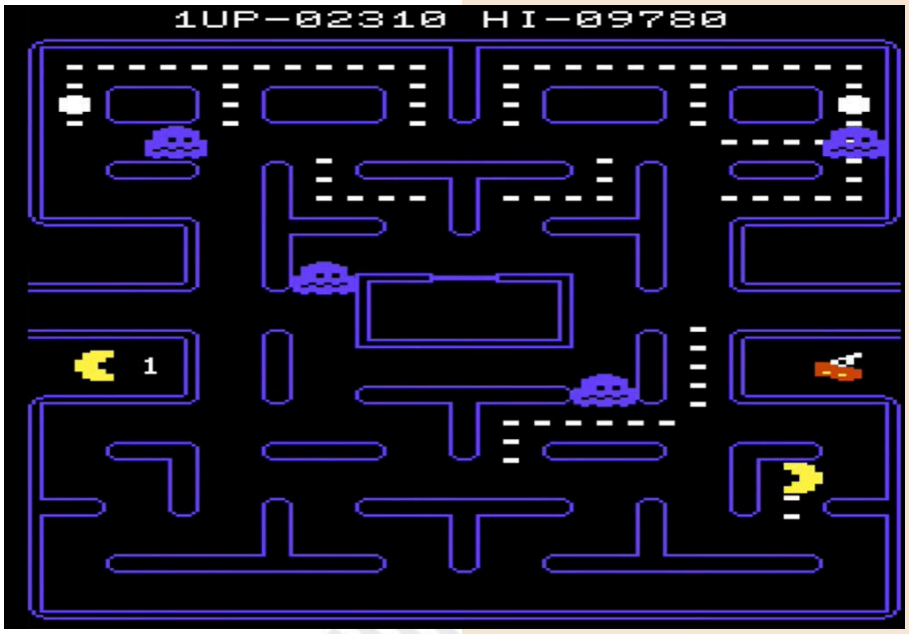
» Giocabilità 78%

Niente fronzoli, niente menu complessi: premete il tasto e siete subito immersi nel gioco. I comandi un po' rigidi e i nemici imprevedibili aggiungono quel pizzico di sfida che trasforma ogni partita in una vera e propria questione d'orgoglio. Basta una svolta sbagliata per passare dalla gloria alla tragedia nel giro di mezzo secondo.

» Longevità 70%

Il concetto è semplicissimo, ma la voglia di stracciare il vostro record provoca una discreta assuefazione. Tra un «Ok, adesso stacco davvero» e un «No, vabbè, ne provo un'altra!», la serata vola via in un attimo. E, naturalmente, l'ultima partita non è mai veramente l'ultima.





BINKY: FROZEN IN TIME

Anno: 2026
Editore/Sviluppatore: Thalamus Digital Publishing
Genere: Platform
Piattaforma: Commodore 64
Sito web: https://thalamusdigital.itch.io/binky-frozen



Martedì sera. L'idea di partenza era dedicarsi a una rilassante skincare con tanto di maschera al viso e candela profumata. L'idea reale è stata accendere il C64 per "provare un attimo" Blinky: Frozen in Time, la nuova chicca targata Thalamus. Risultato? Maschera argillosa completamente crepata sulla faccia a furia di stringere i denti, cremina dimenticata sul comodino e io avvinghiata al joystick col fiato sospeso. A prima vista sembra il solito platform colorato e coccoloso. Dopotutto il protagonista è Binky, un simpaticissimo draghetto. Un po' sfortunato, va detto: ha perso il suo alito congelante, ma compensa con un superpotere bizzarro... congela qualsiasi cosa tocchi con i piedi!

La missione sembra semplice: attraversi la stanza, sali e scendi per scale, nastri trasportatori e ascensori, raccogli monete, trovi la chiave per aprire i bauli con le lettere della parola B-O-N-U-S e ti assicuri di congelare ogni singolo pezzo di pavimento a forza di camminarci sopra. Solo a quel punto si attiva il portale per il livello successivo.

Facile? Nemmeno per sogno. Qui devi pianificare ogni passo come un geometra del joystick: il timer corre e, proprio mentre assapori la vittoria, ecco che arrivano nemici infami a scongelare le mattonelle che avevi appena sistemato. Vederti vanificare tutto all'ultimo secondo fa salire una carogna indescrivibile!



Ma la vera chicca è che ogni livello è un omaggio ai grandi classici del passato. Un'idea poetica e bellissima... finché non ti trovi davanti a quel trauma di Jet Set Willy.





GIUDIZIO FINALE

» Giocabilità 86%

I comandi rispondono che è un piacere e muovere Binky è uno spasso, ma il livello di sfida non regala niente. Calcolare i percorsi al millimetro per congelare ogni mattonella richiede concentrazione assoluta, pazienza e una buona dose di sangue freddo.

» Longevità 86%

Tra la caccia alle lettere B-O-N-U-S, i bauli da aprire, le chiavi da scovare e il continuo desiderio di superare la stanza che vi ha fatto impazzire, vi ritroverete ad afferrare il joystick per ore dimenticando completamente il mondo esterno (e la skincare!).

Quando ho visto spuntare quel maledetto nemico in stile Miner Willy, mi è venuto un colpo. Ma dico io... con tutti i giochi belli della storia, proprio lui dovevano riesumare?! Io Willy lo odio con tutto il cuore, è la rovina della mia infanzia e il motivo per cui negli anni '80 ho rischiato l'esaurimento nervoso! Vederlo sfrecciare sullo schermo come se niente fosse mi fa salire un travaso di bile istantaneo. Ogni volta che lo vedo arrivarmi contro, la tentazione di prendere il joystick e lanciarlo direttamente dal balcone non è solo forte... è un bisogno fisico!







Tutto sommato Blinky: Frozen in Time è una gemma per Commodore 64 capace di unire la nostalgia degli anni '80 a una cura moderna davvero spettacolare. Un'avventura fresca, colorata e spietata al punto giusto, a cui è davvero impossibile non volere bene... anche se la comparsa a tradimento di Miner Willy, a thalamus, non la perdonerò mai!

di **Chiara "Kika" Carrara**



YOKOXEVI

Anno: 2026
Editore/Sviluppatore: GW's Workshop
Genere: Shoot'em up
Piattaforma: MSX, 8Kb
Sito web: https://www.msxdev.org/2026/09/16/msxdev26-01-yokoxevi/

Quando Xevious, MSX e scorrimento orizzontale si incontrano… Il risultato non poteva che essere esplosivo. YokoXevi, sviluppato da GW's-Workshop per il concorso MSXdev26, prende il classico shoot'em up verticale di Namco e lo ruota di novanta gradi, trasformandolo in uno sparatutto a scorrimento orizzontale.



Il risultato è un gioco che sembra semplice solo finché non cominciano ad arrivare nemici, proiettili e ostacoli da ogni direzione… proprio come Xevious! La cosa più sorprendente è che tutto questo viene fatto su un MSX1 con appena 8 KB di RAM e 16 KB di VRAM, sfruttando scrolling multipiano, sprite multiplexati, campioni audio PSG e una gestione grafica molto avanzata per l'hardware. L'ispirazione è evidente, ma YokoXevi non si limita a copiare il modello originale. Il giocatore controlla una piccola navetta che deve farsi strada attraverso paesaggi pieni di nemici, strutture e bersagli da distruggere.

La visuale orizzontale modifica completamente il ritmo dell'azione. Invece di concentrarsi soltanto su ciò che arriva dall'alto, bisogna tenere sotto controllo l'intero spazio davanti alla navetta, cercando di anticipare i movimenti degli avversari e di evitare i proiettili. Il gioco mantiene comunque l'idea di utilizzare attacchi differenti contro bersagli aerei e terrestri. Questa separazione aggiunge un minimo di strategia a un sistema altrimenti immediato: non basta sparare in continuazione, bisogna anche scegliere il bersaglio corretto.



La vera protagonista di YokoXevi è la tecnologia. Il gioco utilizza più piani di parallasse per creare profondità e movimento, mentre gli sprite multiplexati permettono di visualizzare un numero elevato di elementi senza mandare completamente in crisi il sistema.



Lo scrolling è uno degli aspetti più riusciti. Per un computer del 1983, riuscire a mantenere un movimento fluido e continuo in uno shoot'em up orizzontale è già un risultato notevole.





GIUDIZIO FINALE

» Giocabilità 85%
Xevious con un'altra prospettiva e un ottimo controllo di gioco.

» Longevità 75%
Difficile. Richiede pratica per essere padroneggiato.



L'effetto è ancora più impressionante considerando i requisiti minimi del progetto. La grafica non è ricca di dettagli come quella dei moderni homebrew per MSX2, ma la scelta è coerente con l'obiettivo. I colori, gli sfondi e gli elementi sullo schermo sono organizzati per garantire leggibilità e ritmo, non per mostrare effetti spettacolari a ogni costo. Anche il comparto sonoro è stato curato con grande attenzione. YokoXevi include campioni audio riprodotti attraverso il PSG e sfrutta il canale a 1 bit per effetti e voce. Le informazioni sul progetto citano inoltre campioni PCM sincronizzati con batteria e voce. L'audio contribuisce a dare energia alle partite e rende il gioco più spettacolare di quanto ci si aspetterebbe da una produzione per MSX1. Non siamo davanti a una colonna sonora orchestrale, naturalmente, ma a una dimostrazione di come si possano ottenere risultati interessanti lavorando con risorse limitate.

Parlando di gameplay puro, YokoXevi è un gioco arcade nel senso più classico del termine: si parte, si spara, si cerca di sopravvivere e si prova a migliorare il punteggio. La struttura non sembra costruita intorno a una lunga modalità storia, ma a partite brevi e ripetibili. Il ritmo aumenta gradualmente e obbliga il giocatore a rimanere concentrato. Nei primi momenti è possibile studiare il comportamento dei nemici, ma presto lo schermo si riempie di minacce e la situazione diventa molto meno tranquilla. La difficoltà potrebbe sembrare severa a chi è abituato agli shoot'em up moderni, ma è coerente con la tradizione arcade. Il gioco invita a imparare dai propri errori e a riconoscere i pattern dei nemici. Se si viene colpiti, non è sempre colpa del destino, spesso è semplicemente colpa del fatto che si stava guardando il punteggio invece dei proiettili.

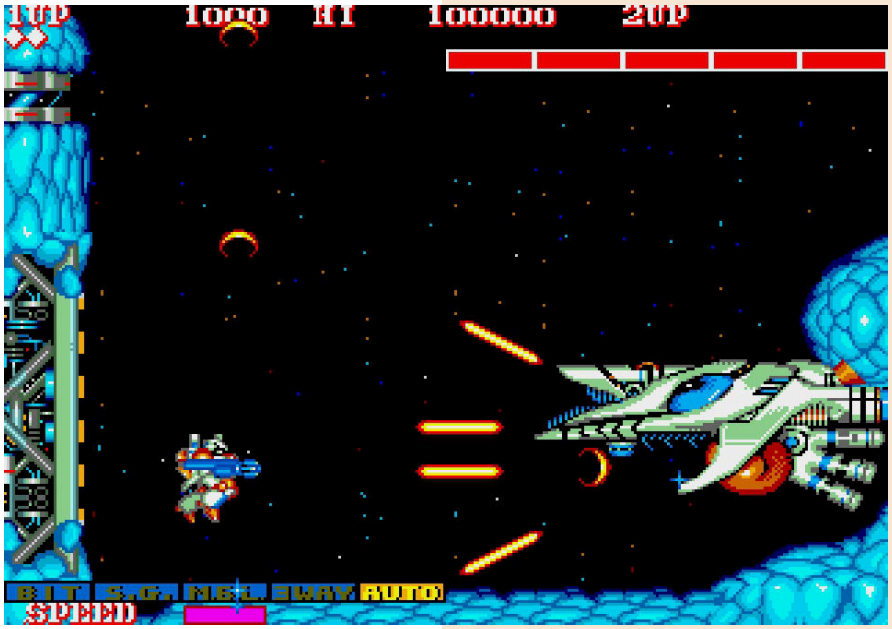
di **Giampaolo Moraschi**





SIDE ARMS MD

Anno: 2026
Editore/Sviluppatore: rester159
Genere: Shoot'em up
Piattaforma: Sega Mega Drive
Sito web: https://rester159.itch.io/side-arms



Nel 1986 Capcom lanciava nelle sale giochi Hyper Dyne Side Arms, uno shoot 'em up che metteva da parte le classiche astronavi per infilarci dentro due soldati corazzati, Henry e Sanders, ultima speranza contro l'invasione aliena dei Bozon. Quarant'anni dopo, quella battaglia torna a infuriare su Mega Drive grazie a rester159, autore di un port amatoriale tanto inatteso quanto interessante.

Side Arms MD nasce infatti da zero: non è una ROM modificata, ma una ricostruzione pensata appositamente per l'hardware Sega, capace di girare a velocemente e di riproporre tutte le sezioni e i boss dell'arcade, compresa la modalità cooperativa per due giocatori. Un'assenza storica finalmente colmata, visto che Side Arms arrivò all'epoca su numerosi computer, Amiga e Atari ST compresi, e su PC Engine, ma saltò completamente Mega Drive.

