



WORLD **RetroMagazine**

future days are back

N° 53

DOI 10.5281/zenodo.200754184



CHRONO RUNNER

CAPOLAVORO su MSX



- **Alla scoperta del Salora Fellow**
- **Waffle CopyPRO Universal**
- **Super Turtle Pilot - Storia di un avventuroso recupero**
- **PiStudio: il nuovo ambiente di sviluppo su CPC**
- **Tutti i modi grafici non-standard del VIC-II**

Con il retrocomputing non si sbaglia mai...

Il retrocomputing, lo sapete bene, non è mai stato un semplice guardarsi indietro con nostalgia. È un viaggio che ogni giorno ci regala scoperte inaspettate, come se il tempo, invece di scorrere in linea retta, si arrotondasse su se stesso lasciando dietro ogni angolo un nuovo tesoro. E proprio quando pensiamo di aver visto tutto – quel computer, quel chip, quella periferica introvabile – ecco saltare fuori un manuale dimenticato in una soffitta, una cassetta con un software mai pubblicato, un libro di programmazione che riscrive le regole di quello che credevamo di sapere. Con il retrocomputing non si sbaglia mai perché lo scavo archeologico non finisce: c'è sempre qualcosa da recuperare, e la sorpresa è parte del gioco.

Ma la vera svolta degli ultimi anni è un'altra. Mentre fuori si discute se i giochi di oggi siano "migliori" o "peggiori" di quelli di trent'anni fa, dentro la nostra nicchia succede qualcosa di meraviglioso: decine di nuovi titoli, sviluppati da appassionati per piattaforme vintage, superano spesso e volentieri la media della produzione originale degli anni '80 e '90. Non è eresia, è un fatto. I nuovi giochi per Commodore 64, Amiga, ZX Spectrum o Amstrad CPC sfruttano decenni di esperienza, tecniche riscoperte, trucchi che all'epoca nessuno aveva nemmeno immaginato. Hanno il sapore autentico del periodo, ma la maturità di chi ha studiato ogni singolo difetto dell'hardware e lo ha trasformato in un pregio.

Il terzo pilastro, forse il più affascinante, è l'hardware. Non più solo riparare, ma reinventare. Una marea di creativi, ingegneri per passione, hobbisti con il saldatore facile stanno costruendo un ponte solido tra ieri e oggi: schede acceleratrici, lettori di floppy alternativi, dispositivi Wi-Fi per macchine che non avevano nemmeno un modem, cartridge moderne per console d'epoca. È un ponte che non guarda indietro, ma proietta il retrocomputing nel futuro, rendendo queste macchine non solo vive, ma aggiornate, connesse, capaci di dialogare con il mondo contemporaneo senza perdere un grammo della loro anima.

Il numero che avete tra le mani è l'esempio perfetto di questo viaggio. Dentro troverete proprio questo mix esplosivo: il racconto di una scoperta recente – un software ritenuto perduto e ora tornato alla luce –, le recensioni di nuovi giochi che avrebbero fatto impallidire i produttori degli anni d'oro, e la presentazione di hardware che trasforma il vostro vecchio computer in una macchina moderna. E, come sempre, alla fine resterete con l'acquolina in bocca, chiedendovi cosa succederà nei prossimi mesi e nei prossimi anni.

Perché con il retrocomputing non si sbaglia mai. E la prossima sorpresa è probabilmente già dietro l'angolo.