Il gioco conserva la caratteristica che nel 1986 lo rendeva immediatamente riconoscibile. Possiamo sparare sia verso destra sia verso sinistra, necessità tutt'altro che secondaria quando i nemici decidono di arrivare da ogni angolo dello schermo. I classici POW permettono inoltre di migliorare velocità e arsenale, passando fra Spread Gun, Mega Bazooka, colpi multidirezionali e altre delizie ad alto contenuto esplosivo. Nell'originale era persino possibile fondere i due combattenti in una singola e potentissima unità.



Su Mega Drive tutto questo funziona magnificamente. Lo scrolling è fluido, l'azione veloce e la quantità di sprite restituisce quella piacevole sensazione da sala giochi in cui fermarsi a pensare equivale generalmente a esplodere. La struttura può apparire semplice rispetto agli shooter arrivati negli





GIUDIZIO FINALE

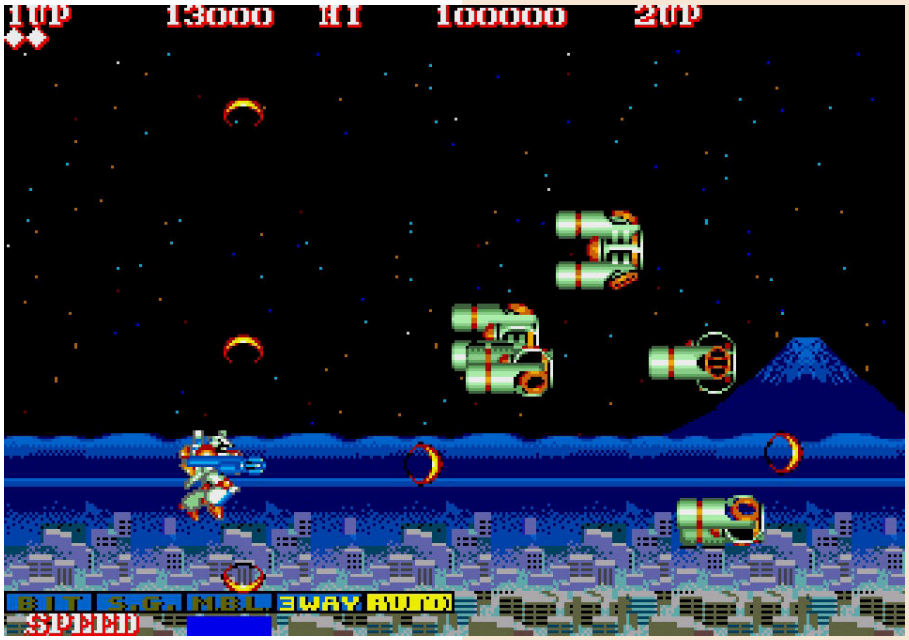
» Giocabilità 80%
Meccaniche interessanti anche a distanza di 40 anni.

» Longevità 78%
Non semplice ne rilassante… ma puramente arcade!

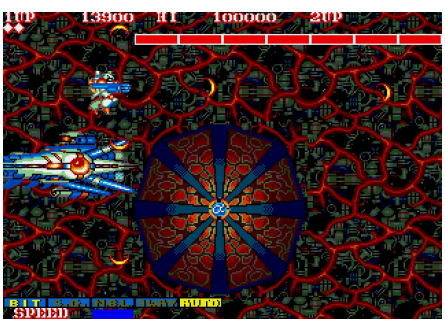
anni successivi, ma Side Arms possiede ancora ritmo, personalità e un sistema di armamenti che invita continuamente a cambiare strategia.

Non è soltanto un recupero nostalgico: Side Arms MD sembra proprio uno di quei giochi che avremmo voluto trovare sugli scaffali Sega all'inizio degli anni Novanta. È arrivato con quasi quarant'anni di ritardo? Pazienza. Henry e Sanders avevano semplicemente perso l'uscita per il Mega Drive.

di **Roberto Del Mar Pirazzini**





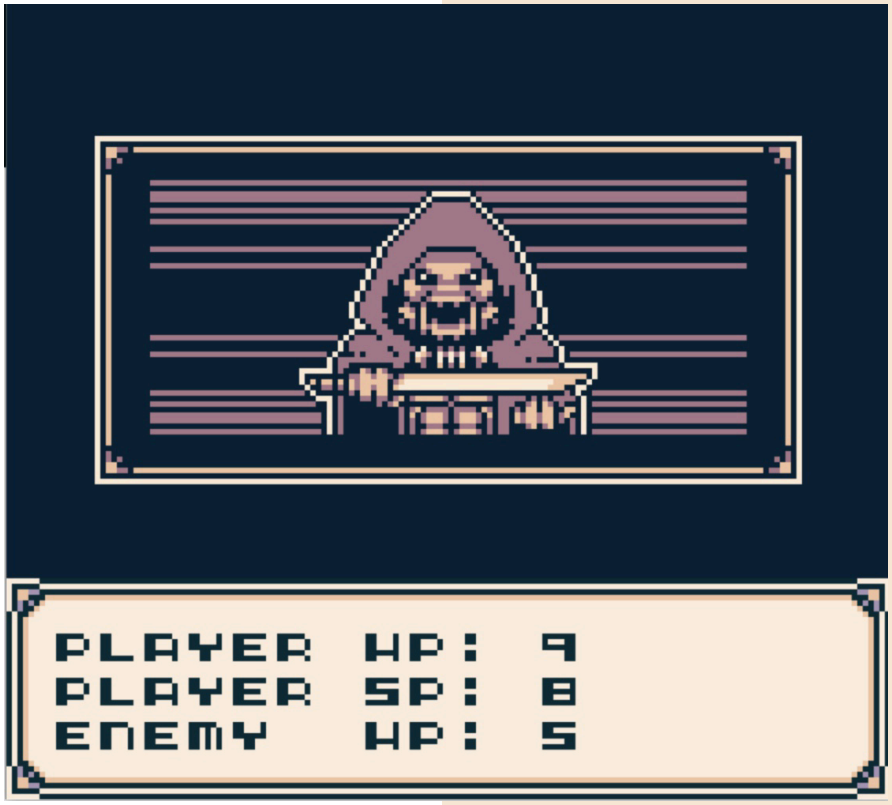




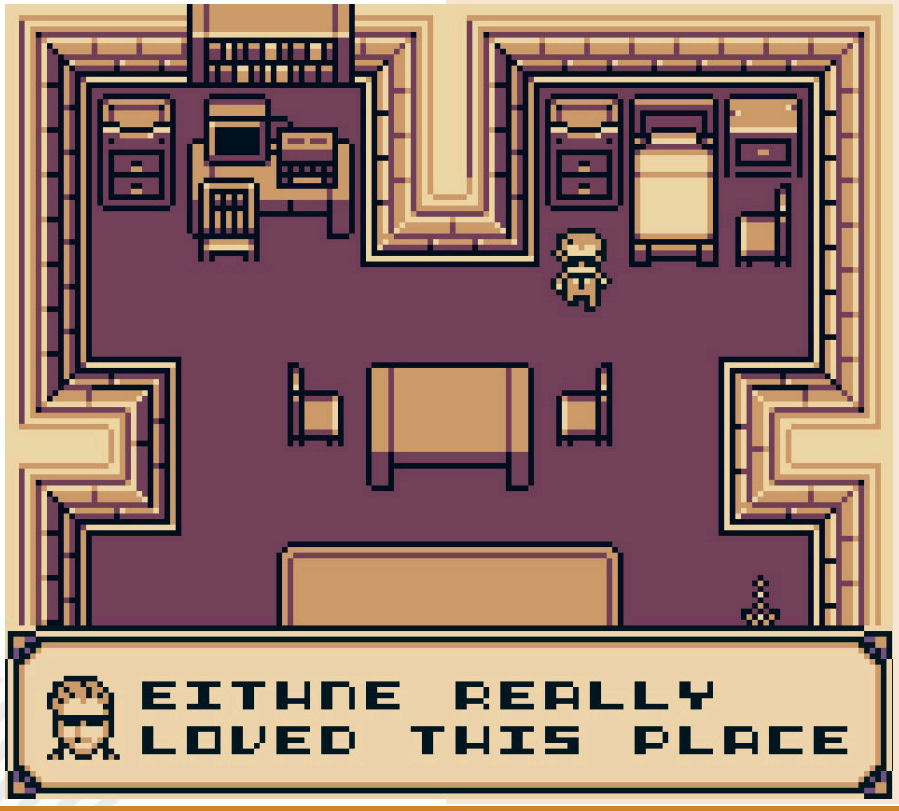
WORM KING

Anno: 2026
Editore/Sviluppatore: Oniric Factor
Genere: Adventure/RPG
Piattaforma: Game Boy
Sito web: https://oniric-factor.itch.io/the-worm-king

Nel panorama sempre più vivace degli homebrew per console classiche, Worm King di Oniric Factor si distingue per una proposta insolita: un'avventura horror investigativa sviluppata per il Game Boy originale. Il gioco è nato circa cinque anni fa utilizzando GB Studio 1.0, quando il programma era ancora molto più limitato rispetto alle versioni attuali, ed è arrivato soltanto ora alla sua versione definitiva. Questa lunga gestazione si riflette nel risultato finale, che conserva il fascino di un progetto autenticamente legato ai primi anni dello sviluppo indipendente su Game Boy.



L'ambientazione si ispira all'horror cosmico e al folklore americano tra Lovecraft ed E.A. Poe. Eventi misteriosi stanno sconvolgendo una piccola comunità, mentre qualcosa di antico sembra muoversi sotto la superficie del mondo conosciuto. Il giocatore veste i panni di un investigatore chiamato a comprendere la natura di questi avvenimenti, esplorando gli ambienti, raccogliendo indizi e cercando di ricostruire una verità sempre più inquietante.



Fin dalle prime battute, Worm King punta soprattutto sull'atmosfera. La grafica in bassa risoluzione e la tavolozza limitata del Game Boy non vengono trattate come semplici limitazioni tecniche, ma diventano strumenti per alimentare il mistero. Gli ambienti sono essenziali, i personaggi vengono rappresentati con pochi elementi e molti dettagli rimangono volutamente nell'ombra. Il risultato è un mondo compatto e



GIUDIZIO FINALE

» Giocabilità 90%

La combinazione tra salute, sanità mentale, oggetti, indizi e combattimenti in prima persona funziona come base per un'esperienza compatta e coerente.

» Longevità 80%

Alcuni limiti derivano inevitabilmente dalla lunga gestazione e dagli strumenti utilizzati durante lo sviluppo, ma fanno anche parte dell'identità del progetto. Worm King è consigliato agli appassionati di horror retro, agli amanti degli homebrew e a chi cerca un'avventura Game Boy diversa dal solito.

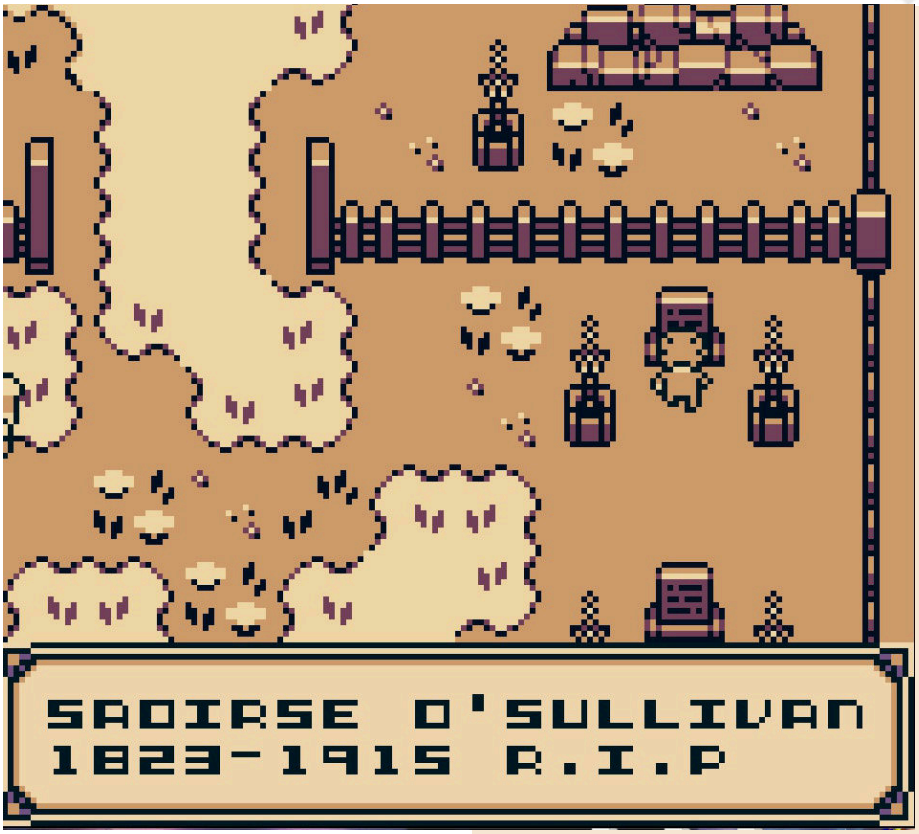
ambiguo, nel quale ciò che non si vede può essere più disturbante di ciò che viene mostrato. La struttura di gioco combina esplorazione dall'alto, investigazione, gestione delle risorse e combattimenti. L'avventura richiede di osservare con attenzione gli ambienti, parlare con i personaggi e raccogliere gli oggetti utili alla progressione. Non è presente il tradizionale sistema basato su esperienza e livelli. Il protagonista non diventa più forte semplicemente affrontando nemici, ma deve imparare a gestire ciò che ha a disposizione e a comprendere come utilizzare ogni elemento trovato durante l'esplorazione. Questa scelta contribuisce a mantenere il ritmo più vicino a quello di un'avventura horror che a quello di un RPG tradizionale, scelta che personalmente apprezzo. Non c'è motivo di fermarsi a eliminare continuamente gli stessi avversari per accumulare punti esperienza. La progressione dipende soprattutto dalla capacità del giocatore di collegare gli indizi, individuare il percorso corretto e affrontare con attenzione le situazioni più pericolose.

Un ruolo importante è affidato alla gestione di salute e sanità mentale. La presenza di due indicatori distinti è coerente con la natura dell'esperienza: il pericolo non riguarda soltanto l'incolumità fisica, ma anche la lucidità del protagonista. In un contesto ispirato all'orrore cosmico, vedere la propria sanità mentale diminuire contribuisce a trasmettere una sensazione di vulnerabilità e di perdita del controllo. L'equipaggiamento e gli oggetti acquistabili assumono quindi un peso significativo. Le risorse non sono infinite



e il giocatore deve valutare con attenzione come prepararsi, quali oggetti conservare e quando utilizzarli. Questo sistema rende l'esplorazione più tesa, perché anche un incontro apparentemente secondario può mettere in difficoltà chi non ha gestito correttamente le proprie risorse.

Gli scontri adottano una soluzione particolare. Mentre l'esplorazione si svolge con visuale dall'alto, i combattimenti passano alla prima persona. Il cambio di prospettiva separa nettamente i momenti investigativi dalle fasi d'azione e dà agli incontri un carattere più diretto. Sul piano estetico, questa scelta richiama alcuni RPG e giochi horror del passato, adattandoli alle possibilità tecniche del Game Boy. L'efficacia del sistema dipende naturalmente dalla varietà delle situazioni e dalle opzioni offerte al giocatore. I combattimenti non sembrano essere il fulcro dell'esperienza, ma piuttosto uno strumento per sottolineare la pericolosità del mondo. Worm King resta infatti un gioco basato sull'esplorazione, sulla raccolta di informazioni e sulla scoperta graduale della trama. Gli scontri servono a ricordare che il protagonista non è un eroe invincibile, ma una persona esposta a forze che non comprende pienamente. La scelta di mantenere l'esperienza compatta è adatta alla formula. Un'avventura investigativa horror troppo lunga rischierebbe di diluire la tensione o di trasformare la ricerca degli indizi in una sequenza ripetitiva. Worm King sembra invece puntare su una campagna contenuta, pensata per essere completata in poche ore e per mantenere costante l'attenzione del giocatore. La durata ridotta potrebbe però rappresentare un limite per chi cerca un RPG ampio, ricco di aree da esplorare e di contenuti secondari. Anche la struttura investigativa potrebbe richiedere qualche tentativo e un certo

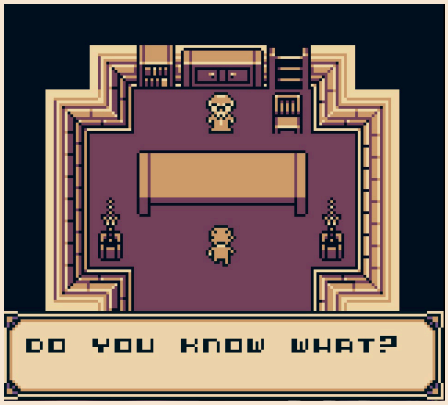


backtracking, soprattutto nei momenti in cui il gioco non indica in modo esplicito il prossimo obiettivo. In un'avventura horror questo senso di smarrimento può essere positivo, ma diventa meno efficace quando gli indizi non sono abbastanza chiari.

Il progetto presenta inoltre un interesse particolare per chi segue la scena dello sviluppo retro. Essendo stato realizzato inizialmente con GB Studio 1.0, Worm King testimonia i limiti e le possibilità degli strumenti disponibili nella prima fase di diffusione del motore. Non è un gioco moderno che imita superficialmente il Game Boy ma un progetto nato direttamente all'interno di quel contesto produttivo e successivamente portato a compimento. Il gioco è distribuito digitalmente in un file ROM. La versione disponibile è in inglese e la durata contenuta la rende adatta anche a sessioni brevi. Il prezzo, inoltre, è accessibile e in linea con la natura indipendente del progetto. In conclusione, Worm King è un homebrew interessante e originale, soprattutto per il modo in cui utilizza l'estetica del Game Boy per costruire un'atmosfera inquietante. Non è un RPG tradizionale e non punta sulla quantità dei contenuti, ma propone un'avventura investigativa breve, basata sull'esplorazione, sulla gestione delle risorse e sulla progressiva scoperta di una minaccia antica.

di **Ingrid Poggiali**





NEW GAME

CANNON FODDER STE

Anno: 2026
Editore/Sviluppatore: KtZ
Genere: Action game
Piattaforma: Atari STE
Sito web: https://ataricne.org/news/index.php/cannon-fooder-atari-ste-version

Esistono videogames che non hanno bisogno di presentazioni e Cannon Fodder è sicuramente uno di questi. Quando Sensible Software lo pubblicò nei primi anni Novanta riuscì a fondere strategia, azione e humour britannico in un gioco controllabile praticamente con due pulsanti del mouse. Un piccolo capolavoro diventato rapidamente uno dei titoli simbolo della sua generazione. Anche l'Atari ST ricevette la propria conversione nel 1994. Il gioco era sostanzialmente tutto lì. Le missioni, gli scenari, i piccoli soldati destinati a diventare veterani o lapidi e quella straordinaria immediatezza che rendeva Cannon Fodder accessibile dopo pochi minuti e impegnativo per molte ore. Eppure, bastava spostare la visuale lateralmente per accorgersi che qualcosa non andava.

Lo scrolling orizzontale procedeva a scatti. Non era un difetto sufficiente a rovinare Cannon Fodder, tanto che le recensioni dell'epoca furono comunque eccellenti, ma risultava particolarmente fastidioso in un gioco nel quale il movimento della telecamera costituisce parte integrante dell'azione.



Sono trascorsi più di trent'anni ed ecco che arriva Krzysztof "KtZ" Jarzyna, decidendo che era giunto il momento di sistemare le cose.

Nasce così questa nuova versione di Cannon Fodder per Atari, ancora indicata come work in progress ma perfettamente giocabile, nella quale l'utilizzo del Blitter permette finalmente di ottenere uno scrolling molto più fluido. Non siamo davanti a un remake e nemmeno a una conversione completamente nuova. È piuttosto una profonda revisione tecnica dell'originale, capace di mostrarci come avrebbe potuto essere Cannon Fodder su Atari se alcune caratteristiche della macchina fossero state sfruttate maggiormente nel 1994. La differenza si nota immediatamente! Muovendo il puntatore verso i bordi dello schermo, lo scenario segue l'azione con una continuità sconosciuta alla vecchia conversione. Può sembrare una modifica marginale, ma in Cannon Fodder cambia radicalmente la percezione del gioco. La telecamera è infatti parte dell'interfaccia. Bisogna continuamente osservare ciò che circonda la squadra, localizzare i nemici, individuare torrette e edifici, seguire i proiettili e capire rapidamente quale percorso utilizzare. Lo scrolling poco fluido della versione originale interferiva inevitabilmente con questa lettura dello scenario. Ora tutto appare più naturale.

Il puntatore si sposta, la visuale lo segue e improvvisamente emerge quanto Cannon Fodder fosse già un ottimo gioco anche su Atari, semplicemente penalizzato da un limite tecnico molto evidente.

KtZ non si è comunque fermato allo





GIUDIZIO FINALE

» Giocabilità 94%
Gameplay semplici ma tattico e con lo scrolling ora è perfetto su STE.

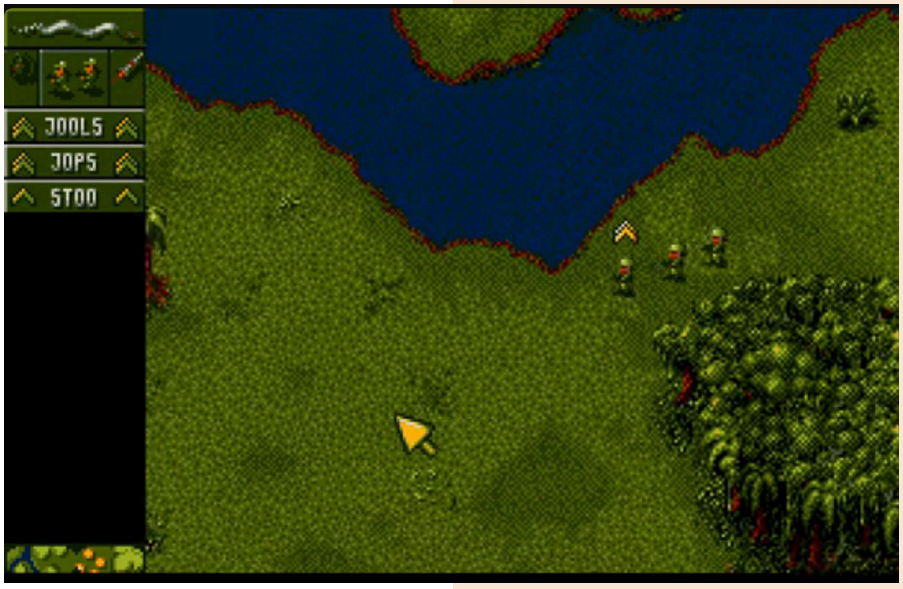
» Longevità 96%
Difficilmente ci si stanca di giocare a Cannon Fodder... praticamente eterno!

scrolling, ma ha introdotto anche alcuni miglioramenti alla presentazione e al comparto sonoro. Nell'introduzione troviamo musica MOD riprodotta in stereo e immagini derivate dalla versione Amiga, visualizzate in sedici tonalità di grigio, mentre sono presenti effetti sonori digitalizzati. Nelle schermate tra una missione e l'altra rimane invece la caratteristica musica chip.

Sono interventi che non stravolgono l'originale ma contribuiscono a creare la sensazione di trovarsi davanti a un'edizione finalmente costruita tenendo maggiormente conto dell'hardware Atari. La nuova versione offre anche un'ottima occasione per scoprire quanto poco sia invecchiato il gameplay ideato da Sensible Software. Con il mouse controlliamo un piccolo gruppo di soldati: un comando permette di indicare dove muoversi e di aprire il fuoco, mentre il secondo pulsante viene utilizzato per le armi speciali. La squadra può inoltre essere divisa in gruppi separati. È tutto incredibilmente semplice.

Eppure, procedendo nelle missioni, Cannon Fodder diventa progressivamente più tattico. Lanciarsi contro il nemico sparando può funzionare inizialmente, ma presto diventa necessario osservare lo scenario, sfruttare le distanze, conservare granate e razzi e scegliere attentamente il percorso. Jeep, carri armati ed elicotteri modificano ulteriormente il ritmo, aumentando temporaneamente mobilità e potenza di fuoco. Ma il vero colpo di genio rimane la gestione dei soldati.

Ogni uomo possiede un nome e i sopravvissuti vengono promossi. Jools, Jops e compagni smettono quindi rapidamente di essere semplici sprite. Dopo averli guidati attraverso numerose missioni cominciamo inevitabilmente





ad affezionarci a loro. Quando muoiono, non tornano! Tra una missione e l'altra la celebre collina mostra le tombe dei caduti, mentre una fila apparentemente infinita di nuove reclute aspetta di essere mandata al fronte.

Dietro il celebre motto "War has never been so much fun" si nasconde così una satira della guerra sorprendentemente efficace. Cannon Fodder riesce a essere divertente e allo stesso tempo piuttosto amaro senza interrompere mai l'azione per spiegarci il proprio messaggio... e di questi tempi è oro!

Naturalmente il progetto riapre anche una vecchia questione. Quanto delle differenze tra le conversioni Amiga e Atari ST fosse realmente dovuto all'hardware e quanto invece alle modalità con cui venivano realizzati i porting. Negli anni Novanta gli sviluppatori lavoravano con scadenze e budget precisi, spesso dovendo convertire rapidamente lo stesso titolo su numerose piattaforme. Oggi uno sviluppatore appassionato può invece trascorrere settimane ottimizzando un singolo aspetto del programma.

Il lavoro di KtZ non dimostra quindi che ST e Amiga fossero macchine equivalenti. Dimostra qualcosa di più interessante... nella conversione originale di Cannon Fodder era rimasto del potenziale inutilizzato.