Francesco Fiorentini

SOMMARIO	PAG.
♦ PiStudio: il nuovo ambiente di sviluppo su CPC	Pag. 3
♦ Salora Fellow	Pag. 6
♦ Sinclair QL: il salto quantico di Sir Clive	Pag. 10
♦ NES Zapper	Pag. 14
♦ WaffleCopyPro-Universal	Pag. 16
♦ Scopriamo Kyvos di George Sokianos	Pag. 18
♦ Caleidoscopio in varie salse (per C64)	Pag. 22
♦ Un gioco di prestigio (C64 principianti)	Pag. 30
♦ Super Turtle Pilot (storia di un recupero)	Pag. 34
♦ Tutti i modi grafici non-standard del VIC-II	Pag. 38
♦ Costruzione di mappe di gioco (C64)	Pag. 46
♦ Preservazione digitale e miglioramento del codice	Pag. 52
♦ AmigaOS 3.3 - L'evoluzione del S.O.	Pag. 58
♦ Intervista a Richard Farrell aka Vyper68	Pag. 60
♦ Collezionare o accumulare?	Pag. 66
♦ 50 anni di Apple: mezzo secolo di Think Different	Pag. 68
♦ Le mascotte (quasi) dimenticate di Nintendo	Pag. 72
♦ AmiGame Jam 2025 - Ritorno all'arcade	Pag. 74
♦ Andoria: il ritorno del GDR su Apple II	Pag. 77
♦ Crystian and The Lost Crystals (C64)	Pag. 79
♦ Ghosts'N'Goblins (Amiga AGA)	Pag. 82
♦ Mappy (Amiga AGA)	Pag. 84
♦ Akalabeth 1000 (Sega SG-1000)	Pag. 86
♦ Wee Ninja (Atari 5200)	Pag. 88
♦ Space Bash: Contact!! (MSX 16KB)	Pag. 90
♦ Celeste (Amiga)	Pag. 92
♦ Randoom (Commodore Plus/4)	Pag. 94
♦ Chrono Runner (MSX)	Pag. 96
♦ Black Castle II (Gameboy)	Pag. 98
♦ Echodrome (Amiga)	Pag. 100
♦ Dragon View (SNES)	Pag. 103
♦ Escape From PETSCII Planet (C64)	Pag. 106
♦ Junk Runner 64 (N64)	Pag. 108
♦ Soul Blaze (Dreamcast)	Pag. 110
♦ Zezito and The Lost Stars (Acorn Electron 32K)	Pag. 112
♦ Burgertom (Atari Jaguar)	Pag. 114
♦ Tony Do It! (C64, Atari 8-bit)	Pag. 116
♦ Wings Of Fear (Amiga AGA)	Pag. 118
♦ Bubble Buddies (C64)	Pag. 121
♦ Miracle Boy in Dragon Land (Atari ST)	Pag. 122
♦ Curse of The Pharaoh (C64)	Pag. 125
♦ Axis (C64 - Plus/4)	Pag. 126
♦ Gravy Duck (C64)	Pag. 128

Hanno collaborato:

- | | |
|---------------------------|-----------------------------|
| • Francesco Fiorentini | • Chiara Carrara |
| • Carlo Del Mar Pirazzini | • Alex Fridman |
| • Marco Bergomi | • Daniele Brahimi |
| • Eugenio Rapella | • David La Monaca |
| • Jari Jalonen | • Stefano Tognon |
| • Takahiro Yoshihoka | • Roberto Del Mar Pirazzini |
| • Roberto Lanciotti | • Tommaso Pinturru |
| • Ingrid Poggiali | • Giampaolo Moraschi |

- Immagine di copertina:
Giuseppe Mangini
- Layout di copertina:
Carlo Del Mar Pirazzini



PiStudio: il nuovo ambiente di sviluppo su CPC

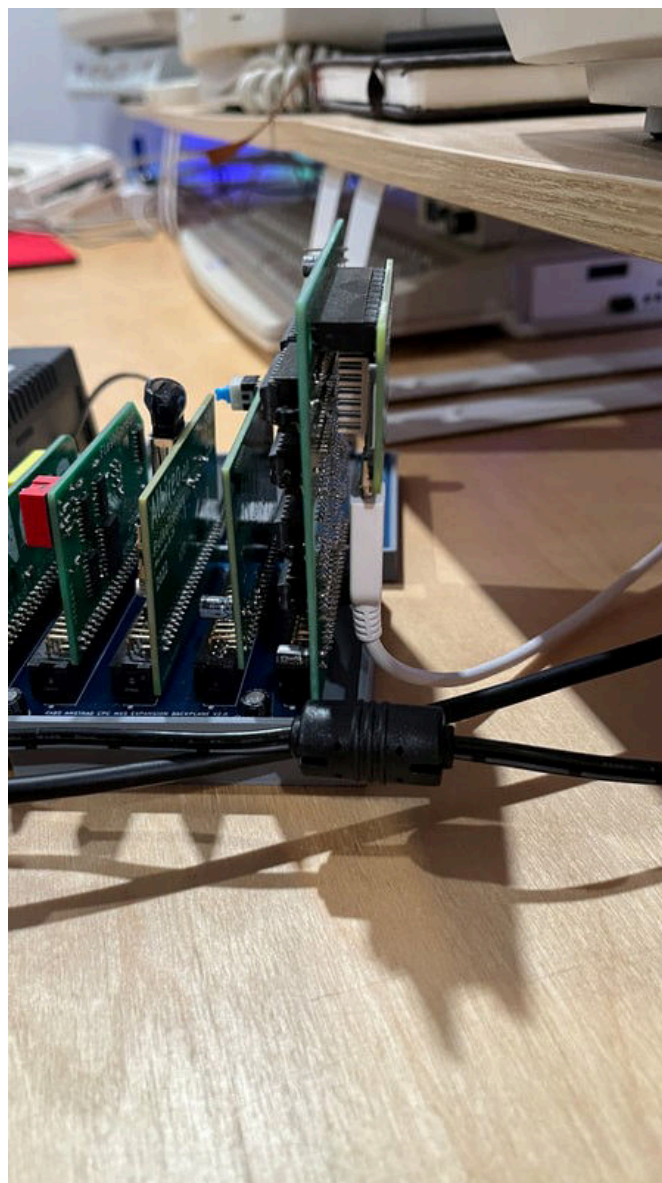
di Carlo Nithaiah Del Mar Pirazzini

PiStudio è un ambiente di sviluppo completo che gira su Raspberry Pi e si collega a un Amstrad CPC tramite l'interfaccia CPC-CPLink di Revaldinho. Dal punto di vista dell'utente, però, "tutto" avviene sul CPC: tastiera, schermo (overscan Mode 2 a 92×32 caratteri), input e output sono quelli originali, mentre la Pi fa da coprocessore silenzioso che esegue compilazioni, debug, I/O avanzato, rete e AI. L'IDE è pensato per funzionare completamente offline, con alcune funzioni opzionali (chatbot, Git remoto, rete) che sfruttano la connettività quando disponibile.

L'architettura ruota attorno a una suite integrata di strumenti che condividono clipboard, interfaccia e persino apertura documenti: puoi passare dall'editor al debugger,



Esempio di "aggiornamento hardware" con il PiStudio



La scheda con il Raspberry

al file manager o al viewer Git senza "uscire" dal contesto CPC. Il client lato CPC è un programma in Z80 (disponibile come ROM o su disco) che imposta l'overscan, legge la tastiera, invia gli eventi alla Pi, riceve solo i "diff" di caratteri modificati per aggiornare rapidamente lo schermo e gestisce breakpoint, dump RAM, accesso ai drive AMSDOS/UniDOS e riconnessione in caso di problemi col link.

Cosa c'è dentro PiStudio?

Il cuore di PiStudio è un editor di codice full-featured, pensato per sviluppo ma valido per qualunque testo. Tra le funzioni principali possiamo elencare:

Più documenti aperti con navigazione tipo tab





Split-view orizzontale/verticale per confronto o doppio editing.

Find & Replace nel file corrente o su tutto il progetto, fuzzy file finder, segnalibri.

Compilazione con scorciatoia: il binario viene streammato direttamente nella RAM del CPC ed eseguito.

Supporto a Locomotive BASIC, assembler RASM e C via SDCC, con Code Inspector per funzioni/variabili in ASM, BASIC e C.

Undo/redo illimitati, cronologia clipboard, calcolatrice con conversione esadecimale/decimale, tabella ASCII/glyph CPC.

Spell checking multi-lingua, auto-formatting ASM, word wrap e tab configurabili, supporto stampa, persistenza completa della sessione (file aperti, cursori, split).