Non è nemmeno il primo tentativo di migliorare il gioco. Nel 2015 Peter Putnik aveva già realizzato una versione capace di sfruttare l'audio DMA dell'Atari STE, introducendo musica digitalizzata durante il gioco. Rimaneva però irrisolto proprio il problema più evidente: lo scrolling. KtZ sembra finalmente aver chiuso il cerchio.

Ed è questa una delle caratteristiche più affascinanti dell'attuale scena retrocomputing. Questi computer non vengono soltanto conservati o emulati. Continuano a essere studiati e programmati e, qualche volta, gli sviluppatori contemporanei riescono persino a correggere compromessi accettati più di trent'anni fa.

Concludendo, Cannon Fodder STE



funziona proprio perché non cerca di trasformare il gioco in qualcosa di diverso. Gli sprite rimangono minuscoli, gli scenari mantengono l'aspetto originale e non compaiono improbabili effetti grafici moderni. È ancora Cannon Fodder.

Semplicemente, adesso si muove come avrebbe dovuto. Lo scrolling più fluido rende l'azione maggiormente leggibile e il controllo più naturale. Dopo qualche minuto, si finisce addirittura per dimenticare che stiamo giocando una versione modificata. Ed è probabilmente il miglior complimento che possiamo fare al lavoro di KtZ.

Cannon Fodder era un grande gioco nel 1994 e continua a esserlo oggi. Quando la telecamera comincia finalmente a seguire dolcemente Jools, Jops e gli altri soldati attraverso il campo di battaglia, viene spontaneo chiedersi come sarebbe stata accolta questa versione trent'anni fa. Poi arriva una granata, perdiamo uno dei nostri veterani e sulla collina compare un'altra lapide. E improvvisamente siamo di nuovo nel 1994.

War has never been so much fun... Questa volta, su Atari, non è mai stata nemmeno così fluida.

di **Carlo Nithaiah Del Mar Pirazzini**



NEW GAME

HYPER RUNNER

Anno: 2026
Editore/Sviluppatore: Dark Age Software, Raster Wizards
Genere: Driving game
Piattaforma: Amiga AGA
Sito web: https://darkagesoftware.it/

Nascono nuovi homebrew per celebrare il passato e altri che sembrano volerlo ignorare completamente, comportandosi come se gli ultimi trent'anni non fossero mai trascorsi. Hyper Runner, sviluppato da Raster Wizards, appartiene decisamente alla seconda categoria. Accenderlo significa trovarsi davanti a qualcosa che non vuole semplicemente ricordarci l'Amiga, ma vuole essere, senza troppe cerimonie, un nuovo videogioco per Amiga.



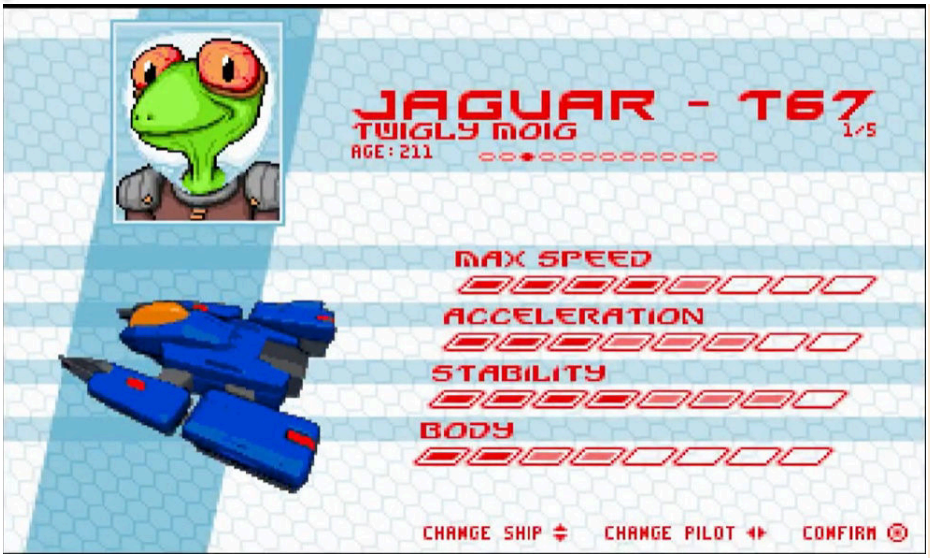
Raster Wizards lavora al progetto da diversi anni e Hyper Runner ha attraversato una lunga fase di creazione prima di arrivare alla sua forma attuale. Il risultato è un racing game fantascientifico progettato per i sistemi dotati di chipset AGA, costruito intorno a una filosofia completamente arcade. Qui si corre senza fiato, si evitano gli ostacoli, si gestisce l'integrità del mezzo e si cerca di raggiungere il traguardo prima che il tempo finisca.

Una formula apparentemente semplice, dietro la quale si nasconde però un sistema di gioco decisamente più interessante di quanto possa sembrare. Fin da subito, Hyper Runner non perde tempo. Ti fa scegliere il mezzo, selezionare il pilota e si parte. Proprio questa fase iniziale introduce però una delle caratteristiche più interessanti del gioco.

Sono disponibili cinque differenti veicoli e dodici piloti, e la scelta del personaggio non rappresenta una semplice variazione estetica. Ogni pilota influenza le caratteristiche del mezzo, modificando parametri quali velocità massima, accelerazione, manovrabilità e resistenza. Il risultato è un sistema che invita alla sperimentazione. Puntare esclusivamente sulla velocità potrebbe sembrare inizialmente la soluzione migliore, ma un mezzo molto rapido e difficile da controllare può diventare problematico nelle sezioni più impegnative. Una maggiore resistenza, al contrario, permette di sopportare qualche errore senza compromettere immediatamente la corsa. Dopo qualche partita si comincia inevitabilmente a cercare la combinazione più adatta al proprio stile di guida. Una volta sulla pista, la prima sensazione è quella che dovrebbe trasmettere qualsiasi buon racing arcade: velocità … e vi assicuro bimbi… che la velocità c'è!

Lo scenario scorre rapidamente verso il giocatore, gli elementi aumentano progressivamente di dimensione e gli altri mezzi diventano ostacoli da valutare e superare in pochi istanti.





Hyper Runner è immediato, ma sarebbe sbagliato interpretarlo come un gioco nel quale sia sufficiente tenere premuto l'acceleratore.

Uno degli elementi fondamentali è infatti il booster, capace di aumentare considerevolmente la velocità. La tentazione di utilizzarlo continuamente è forte, ma la sua riserva energetica è limitata. Quando non viene attivato l'energia torna lentamente disponibile, trasformando il booster in una risorsa da amministrare con attenzione.

Utilizzarlo su un rettilineo, conservarlo per recuperare dopo un errore oppure rischiare tutto quando il cronometro sta per raggiungere lo zero sono decisioni che devono essere prese rapidamente. Ed è proprio questa continua gestione del rischio a dare carattere alle gare. Il vero avversario in gioco è soprattutto il tempo.

Bisogna raggiungere le varie sezioni prima che il conto alla rovescia termini, ma guidare in maniera eccessivamente aggressiva può essere altrettanto pericoloso. Le collisioni fanno perdere velocità e danneggiano il mezzo; se la sua integrità viene completamente compromessa, la nostra corsa termina. Durante le prime partite si reagisce semplicemente a ciò che compare sullo schermo. Dopo qualche tentativo, però, qualcosa cambia… si comincia ad anticipare.

Una curva particolarmente difficile non arriva più inaspettata. Un rettilineo diventa il punto ideale per utilizzare il booster. Un ostacolo che inizialmente ci aveva sorpreso viene evitato quasi automaticamente.

La pista smette progressivamente di essere soltanto uno scenario e diventa qualcosa da apprendere e memorizzare. Ad aumentare la varietà interviene la struttura stessa del gioco. Hyper Runner ci mostra quindici stage e durante la corsa possiamo incontrare dei bivi che permettono di scegliere quale direzione seguire. Una singola partita non consente quindi di vedere tutto ciò che il gioco offre. È una soluzione semplice ma efficace, che ci ricorda tanto Out Run, perché aggiunge curiosità alla voglia di migliorare. Terminata una corsa rimane inevitabilmente la domanda… cosa avremmo trovato scegliendo l'altra strada?

Naturalmente esiste un solo modo per scoprirlo, premere start e ricominciare. Come da tradizione, lungo i vari percorsi compaiono inoltre diversi power-up. Alcuni riparano il mezzo, altri ripristinano l'energia del booster, mentre altri ancora aggiungono secondi preziosi al cronometro. La loro presenza introduce ulteriori decisioni, ad esempio deviare dalla traiettoria ideale per raccogliere un bonus può essere fondamentale, ma può anche costare tempo prezioso.

Graficamente Hyper Runner cattura immediatamente l'attenzione. Raster Wizards utilizza una rappresentazione pseudo-tridimensionale della pista nella quale scaling, prospettiva e movimento degli elementi contribuiscono a creare un convincente senso di profondità.

Gli oggetti aumentano rapidamente di

GIUDIZIO FINALE

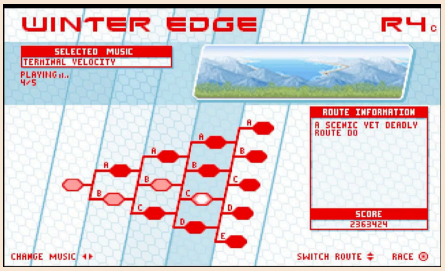
» Giocabilità 90%

Tanti piloti e veicoli differenti e 15 livelli tutti da scoprire. Difficoltà bilanciata durante il percorso e una marea di power up… Dimenticavo… c'è la musica di Allister Brimble!!

» Longevità 90%

Un gioco che va "maneggiato con cura". Regala il vero senso di velocità, ma richiede un po' di pratica per essere apprezzato.





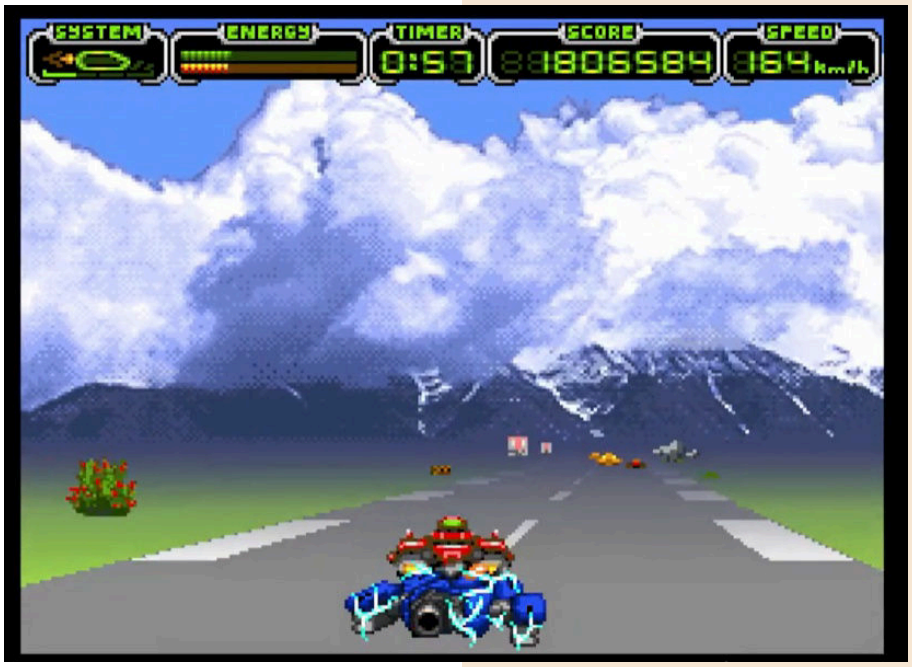


dimensione avvicinandosi al giocatore, i mezzi occupano progressivamente una porzione maggiore dello schermo e lo scenario sembra scorrere sotto di noi. Ma a colpire è soprattutto l'utilizzo del colore.

Il chipset AGA permette di creare ambientazioni vivaci, gradienti e fondali che conferiscono al gioco una personalità immediatamente riconoscibile, grazie anche ad un universo fantascientifico super colorato. La velocità rappresenta però il vero effetto speciale. Quando il booster entra in funzione, la sensazione di accelerazione aumenta sensibilmente e tutto sembra arrivare verso il giocatore ancora più rapidamente. Anche i controlli contribuiscono positivamente all'esperienza. In un gioco così rapido la risposta del mezzo deve essere immediata, perché non esiste il tempo per effettuare continue correzioni. Dopo alcune partite le differenze tra le varie combinazioni di veicolo e pilota diventano più evidenti e trovare quella maggiormente adatta al proprio modo di giocare diventa parte del divertimento.

La qualità forse più importante di Hyper Runner emerge proprio quando una gara termina. Si pensa immediatamente a ciò che avremmo potuto fare diversamente.Quel booster utilizzato troppo presto. Quella collisione evitabile. Quel bonus mancato. Quel bivio nel quale abbiamo scelto una direzione invece dell'altra…. e arriva inevitabilmente la voglia di riprovare. Il gioco costruisce buona parte del proprio fascino proprio su questa dinamica. Non cerca di trattenere il giocatore attraverso sistemi inutilmente complessi, ma attraverso la sensazione che la partita successiva possa essere migliore della precedente. Ed è forse questo l'aspetto più riuscito dell'intero progetto.

Per giocare a questa bellezza occorre attendere ottobre per scaricarlo in download digitale, ma potete ordinare la vostra copia fisica dal sito di Dark Age. Consigliata un po' di Fast Ram per poter sfruttare a pieno la velocità



complessiva.

Hyper Runner è ben sviluppato e mescola la fantascienza di F-ZERO con Out Run in modo ottimale.

È divertente correre?

La risposta arriva nel momento in cui il cronometro scende, il checkpoint è ancora lontano e decidiamo di consumare l'ultima parte di energia del booster.

Lo schermo accelera.

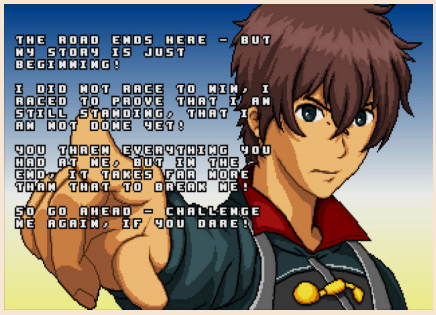
Il tempo continua a scendere.

E noi stringiamo il joystick.

Tre. Due. Uno. GO!

di **Carlo Nithaiah Del Mar Pirazzini**



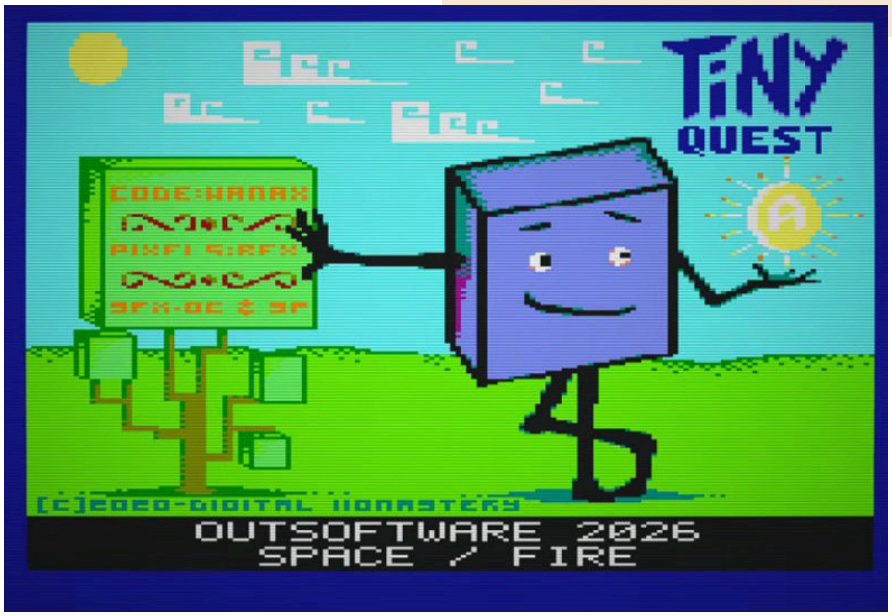




NEW GAME

TINY QUEST

Anno: 2026
Editore/Sviluppatore: Outsoftware (basandosi sui codici originali)
Genere: Platform game
Piattaforma: Amstrad CPC, Atari ST/Falcon
Sito web: vedi ultima pagina

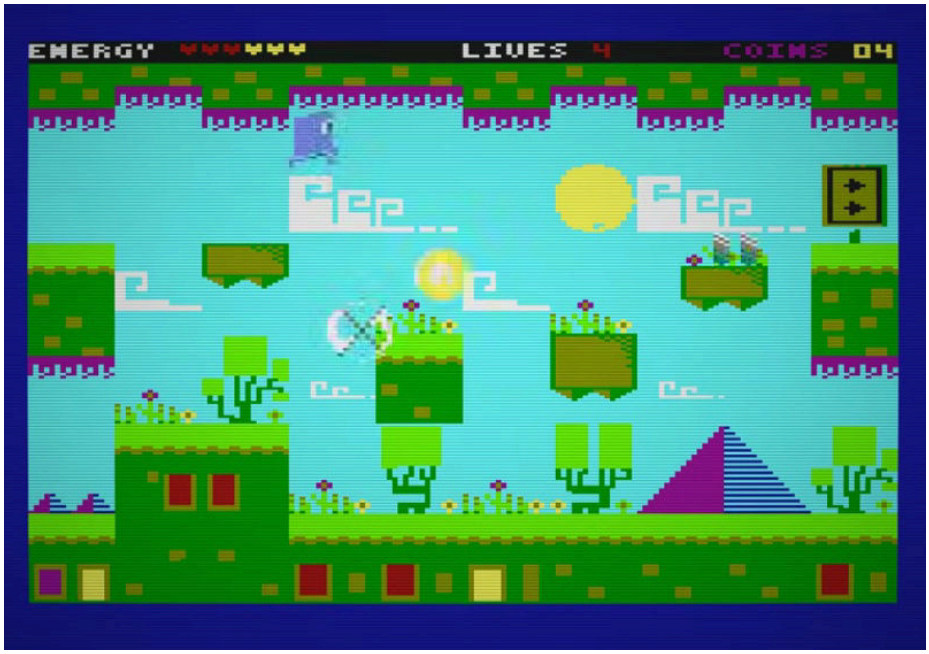


Nel panorama dell'homebrew dedicato ai computer a 8 e 16 bit, Tiny Quest è uno di quei giochi capaci di conquistare al primo sguardo. Nato su Commodore 64 (recensito nel n° 28 di RMW) nel 2020 per opera di Andrea "Wanax" Schincaglia, con grafica di Federico "Raffox" Sesler e musiche di Gaetano Chiummo e Stefano "Dustbin" Palmonari, il gioco è recentemente approdato su Amstrad CPC grazie a Outsoftware e, poco dopo, anche su Atari ST e Falcon. La storia è semplice, quasi fiabesca. Mr Cube si è innamorato, ma la sua compagna vive dall'altra parte del mondo. Il nostro eroe è anche completamente al verde e dovrà raccogliere monete durante il viaggio per pagare il ritorno a casa di entrambi. Il problema è che il tempo corre veloce e l'energia si esaurisce rapidamente. In altre parole, bisogna arrivare, raccogliere il necessario e non morire: l'amore può attendere, il timer decisamente no. Cosa non si fa per amore, vero?

In soldoni, siamo di fronte ad un platform a schermata fissa costruito attorno a una regola molto precisa: ogni livello è un piccolo problema da risolvere in pochi secondi. Mr Cube deve saltare sulle piattaforme, evitare trappole e creature pericolose, raccogliere le monete presenti e raggiungere l'uscita. Quest'ultima si attiva soltanto dopo aver completato l'obiettivo dello schermo, costringendo il giocatore a pianificare il percorso anziché correre alla cieca. Il ritmo è frenetico, ma non dipende soltanto dai riflessi. Servono memoria, capacità di osservazione e soprattutto una buona dose di autocontrollo. Nei primi livelli è possibile procedere d'istinto; andando avanti, invece, ogni salto deve essere preparato con attenzione. Una piattaforma apparentemente innocua può trasformarsi in una trappola, mentre un percorso che sembra impossibile diventa praticabile dopo averne studiato tempi e sequenze. Il sistema di controllo è immediato e ridotto all'essenziale; movimento e salto. Proprio per questo, quando si sbaglia, è difficile dare la colpa ai comandi. Il gioco chiede precisione e raramente concede una seconda possibilità. Ogni errore costa una delle cinque vite disponibili e, una volta esaurite, la partita termina.

A rendere l'esperienza più gestibile intervengono le password, assegnate al completamento di ogni gruppo di quindici livelli. Sono checkpoint utili per riprendere il viaggio dopo un





Game Over, anche se presentano una caratteristica curiosa, cambiano a ogni ricaricamento e restano valide soltanto durante la sessione di gioco. Non sono quindi password permanenti nel senso tradizionale, ma un sistema pensato per rendere meno frustrante la progressione senza eliminare la tensione.
Dal punto di vista estetico, Tiny Quest possiede una personalità immediatamente riconoscibile.
Il mondo è costruito attorno a forme geometriche: alberi quadrati, nuvole squadrate, personaggi ridotti a blocchi colorati e ambientazioni che sembrano uscite da un libro illustrato realizzato con righello e mattoncini. La grafica non cerca il realismo e non ha bisogno di farlo. Ogni elemento è leggibile, funzionale al gameplay e inserito in una palette vivace. Il risultato è allegro, coerente e molto "da home computer", ma anche abbastanza elegante da non sembrare una semplice esercitazione tecnica.

La versione Amstrad CPC

Il porting CPC nasce in modo piuttosto particolare. Outsoftware ha raccontato di aver dedicato circa tre settimane alla conversione di oltre trenta titoli verso piattaforme spesso trascurate dalla scena homebrew, tra cui Sam Coupé, Einstein 256, Timex 2068 e Sinclair QL. Tra questi progetti è entrato anche Tiny Quest, grazie al codice sorgente C64 messo a disposizione dallo stesso Andrea Schincaglia. Il risultato è una conversione fedele nello spirito e nella struttura e, soprattutto, il gameplay resta intatto. La versione CPC richiede 128 KB di RAM, un requisito da tenere presente per chi utilizza un modello reale o un emulatore configurato in modo conservativo. Il passaggio all'Amstrad non modifica la natura del gioco, ma gli dona una colorazione differente. La palette è molto brillante e, secondo alcuni di noi in redazione, persino un po' troppo accesa. È una scelta che può dividere, ma si sposa bene con il carattere spensierato del titolo. La conversione dimostra inoltre quanto il CPC sia ancora adatto a ospitare platform colorati e tecnicamente ambiziosi, soprattutto quando il progetto nasce con una struttura a schermate fisse.

L'arrivo su Atari ST/Falcon

La versione per Atari ST rappresenta il capitolo più recente e, per certi versi, più curioso. Il gioco è stato portato su Atari ST, STE e Falcon, con supporto sia per la bassa risoluzione a colori sia per l'alta risoluzione monocromatica. Il porting funziona con 1 MB di memoria e nasce grazie al sorgente C64 fornito dall'autore originale. Su ST il gioco cambia naturalmente presenza scenica. Il Motorola 68000 offre un ambiente molto più potente rispetto al 6502 del C64, ma Tiny Quest non cerca di trasformarsi in un platform

GIUDIZIO FINALE

» Giocabilità 85%

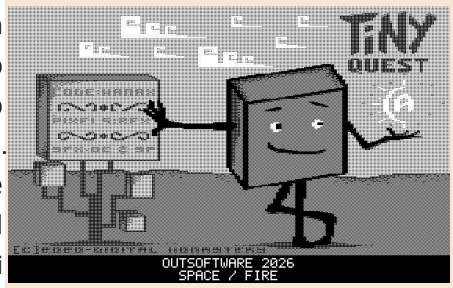
In entrambe le versioni resta invariato il sistema di gioco. Il sistema di gioco è semplice da comprendere, ma abbastanza profondo da richiedere pratica, memoria e precisione. La difficoltà elevata può allontanare chi preferisce esperienze più permissive, ma rappresenta anche il principale motivo per cui si continua a giocare.

» Longevità 85%

Un platform piccolo solo nel titolo… che offre una sfida capace di occupare parecchio spazio nella memoria muscolare.







completamente diverso. La forza della conversione sta proprio nel conservare l'impostazione originale, adattandola alle possibilità dell'hardware Atari. La modalità a colori valorizza maggiormente il carattere vivace delle ambientazioni, mentre il supporto all'alta risoluzione monocromatica è una chicca particolarmente interessante per chi possiede un monitor Atari tradizionale. Non è soltanto un'aggiunta tecnica: dimostra l'attenzione del porting verso configurazioni diverse, comprese quelle meno orientate al gioco. Il risultato è probabilmente meno sorprendente sul piano tecnico rispetto alla versione C64, semplicemente perché l'ST avrebbe risorse sufficienti per gestire qualcosa di molto più complesso. Tuttavia, proprio questa fedeltà è il suo pregio principale. Tiny Quest rimane riconoscibile, giocabile e coerente, senza perdere il suo carattere nel passaggio a una macchina più potente.

Dopo l'analisi delle due versioni arriviamo al rush finale. Il difetto principale di Tiny Quest coincide anche con la sua identità: la difficoltà. Il tempo a disposizione è spesso severo e alcuni livelli sembrano progettati per far consumare vite con una rapidità quasi offensiva. Il gioco, però, non si affida al caso. Ogni schermata può essere completata e, dopo i primi tentativi, la disposizione degli ostacoli comincia a diventare leggibile.