In più c'è un tool di traduzione via IA integrato direttamente nell'editor, utile per commenti, documentazione e testi dentro i progetti.

Debugger su hardware reale

Il debugger permette ispezione e stepping in tempo reale di programmi che girano sul CPC vero. Quando il codice incontra una BRK, il client CPC cattura lo stato completo

dello Z80, parte dello stato Gate Array (ROM, banking RAM) e un dump dei 64 KB di RAM, inviando tutto a PiStudio.

Hex editor e file manager

L'hex editor è pensato per lavorare su file binari, immagini disco o dump memoria con vista Hex+ASCII affiancata, insert/overwrite, selezioni con lunghezza e checksum live, cut/copy/paste, import/export blocchi, ricerca/replace testo ed hex, goto offset, split-view e undo/redo, con stampa.

Il file manager è un browser dual-pane che unifica lo storage della Pi (PI:), drive fisici A:/B: del CPC, eventuali istanze PiStudio remote in LAN (via hostname).

Offre pieno supporto UniDOS (long filename, subdirectory, storage moderni), creazione cartelle, rename, copia/sposta/elimina, selezione per pattern, toggle file nascosti, colonne configurabili e sort per vari criteri, bookmarks, fuzzy finder e apertura diretta di file in editor di testo o hex. Gestisce anche creazione/navigazione di immagini DSK e archivi ZIP, e lettura degli header AMSDOS, più una shell interna.

Il Debugger al lavoro





Git, chatbot e enciclopedia offline

PiStudio include un frontend Git completo, pensato per workflow da tastiera. Questo permette la creazione/clone repo, apertura recenti, la vista cambi nella working tree, lo storico commit, diff side-by-side e per singola linea, stage/unstage, commit, revert, reset a revisioni specifiche e push/pull remoto. Il chatbot/assistente di codice integra provider OpenAI (ChatGPT) e Anthropic (Claude), con gestione conversazioni (storico salvabile), configurazione modelli/API key, split-view. Può analizzare codice, migliorare algoritmi, generare commenti, suggerire rename di label, tradurre documenti, generare file nuovi, il tutto senza uscire dall'IDE; è richiamabile anche come dialog tool dall'editor. La Developer Guide è un'enorme enciclopedia offline (~3 MB di testo) con articoli navigabili via hyperlink e ricerca full-text, che copre un po' di tutto. Troviamo davvero una marea di roba: introduzione a BASIC/ASM/C, storia e cultura CPC, manuali software e linguaggi, reference completo Locomotive BASIC, C con SDCC, hardware e firmware, programmazione Z80, algoritmi, formati file, grafica, suono, game dev, demomaking, tips & tricks e raccolte di sorgenti. Insomma un gioiello di preservazione! Dalle impostazioni globali puoi configurare

Wi-Fi, stampante di rete, API key per il chatbot, identità Git, tema colori e preferenze di display, adattando l'IDE sia all'uso in puro offline sia a un contesto più connesso.