La frustrazione nasce più dalla necessità di ripetere un'azione già conosciuta che da problemi di programmazione. È il classico meccanismo da arcade, quello di osservare, fallire, memorizzare, riprovare. Chi cerca un platform rilassato potrebbe trovarlo eccessivamente punitivo; chi ama migliorare il proprio percorso schermata dopo schermata troverà invece una sfida sorprendentemente appagante. La longevità non dipende soltanto dal numero dei livelli, ma dalla volontà di superarli davvero. Tiny Quest non è un gioco da finire distrattamente in una sera: è un piccolo percorso di perfezionamento, nel quale ogni errore insegna qualcosa.



di **Giampaolo Moraschi** (Versione ST) e **Marta Rossmann** (Versione CPC)

Sito web ST:
https://www.atarimania.com/games/atari-st-games-tiny-quest-46068

Sito web CPC:
https://forum.cpcwiki.eu/t/tiny-quest/499





NEW GAME

WOLF64

Anno: 2026
Editore/Sviluppatore: Kweepa
Genere: FPS
Piattaforma: Commodore 64
Sito web: https://kweepa.itch.io/wolf64

Portare Wolfenstein 3D su un Commodore 64 standard sembrava un'impresa quasi impossibile. L'amato home computuer dispone di un processore 6502 a circa 1 MHz, memoria limitata e un hardware progettato soprattutto per la grafica bidimensionale. Il titolo id Software, invece, richiede un motore capace di simulare ambienti tridimensionali, texture, nemici e combattimenti in tempo reale... eppure... eccolo qui! Wolf64, progetto sviluppato da Kweepa, dimostra che l'impossibile può diventare possibile con una buona dose di coding, ottimizzazione e follia creativa.



Si tratta di un porting non ufficiale di Wolfenstein 3D per Commodore 64, realizzato interamente in Assembly 6502 e pensato per funzionare su un C64 standard dotato di disk drive. Il gioco comprende il primo episodio shareware, con otto livelli e lo scontro finale contro Hans Grosse. È distribuito gratuitamente per uso non commerciale e può essere utilizzato su hardware reale, emulatori, SD2IEC, EasyFlash e altri dispositivi compatibili.

La caratteristica più impressionante di Wolf64 è la presenza di un vero motore raycasting. Il giocatore può muoversi liberamente nei corridoi, ruotare la visuale, aprire porte e combattere i nemici, proprio come nella versione originale per PC. Il progetto non usa schermate prerenderizzate né trucchi per simulare soltanto l'effetto tridimensionale. Calcola la distanza dalle pareti, disegna le colonne verticali e gestisce la posizione degli oggetti nello spazio. Per ridurre il carico sul processore vengono utilizzate tabelle precalcolate e tecniche di ottimizzazione specifiche per il 6502. Le texture sono state adattate alla palette del Commodore 64 e in parte ripixelate manualmente. Il risultato è naturalmente più semplice rispetto alla versione DOS, ma riesce a ricreare l'impressione di trovarsi in un labirinto tridimensionale. La finestra di gioco utilizza una modalità multicolore estesa e una risoluzione ridotta, necessaria per mantenere una velocità accettabile. È un compromesso evidente, ma anche una scelta intelligente e accettabile.





GIUDIZIO FINALE

» Giocabilità 90%
Il classico degli fps con alcune aggiunte interessanti come il salvataggio e la possibilità di saltare il livello. Ottima la possibilità di utilizzare il mouse e il joystick a doppio pulsante.

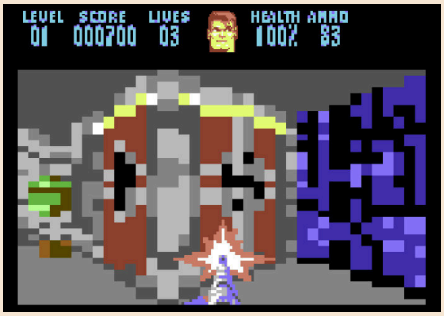
» Longevità 85%
Il gioco è bello e divertente ma gli accorgimenti tecnici spesso limitano la visibilità in alcuni punti.

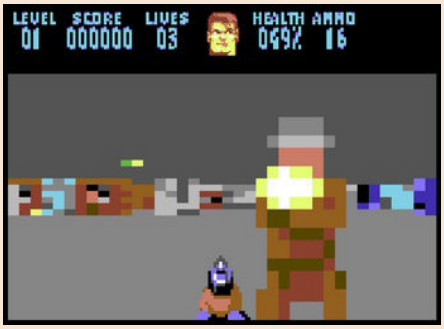
Meglio una visuale più piccola e leggibile che uno schermo più grande completamente immobile.

Il contenuto principale riproduce il primo episodio di Wolfenstein 3D. Nei panni di B.J. Blazkowicz bisogna attraversare otto livelli pieni di corridoi, stanze, guardie, cani e ufficiali. La struttura è quella classica: esplorare, cercare chiavi, raccogliere munizioni e tesori, aprire porte e raggiungere l'uscita. I livelli sono labirintici e invitano a controllare ogni stanza, anche perché le aree segrete possono contenere armi, salute e oggetti utili. La disposizione dei corridoi ripropone il feeling dell'originale, anche se la navigazione richiede un po' di pratica. Con una visuale più piccola e texture ridotte, alcuni passaggi possono sembrare simili tra loro. È quindi facile ritrovarsi a percorrere più volte lo stesso corridoio chiedendosi se il problema sia la mappa o il proprio senso dell'orientamento. Le prime versioni del progetto non includevano tesori e punteggio, probabilmente per risparmiare memoria. Gli aggiornamenti successivi hanno però reintrodotto entrambi gli elementi, insieme a vite extra, correzioni al comportamento dei nemici e ulteriori ottimizzazioni per la grafica.

Il sistema di combattimento è immediato. Si comincia con un coltello e una pistola, mentre nei livelli successivi è possibile raccogliere armi più potenti e munizioni. Gli scontri richiedono di controllare la posizione dei nemici e di reagire rapidamente agli attacchi. I soldati possono trovarsi davanti, ai lati o dietro il giocatore, rendendo importante ruotare spesso la visuale e non avanzare alla cieca. Le prestazioni sono generalmente sorprendenti per l'hardware utilizzato, anche se la fluidità non è sempre costante. Quando lo schermo contiene molti oggetti o nemici, il ritmo può rallentare. Gli aggiornamenti usciti in seguito alla prima release hanno migliorato il disegno delle pareti e la gestione delle scene affollate, ma i limiti del C64 rimangono inevitabili. Un altro problema riguarda la leggibilità. Alcuni nemici possono confondersi con le pareti, soprattutto nelle zone in cui le uniformi hanno colori simili allo sfondo. In certe situazioni ci si accorge della presenza di una guardia soltanto quando è già abbastanza vicina da rendere la conversazione molto breve.

Si tratta di un difetto comprensibile, ma concreto. La versione DOS era più ricca di colori e dettagli, mentre il C64 deve gestire raycasting, texture, sprite, collisioni e intelligenza dei nemici con risorse estremamente limitate. Wolf64 può essere giocato con joystick e tastiera. I comandi permettono di muoversi, ruotare e sparare, mentre le versioni







più recenti includono anche il movimento laterale tramite tastiera.

È disponibile il supporto al mouse 1351 e a periferiche compatibili. In ambiente VICE è possibile configurare il mouse collegandolo alla porta di controllo; su hardware reale sono stati effettuati test positivi con dispositivi come MouSTer. Il suo utilizzo migliora sensibilmente la precisione e rende i combattimenti più naturali. Il joystick resta perfettamente utilizzabile, ma la gestione della visuale e degli scontri è più comoda con una periferica a puntamento. E' previsto anche il supporto di joystick con due pulsanti, una soluzione interessante per chi utilizza periferiche moderne compatibili con il Commodore 64.

Il gioco è stato rilasciato in diversi formati. Il file D64 può essere utilizzato in VICE oppure trasferito su un C64 reale tramite un'unità 1541, SD2IEC o dispositivi analoghi. È disponibile anche un file CRT destinato a EasyFlash. Per ridurre i tempi d'attesa viene utilizzato il fastloader di Krill/Plush. La versione fastloader, però, presenta una limitazione: permette di caricare rapidamente il gioco, ma non di salvare i progressi. L'avvio può richiedere qualche configurazione, soprattutto in VICE. È necessario utilizzare l'immagine corretta e configurare adeguatamente il drive per evitare blocchi alla schermata del titolo. Non è un ostacolo grave per chi frequenta la scena C64, ma l'esperienza sarebbe più immediata con istruzioni ancora più dettagliate o con pacchetti già pronti per le principali configurazioni.

Il salvataggio è stato uno degli aspetti più richiesti dalla comunità. Le versioni iniziali non prevedevano una funzione tradizionale, ma gli aggiornamenti hanno introdotto quick save e quick load tramite i tasti F5 e F7. È inoltre disponibile una funzione per raggiungere rapidamente un livello usando la combinazione F3+W. Queste aggiunte sono importanti perché



completare un episodio intero in una sola sessione non è sempre pratico. I livelli sono relativamente brevi, ma il ritmo lento e l'esplorazione possono allungare la durata delle partite.
Kweepa ha inoltre spiegato di aver utilizzato strumenti basati sull'intelligenza artificiale per alcune conversioni, prototipi e blocchi di codice. Il controllo tecnico del progetto è comunque rimasto nelle sue mani, attraverso la definizione degli algoritmi, delle strutture dati e delle funzioni da mantenere o eliminare.

Wolf64 è uno dei progetti homebrew più impressionanti realizzati per Commodore 64. Il suo valore non dipende soltanto dal nome che porta, ma dal modo in cui riesce a tradurre il concetto di Wolfenstein 3D su un hardware completamente diverso. Non è una copia perfetta della versione DOS e non pretende di esserlo. È piuttosto una reinterpretazione tecnica e appassionata, capace di dimostrare che il Commodore 64 può ancora sorprendere.

di **Carlo Nithaiah Del Mar Pirazzini**



NEW GAME

BABYLONIAN TWINS

Anno: 1993/2026
Editore/Sviluppatore: bithunch
Genere: Platform/Puzzle
Piattaforma: Amiga
Sito web: https://babyloniantwins.com/1993/



Due principi, un regno dimenticato e un Amiga 500 che sfidò persino il deserto. Questo avvenne nel cuore di Baghdad, durante un'estate rovente del 1993, dove tre giovani studenti iracheni stavano compiendo una piccola impresa destinata a entrare nella storia dei videogiochi. Fuori, il termometro sfiorava i 50 gradi, l'elettricità era un lusso intermittente e le sanzioni economiche rendevano praticamente impossibile procurarsi qualsiasi componente informatico. Eppure, davanti a un solitario Commodore Amiga 500, quei ragazzi continuavano a programmare, riga dopo riga, un'avventura che avrebbe visto la luce soltanto molti anni dopo. Il suo nome? Babylonian Twins.

Sviluppato interamente in linguaggio Assembly, il gioco rappresenta una testimonianza straordinaria della passione e della determinazione dei suoi creatori. Si racconta persino che il caldo infernale abbia messo fuori uso il lettore floppy per ben tre volte! Altro che problemi di connessione: qui il vero boss finale era riuscire ad accendere il computer senza provocare un barbecue elettronico.

Considerato il primo videogioco commerciale realizzato in Iraq, Babylonian Twins rimase a lungo un progetto incompiuto dal punto di vista editoriale. Il declino di Commodore contribuì a impedirne la commercializzazione e il titolo finì nel dimenticatoio, trasformandosi progressivamente in una piccola leggenda tra gli appassionati Amiga. Dopo anni di discussioni e curiosità sui forum dell'English Amiga Board, il gioco è finalmente arrivato a una distribuzione digitale gratuita nel settembre 2026.

Ma veniamo al gioco, perché dietro questa incredibile storia si nasconde un platform che non ha alcuna intenzione di trattarci con i guanti.

L'avventura ci trasporta tra gli antichi palazzi dell'Assiria, dove due principi gemelli devono collaborare per superare labirinti, trappole e ostacoli di ogni genere. La meccanica fondamentale consiste nell'alternare il controllo dei due fratelli, sfruttandone le differenti abilità per risolvere enigmi ambientali e aprire nuovi passaggi. Interruttori da azionare, percorsi da





liberare e trappole da evitare trasformano ogni livello in un piccolo rompicapo, dove ragionare è importante almeno quanto possedere riflessi felini.

Il comparto grafico è indubbiamente uno dei maggiori punti di forza. Gli scenari, disegnati pixel dopo pixel, ricostruiscono con grande attenzione suggestivi ambienti orientali, tra architetture monumentali, rovine e decorazioni che restituiscono un'atmosfera affascinante. Il risultato richiama piacevolmente le suggestioni di Arabian Nights di Core Design, pur conservando una propria identità.

Anche il sonoro contribuisce a rendere memorabile l'esperienza. Le melodie di ispirazione mediorientale accompagnano l'esplorazione con efficacia, immergendo il giocatore in un mondo antico e misterioso. Lo scrolling, inoltre, è sorprendentemente fluido e permette di affrontare le sezioni più movimentate senza fastidiosi rallentamenti.

Fin qui sembrerebbe tutto meraviglioso, vero? Ebbene, preparatevi a conoscere il lato sadico degli antichi Babilonesi... perché Babylonian Twins è difficile. Maledettamente difficile.

Il level design non perdona distrazioni, le trappole sembrano progettate da un architetto particolarmente vendicativo e basta un movimento sbagliato per compromettere una partita. Come se non bastasse, mancano completamente salvataggi e password... ergo...se sbagliate nel momento meno opportuno, potreste ritrovarvi nuovamente davanti alla schermata iniziale, accompagnati da un vocabolario di imprecazioni che probabilmente non compare su nessuna tavoletta cuneiforme.

Questa struttura decisamente vecchia scuola finisce per penalizzare l'accessibilità e rischia di scoraggiare chi non possiede una pazienza degna di un monaco tibetano. Tuttavia, superata la frustrazione iniziale, emerge un puzzle-platform solido, intelligente e tecnicamente molto curato, capace di

GIUDIZIO FINALE

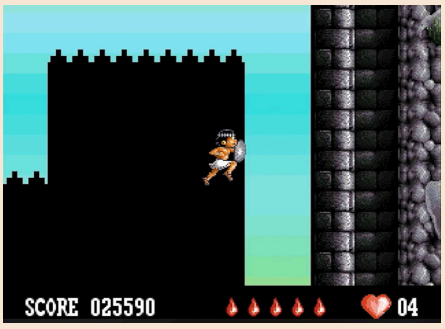
» Giocabilità 80%

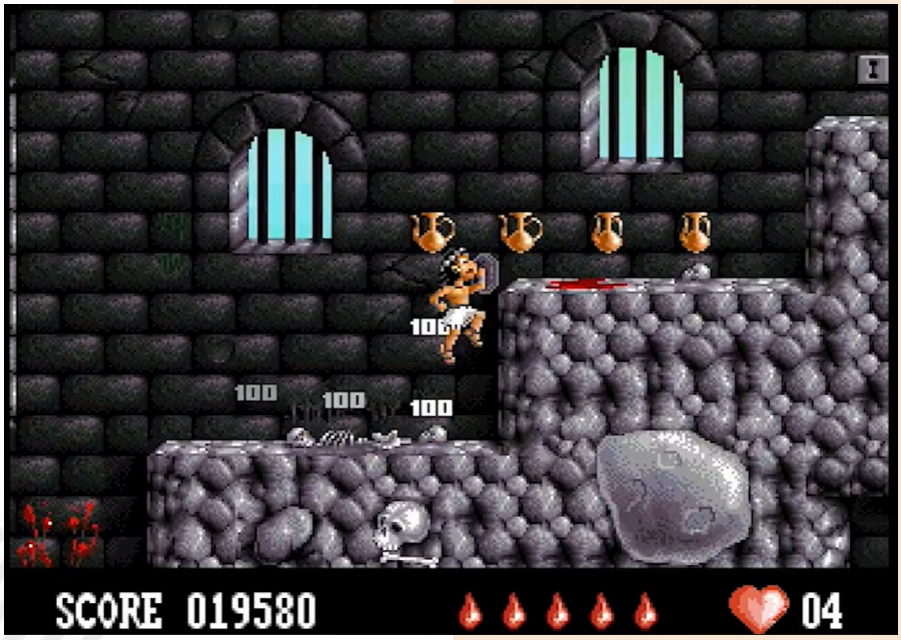
Interessante combinazione tra puzzle e platform game. Comandi semplici e un buon design dei livelli.

» Longevità 68%

Difficile e infernale! L'assenza di save point o di password è micidiale. Per fortuna in emulazione si può salvare lo stato... Su real... le imprecazioni cadranno copiose dal cielo.









offrire soddisfazioni autentiche a chi ama le sfide impegnative.

Babylonian Twins è molto più di una semplice curiosità archeologica del software. È il risultato della determinazione di tre ragazzi che, in condizioni proibitive, riuscirono a trasformare un Amiga 500 in uno strumento di creatività e libertà. Un'opera che merita di essere conosciuta non soltanto per il suo valore storico, ma anche per le qualità ludiche che ancora oggi riesce a esprimere.

Disponibile gratuitamente in formato ADF, utilizzabile su emulatori Amiga e hardware compatibile, rappresenta un'occasione imperdibile per riscoprire una pagina poco conosciuta della storia del nostro amato computer.

E ricordate: nell'antica Mesopotamia inventarono la scrittura, la ruota e tante altre meraviglie della civiltà. Evidentemente, però, nessuno aveva ancora pensato di inventare il salvataggio della partita!

Babylonian Twins è una di quelle riscoperte che ricordano perché amiamo ancora l'Amiga. Grafica ben fatta, scrolling impeccabile, musiche evocative e un gameplay intelligente ne fanno un piccolo gioiello dimenticato. Peccato per la difficoltà decisamente brutale e l'assenza di salvataggi, capaci di trasformare una tranquilla serata di retrogaming in una crisi diplomatica con il joystick. Ma dietro ogni trappola si nasconde una storia straordinaria, fatta di passione, sacrifici e puro talento. Un pezzo di storia videoludica che merita finalmente il suo posto sotto i riflettori.



di **Carlo Nithaiah Del Mar Pirazzini**



DRACULA'S CASTLE

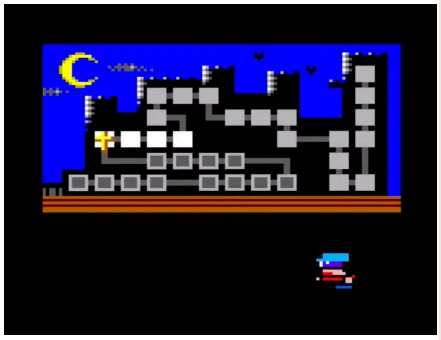
Anno: 2021
Editore/Sviluppatore: Game Impact, Habit Soft
Genere: Puzzle game
Piattaforma: Sega Game Gear

Dimenticate castelli gotici, croci benedette e interminabili frustate! In Dracula's Castle il mestiere dell'eroe consiste soprattutto nel correre, schivare mostri e raccogliere tesori prima che il tempo scada!

Sviluppato da Habit Soft e pubblicato da Game Impact nel 2021, questo titolo per Game Gear è la conversione a colori dell'omonimo gioco uscito su Game Boy nel 2019. La missione? Salvare la principessa Psycho (un nome che è tutto un programma) attraversando 32 livelli a schermata fissa, aprendo tre forzieri nascosti per sbloccare l'uscita.

Pipistrelli, licantropi e mummie renderanno la passeggiata meno rilassante del previsto. Coltelli e attacchi di fuoco saranno preziosi alleati, soprattutto contro le mummie, immuni alle altre armi. Il gameplay è semplice, immediato e decisamente arcade, con una difficoltà che richiede di studiare i movimenti nemici e scegliere il momento giusto per agire. La grafica colorata e il sonoro gradevole valorizzano l'hardware Sega, mentre la struttura a livelli invita a giocare anche per brevi sessioni. Peccato per l'assenza di password: perdere i progressi dopo aver superato una lunga serie di schermate è una punizione degna del Conte in persona! Dracula's Castle è dunque un piccolo gioiello per gli amanti del retrogaming portatile con tanto divertimento vecchia scuola. Se amate i labirinti e avete pazienza da vendere, preparatevi a fare gli straordinari al servizio della principessa!

di **Ingrid Poggiali**











GIUDIZIO FINALE

» Giocabilità 78%
Semplice e si lascia giocare con gusto.

» Longevità 65%
Peccato per l'assenza di un sistema di password o salvataggio.

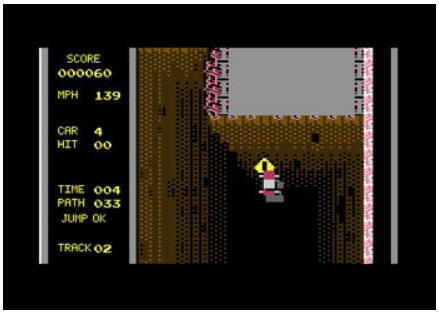
NEW GAME

RUOTE FUMANTI

Anno: 2026
Editore/Sviluppatore: Charger0277
Genere: Racing game
Piattaforma: Commodore 64

Questo nuovo titolo sta risvegliando in me piacevoli ricordi. Senza nemmeno finire la frase avrete già capito di quale gioco sto parlando, ma non riesco a fare nome e cognome di colui che ci ha tenuti incollati al cabinato del bar latteria sotto casa e nelle sale giochi in vacanza, con valanghe di gettoni e monete da 200 lire (Burning Rubber, NdFF). E dopo tanti anni, stiamo parlando addirittura del secolo scorso, ci viene proposto questo gioiellino molto simile allo storico titolo arcade, con un nome tutto italiano: "Ruote Fumanti". Quando lo si sente, viene da pensare a una rivista di automobili tamarrissime e super-elaborate.

Il gioco si presenta subito bene, con una colonna sonora molto piacevole e rilassante che ci tiene compagnia per l'infinito tragitto su un rettilineo pieno di pericoli, come auto nemiche - tra cui l'odiosa macchina della polizia che cerca di speronarci - e tante altre. Come se non bastasse, ci saranno precipizi e distese d'acqua da superare; ma ecco che ci iviene in soccorso il particolare di questo gioco: il salto! Ebbene sì: premendo il pulsante fire, l'automobile spiccherà un balzo che la farà ingrandire ai nostri occhi, simulando il salto con la telecamera a volo d'uccello. Chissà quanti di voi rideranno adesso leggendo che, quando avevo sei anni e vidi per la prima volta il suo predecessore, pensai che la macchina si gonfiasse come un canotto e che grazie a esso potesse superare anche gli ostacoli più incredibili.



Il gioco va avanti sempre e spetta a voi decidere quando smettere, a meno che non si perdano le vite prima; e devo ammettere che "smettere" è molto complicato. È uno di quei giochi che non invecchia mai e che una partitina durante i momenti morti della giornata ci sta sempre; meglio ancora se la musica rimane attiva (dopotutto, un'automobile senza autoradio sembra senz'anima). La giocabilità è abbastanza semplice e non richiede particolari abilità, se non magari memorizzare i punti critici del tracciato, in modo da avere i riflessi pronti per i salti improvvisi e il calcolo esatto se ci dovessero essere automobili nemiche tra noi e l'ostacolo.

Recentemente ho fatto il confronto tra Ruote Fumanti e il suo predecessore e, personalmente, questo titolo mi ha entusiasmato un po' di più, senza nulla togliere al primo; però questo ti trasporta in un modo che è una magia. Vi assicuro che non smetterete di giocare facilmente. Ora che l'estate è finita, ognuno è tornato al proprio posto di lavoro, studio, routine giornaliera e le giornate fredde sono alle porte: ecco a voi un gioco che vi terrà in ottima compagnia e che vi farà sentire sotto le mani delle vere e proprie Ruote Fumanti!

di **Daniele Brahimi**







GIUDIZIO FINALE

» Giocabilità 80%
È un piacere pilotare e ed eseguire super salti a parte lievi imperfezioni.

» Longevità 80%
Potrebbe stancare presto, oppure potrebbe durare come un diamante… Per sempre!

NEW GAME

VOID LANCER

Anno: 2026
Editore/Sviluppatore: Snow-G
Genere: Shoot 'em up
Piattaforma: Commodore 64

Uno sparatutto a scorrimento in cui la protagonista è una navicella spaziale non poteva mancare tra le nuove uscite; è anche un genere vintage che, secondo me, ricorda parecchi titoli giocati nelle sale giochi e sugli home computer, come Nemesis, Armalyte, ecc. Void Lancer si presenta molto simile ai titoli appena citati e a molti altri (forse troppi) usciti negli anni ’80 e ’90: a quell’epoca, come genere, non mi attirava, ma con il tempo li ho rivalutati, giocati e alcuni anche terminati. Questo nuovo titolo non potevo lasciarmelo sfuggire, perché è anche ben fatto anche se, ahimè, un pochino difficile.

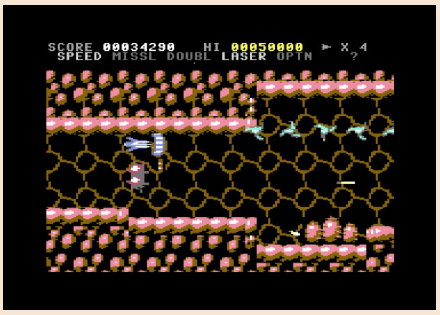
La giocabilità è abbastanza fluida e non presenta particolari problemi; inoltre la navicella verrà potenziata con spari multipli aggiuntivi, laser, velocità, ecc. Se riusciremo a distruggere l’intera ondata di nemici che si presenta sullo schermo, potremo proseguire; qui trovo doveroso darvi un consiglio sui potenziamenti: non prendete più di una volta la velocità, altrimenti andrete troppo veloci e, di conseguenza, perderete il controllo del mezzo, finendo per schiantarvi alle estremità. Ci sono sei livelli belli graficamente e alla fine di ognuno ci sarà un "boss finale" ad attenderci; dovremo trovare il suo punto debole per avere la meglio su di lui in pochi secondi. Il comparto audio ha una bella musica di accompagnamento e potrete sentire anche gli effetti sonori, seppur limitati ai colpi sparati dalla propria astronave. La longevità sarà motivata dai sei livelli, che non sono moltissimi, e dovrebbe invogliare ad arrivare fino in fondo, oltre alla curiosità di vedere come sono fatti i sei boss di fine livello, compreso l'ultimo alla fine del gioco.