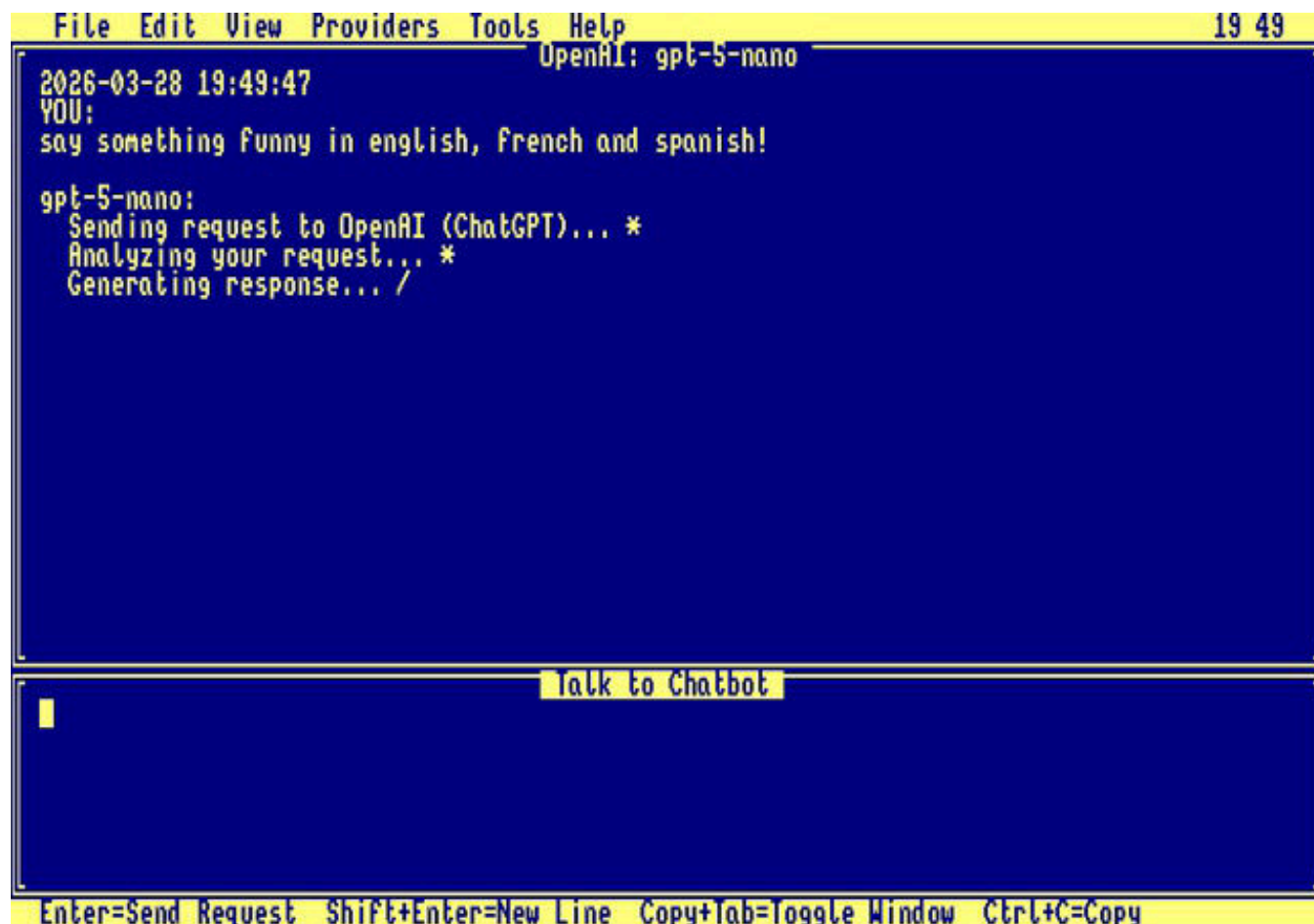
Quando arriverà il progetto??

PiStudio è al momento annunciato ma non ancora distribuito pubblicamente. L'autore (NoRecess) vuole prima completare un proprio progetto reale usando l'IDE, in modo da rifinire UX e stabilità. La distribuzione avverrà come immagine disco per SD Raspberry Pi, così da ridurre al minimo la complessità d'installazione per l'utente finale.

In sintesi, PiStudio prende un Amstrad CPC del 1984 e, grazie a una Raspberry Pi, lo trasforma in una vera workstation di sviluppo con editor moderno, debug su ferro, gestione file/Git, AI e documentazione estesa, il tutto restando "dentro" schermo e tastiera originali.

Sito web del progetto: <https://norecess464.weebly.com/news/pistudio-a-raspberry-pi-powered-ide-for-the-amstrad-cpc>

Ringraziamo l'autore del PiStudio per averci concesso l'utilizzo delle foto.



Chatbot





Salora Fellow

di Jari Jalonen

Il contesto di Salora

C'è stato un periodo in cui Salora era un importante attore industriale in Finlandia. Per decenni, l'azienda produsse varie radio, poi televisori e un'ampia gamma di altri dispositivi elettronici. Successivamente, Nokia acquistò Salora e iniziò a produrre telefoni cellulari sfruttando le competenze tecniche di Salora – ma questa è un'altra storia. Anni prima dell'era dei telefoni cellulari, Salora intendeva lanciare sul mercato un computer a marchio Salora. Il primo di questi computer fu il Salora Fellow, basato sulla tecnologia del VTech Laser 200.

In pratica, una semplice etichetta "Salora Fellow" veniva applicata sopra il marchio originale. Lo stesso valeva per il software: un adesivo del Salora Fellow veniva incollato sopra la scritta VTech 200 sulle confezioni dei giochi. Purtroppo non ho informazioni sul motivo per cui scelsero proprio il VTech 200, ma così è stato.

Specifiche tecniche

In Finlandia, il Salora Fellow, insieme al computer Aquarius, era tra i dispositivi più economici disponibili, e la sua

capacità era minima. Aveva 4 KB di RAM, di cui circa 2 KB utilizzabili per la grafica. Era disponibile un'espansione di memoria da 16 KB, praticamente indispensabile. Il dispositivo monta un processore Zilog Z80A, un chip comune in molti computer a 8 bit degli anni '80, come il Sinclair Spectrum o l'Amstrad CPC.

Salora avrebbe potuto includere quei 16 KB di memoria internamente fin dall'inizio, ma ciò probabilmente ne avrebbe aumentato il prezzo. Fu più strategico vendere l'unità a 1.000 marchi finlandesi – che, aggiustati per l'inflazione, corrispondono a circa 454 euro odierni. Per confronto, un Commodore 64 all'epoca poteva costare facilmente 2.000 marchi (circa 900 euro oggi). Inoltre, l'espansione di memoria doveva essere acquistata separatamente se l'utente voleva giocare alla maggior parte dei giochi. La risoluzione del display è di 32x16 caratteri in modalità testo con 8 colori, mentre la modalità grafica offre una risoluzione di 128x64 con 4 colori.

L'audio è prodotto da un altoparlante interno al computer. Inizialmente ho cercato sul lato del mio dispositivo dei



Fig. 1 - Il Salora Fellow





Fig. 2 - Il setup del mio Salora Fellow: il computer, il Datassette del Salora Fellow, due joystick, l'espansione di memoria da 16k e quel cavo a Y (cavo da stereo a doppio mono). Ora fa parte della mia famiglia. Quella cassetta blu non fa parte del Salora Fellow, fa parte della mia collezione MSX (SVI7x8). Penso che tutti sappiate cosa significa quando si è collezionisti, vero?

connettori RCA per l'uscita audio, aspettandomi due o tre porte. Alla fine ho capito che c'è una sola porta RCA, ed è esclusivamente per il segnale video composito (giallo). Poiché l'audio proviene dall'altoparlante interno, non può essere inviato agli altoparlanti della televisione. Per una volta, è meravigliosamente semplice.

Una manciata di giochi, una stampante, un joystick, espansioni di memoria e un registratore a cassette furono rilasciati con il marchio Salora Fellow. Inoltre, tutte le periferiche progettate per il VTech 200 sono completamente compatibili. Ho visto video in cui qualcuno usa un computer Salora Fellow con un'espansione di memoria del VTech 200.

Il mio Salora Fellow

Sono nato nel 1975, quindi ricordo che quando ero bambino si parlava del Salora Fellow. Ricordo di aver visto il dispositivo in una vetrina espositiva in un negozio chiamato Musta Pörssi a Forssa. All'epoca veniva percepito come molto primitivo e "economico" in senso negativo. Allora non mi interessava affatto; il Commodore 64 era ciò che catturava la mia attenzione. Il mio interesse per il Salora Fellow si è acceso solo negli anni 2020, principalmente perché questi dispositivi sono diventati

piuttosto rari.

Ho comprato la mia unità a cavallo del millennio in un mercatino dell'usato a Forssa per 200 marchi (circa 52 euro oggi). Veniva con alcuni giochi, un'espansione di memoria e un registratore a cassette. Ho quasi completamente dimenticato il dispositivo fino al 2023, quando sono rimasto disoccupato. Vivo in campagna, in un posto chiamato Ypäjä, precisamente nel villaggio di Ypäjänkylä. Ho iniziato a esplorare la mia soffitta e ho trovato diversi Commodore 64, un paio di VIC-20, uno Spectravideo 328 e, con mia grande sorpresa, il Salora Fellow.