Come tutti i videogame che si rispettino, basterà qualche tempo speso ad allenarci ed ecco che avremo preso la mano per poter avanzare e arrivare fino in fondo. Inizialmente è del tutto normale perdere subito in maniera assurda; dopotutto, chi ha detto che sia facile pilotare un’astronave? Senza darvi ulteriori consigli, che uno basta e avanza, provate tutti i potenziamenti nelle prime partite, in modo da trovare quello che fa più per voi, prendere confidenza e divertirvi a blastare ondate multiple di nemici il cui solo scopo è il vostro annientamento.

Se siete nostalgici dei vecchi arcade sparatutto a scorrimento, ecco il titolo giusto per voi, con una veste rinnovata ai giorni nostri ma pur sempre con il retrogusto di quegli anni… Però, con la difficoltà, il team di sviluppo poteva essere più generoso; invece l'ha mantenuta molto, molto alta! Pazienza: vuol dire che non ci annoieremo mai, affronteremo orgogliosamente grandi sfide e alla fine le nostre fatiche saranno ripagate!

di **Daniele Brahimi**







GIUDIZIO FINALE

» Giocabilità 70%
Difficile all’inizio e ci vuole un po’ per padroneggiarlo ma ben giocabile.

» Longevità 75%
La difficoltà potrebbe demotivare, ma dopo un po’ tutto sarà più fluido.

NEW GAME

WOLF

Anno: 2026
Editore/Sviluppatore: Thalamus Digital Publishing Ltd.
Genere: FPS
Piattaforma: Game Boy Color
Sito web: https://thalamusdigital.itch.io/wolf-game-boy-color

Chi ha detto che per giocare a uno sparatutto in prima persona servono un PC “muscoloso”, una scheda video da capogiro e monitor grandi come finestre? WOLF, pubblicato da Thalamus Digital per Game Boy Color, arriva a dimostrare che anche la piccola console portatile Nintendo può trasformarsi in un discreto teatro di guerra. Il progetto, realizzato da Shane McCafferty per Rocketship Park, punta tutto su un motore di ray casting personalizzato, capace di ricreare corridoi texturizzati, prospettiva e senso della profondità su un hardware nato per ben altre ambizioni. Il riferimento ai grandi FPS degli anni Novanta è evidente: l'ombra di Wolfenstein 3D aleggia tra le pareti, anche se qui la missione è decisamente più essenziale.

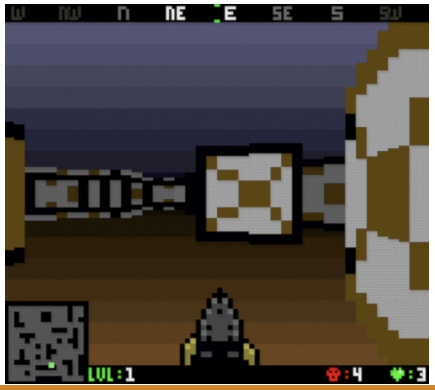
Il meccanismo è immediato e semplice. Bisogna eliminare un numero prestabilito di nemici, recuperare la chiave d'uscita e raggiungere la fine del livello prima che i nostri punti salute finiscano. I nemici pattugliano gli ambienti, ci inseguono e lanciano proiettili da evitare con un minimo di riflessi. Si parte con appena tre punti vita… insomma, niente passeggiate turistiche nei corridoi!

A rendere l'esperienza più interessante ci pensano le mappe generate proceduralmente, le diverse palette cromatiche e un radar che aiuta a orientarsi nel labirinto. Raccogliendo l'apposito potenziamento, è possibile individuare anche le posizioni degli avversari. I comandi sono adattati ai pochi pulsanti della console, con il tasto A dedicato al fuoco e il B utilizzato per le bombe.

Ma il vero protagonista è il comparto tecnico. WOLF non cerca di nascondere i limiti del Game Boy Color, li trasforma in parte integrante della sua identità, dimostrando quanto ingegno possa ancora spremere una macchina a otto bit. Naturalmente, non aspettatevi la complessità o la varietà di uno sparatutto moderno: l'impianto resta essenziale e il suo fascino dipende soprattutto dalla curiosità di vedere un FPS funzionare su questo piccolo schermo.

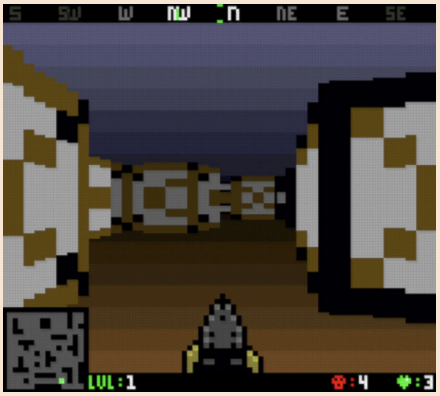
WOLF è soprattutto una dimostrazione tecnica giocabile, un omaggio alla fantasia degli sviluppatori homebrew e alla voglia di superare i limiti dell'hardware. Per chi ama il retrogaming sperimentale, vale la pena infilare la cartuccia flash e perdersi nei suoi corridoi. Il Game Boy Color non diventerà una macchina da guerra, ma sa ancora sorprendere!

di **Roberto Del Mar Pirazzini**











GIUDIZIO FINALE

» Giocabilità 90%

Essenziale! Semplice! In due pulsanti e la croce direzionale c’è tutta la magia di un FPS davvero ben strutturato.

» Longevità 90%

Non è un titolo che si porta “a casa” subito, ma richiede una certa strategia e padronanza. Vi farà sudare e godere!

AI nel minimondo dei microcomputer

A torto o a ragione l'uso dell'AI ha invaso anche la nostra passione retrocomputeristica. Non si contano più i casi di conversioni più o meno note sui nostri vecchi computer.
Non che la cosa ci dispiaccia, anzi, piuttosto la preoccupazione più grande è rivolta a quell'aspetto poco tenuto in considerazione, prevaricato dal desiderio di ottenere più titoli possibili: la conoscenza! La conoscenza di cosa l'AI ha prodotto concretamente. I sistemi automatizzati esperti (così si chiamavano una volta) producono codice senza avvisare, spiegare o dettagliare, a meno che non lo si chieda esplicitamente. Più che sviluppatori, assistiamo in questi ultimi mesi, a produzioni di esperti del 'prompt'. Badate bene, non è tutto oro quello che luccica, nel senso più deteriore del termine.
D'altro canto non è possibile definire inesperti, sviluppatori che istruiscono le AI alla concezione di programmi e routine.
Se non si sa cosa si sta facendo, a livello teorico, è impossibile indirizzare gli agenti intelligenti a formulare una qualsiasi forma di prodotto finito. Intendo con questo che l'AI da sola non basta per produrre codice che abbia un senso! Pertanto, non si può definire mero lavoro di copia/incolla, quello svolto da chi si dà il titolo di "sviluppatore". Durante gli anni '80 si gridava al miracolo se il nostro glorioso Commodore 64 produceva sintesi sonora vocale, adesso lo diamo per scontato. Sono fiducioso nel credere che c'e' ancora speranza per il popolo del joystick, in quanto l'AI costituisce un istruttore, capace, se glielo si chiede, di informare di ogni passaggio o fase di sviluppo condotto, qualsiasi sia il compito assegnato.
Per cui, a patto che si decida di continuare ad essere protagonisti, ben venga il vibe coding. Anzi, auspico che le produzioni, con l'aiuto dell'AI non si limitino a ripetere a pappagallo la lezioncina ma che costituiscano uno strumento di supporto informatico. Dovrebbe costituire un processo di accelerazione, non di mera sostituzione del programmatore.
In queste ore di fermento per l'imminente rilascio di TheAmiga1200, Battilana e soci, ci decantano le innumerevoli dotazioni di cui dispone la neonata macchina. A ben pensare, quello che si propone è unicamente una replica, utile solo a tenere in vita l'hardware che diversamente sarebbe, prima o poi, destinato all'obsolescenza. Ma noi si sa, siamo nostalgici, romantici e ci emozioniamo con poco.
Acquisteremo la macchina e giocheremo titoli che sul nostro A1200 classico, magari funzionante ancora sulle nostre scrivanie, abbiamo già consumato per parecchio tempo. Non posso che applaudire alla neonata macchina Atari. A breve assisteremo all'ennesima repro: l'Atari 800 XL (THE800XL è previsto in uscita per aprile 2027). Devo dire che acquistarlo, varrà tutta la spesa e che in fondo sono sempre soldi ben spesi, a dispetto di mogli e fidanzate che bistrattano ciò che amiamo in modo viscerale. In fondo per tenere in vita la nostra passione, alimentarla e darle nuova luce, è il modo più efficace.
Con questa consapevolezza nel cuore, vogliamo darvi appuntamento al periodo natalizio che mai come quest'anno ricalca i fasti dei Natali passati. Come nella migliore delle favole disneyane, le prossime festività, saranno da tutti noi vissute con quello spirito e quella luce negli occhi, che brillavano all'ombra delle nostre camerette, dinanzi a quel monitor o tv, che in alcuni casi era in bianco e nero ma risplendeva e si colorava con la fantasia che caratterizzava noi ragazzi di quegli anni incredibili.
Se si vuol fare un paragone: negli anni Ottanta assistevamo a timidi tentativi di democratizzare lo sviluppo dei videogiochi, rendendo accessibile anche ai semplici appassionati un'attività fino ad allora riservata prevalentemente a chi possedeva competenze di programmazione. Strumenti come Games Designer, H.U.R.G. e White Lightning consentivano di trasformare la propria fantasia in un videogioco, pur entro i limiti imposti dai programmi stessi.
A oltre quarant'anni di distanza, l'avvento dell'intelligenza artificiale e del vibe coding, ripropone quella stessa aspirazione su una scala completamente diversa. Oggi non si tratta più soltanto di modificare sprite, suoni e movimenti attraverso menu predefiniti, ma di collaborare con sistemi capaci di scrivere codice, analizzare problemi e contribuire alla realizzazione di progetti complessi. Rimane tuttavia una distinzione fondamentale: facilitare la produzione non significa necessariamente migliorare la qualità o l'originalità delle opere. La creatività, la sensibilità progettuale e la capacità di valutare il risultato continuano a essere determinanti e costituiscono doti proprie insostituibili.
Se negli anni Ottanta la sfida era permettere a chiunque di creare un videogioco senza conoscere il linguaggio macchina, oggi l'AI offre qualcosa in più: la possibilità di realizzarlo e, contemporaneamente, imparare a comprendere la tecnologia che lo rende possibile. In fondo, il desiderio è rimasto lo stesso. Sono cambiati radicalmente gli strumenti a disposizione per trasformarlo in realtà.
Se ci credi lo vedi, si diceva. Mai parole furono più azzeccate!

Filippo Occulto

Disclaimer

RetroMagazine World (fanzine aperiodica) è un progetto interamente no profit e fuori da qualsiasi circuito commerciale. Tutto il materiale contenuto è prodotto dai rispettivi autori e pubblicato grazie alla loro autorizzazione.

RetroMagazine World viene concessa al pubblico con licenza: Attribuzione - Non commerciale - Condividi allo stesso modo 4.0 Internazionale (CC BY-NC-SA 4.0 INT) https://creativecommons.org/licenses/by-nc-sa/4.0/deed.it

In pratica sei libero di: condividere, riprodurre, distribuire, comunicare o esporre in pubblico, rappresentare, eseguire e recitare questo materiale con qualsiasi mezzo e formato, modificare, rielaborare, trasformare il contenuto e basarti su di esso per altre opere, alle seguenti condizioni:

Attribuzione
Devi riconoscere una menzione di paternità adeguata, fornire un link alla licenza e indicare se sono state effettuate delle modifiche. Puoi farlo in qualsiasi maniera ragionevole possibile, ma non con modalità tali da suggerire che il licenziante avalli te o l'utilizzo del materiale da parte tua.

NonCommerciale
Non puoi utilizzare il materiale per scopi commerciali.

StessaLicenza
Se rielabori, trasformi il materiale o ti basi su di esso, devi distribuire i tuoi contributi con la stessa licenza del materiale originario.

Il licenziante non può revocare questi diritti fintanto che tu rispetti i termini della licenza.

Divieto di restrizioni aggiuntive
Non puoi applicare termini legali o misure tecnologiche che impongano ad altri soggetti dei vincoli giuridici su quanto la licenza consente loro di fare.

AI Disclaimer *- Ove non esplicitamente dichiarato dagli autori, collaboratori ed editore RMW, nessun contenuto di questa rivista digitale (editoriali, articoli, recensioni, testi o immagini) è stato prodotto o generato da strumenti AI.*

ISSN: 3103-697X - RetroMagazine World [Online]
DOI: 10.5281/zenodo.23510697

© 2017-2026 RetroMagazine World - All rights are reserved.

Publisher: Associazione Culturale RetroMagazine World
ID: IT-93031480580 - Registered office and HQ: Rome, Italy
retromagazineworld@gmail.com - https://www.retromagazine.net

RetroMagazine World
Anno 9 - Numero 55 - OTTOBRE 2026

Direttore Responsabile
Francesco Fiorentini
Vice Direttore
Marco Pistorio
Coordinatore Redazione/Editing
David La Monaca
Area Web/Social Media Manager
Giorgio Balestrieri / Carlo Del Mar Pirazzini