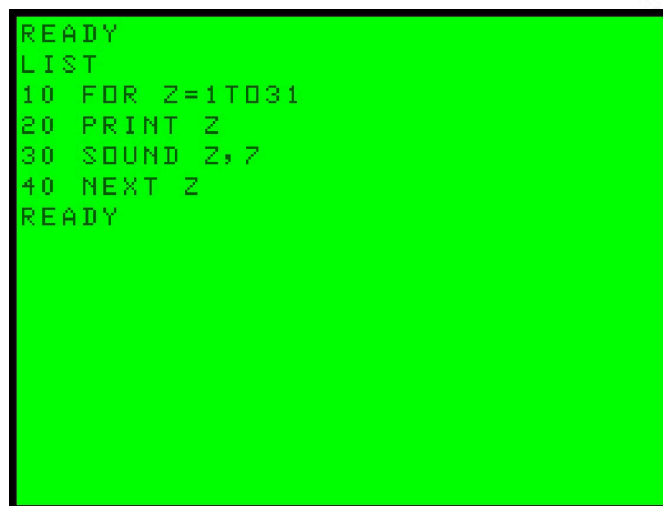


Fig. 3 - Esempio di generazione del suono in VZ Basic





Il dispositivo aveva resistito al clima finlandese – da inverni a -30°C a estati a $+30^{\circ}\text{C}$ – in soffitta per oltre 20 anni. Inizialmente non si avviava. Su eBay era in vendita un VZ-200 e ho visto lì un alimentatore molto familiare; così ho avuto l'idea di tornare in soffitta e, dopo aver cercato, ho trovato un alimentatore simile. Era quello giusto: la luce si è accesa e avevo un Salora Fellow funzionante! Alla fine ho trovato online un registratore a cassette funzionante per 10 euro. Ho collegato il registratore al computer usando un normale cavo TRS da 3,5 mm (minijack).

Il dispositivo stesso funziona magnificamente. Inizialmente i tasti avevano qualche problema di contatto, dato che la macchina non era stata usata per oltre 23 anni. I tasti sono in gomma, tipo chiclet, simili a quelli dello Spectravideo 318 o del Sinclair ZX Spectrum. La sensazione è la stessa, ma la tecnologia sembra superiore: mentre lo Spectrum è famoso per i guasti alle membrane della tastiera, i tasti del Fellow funzionano ancora perfettamente.

L'audio è un suono molto stretto e semplice, mono, dall'altoparlante interno. Ci sono 31 livelli di altezza (pitch) e la programmazione segue il formato SOUND 15, 9, dove

15 è l'altezza e 9 la durata. In termini di operatività, la tastiera funziona come quella dello Spectrum; si possono usare combinazioni di tasti (scorciatoie) per velocizzare la programmazione – ad esempio, CTRL+P produce rapidamente il comando PRINT.

Sono un programmatore di natura. La rivista locale Mikrobitti pubblicava listati di programmi per il Fellow, e avevo un forte desiderio di capire l'unico linguaggio VZ Basic. Un particolare dettaglio tecnico causò inizialmente dei problemi: la porta del computer è un jack stereo da 3,5 mm (TRS), ma il registratore a cassette ha porte mono separate EAR (In) e MIC (Out). Un canale del cavo stereo trasporta i dati di caricamento, l'altro i dati di salvataggio. Alla fine ho trovato il cavo originale a Y, da stereo a doppio mono, dove l'estremità stereo si divide in due minijack mono (rosso per Mic/Salvataggio e nero per Ear/Caricamento). Finalmente potevo salvare il mio codice!

Esperimenti e soluzioni moderne

Ho trovato un sito web (vz200.org) con molti giochi per il VTech 200. Ho provato a trasferire i giochi dal PC a una cassetta fisica usando il cavo originale, ma non ha

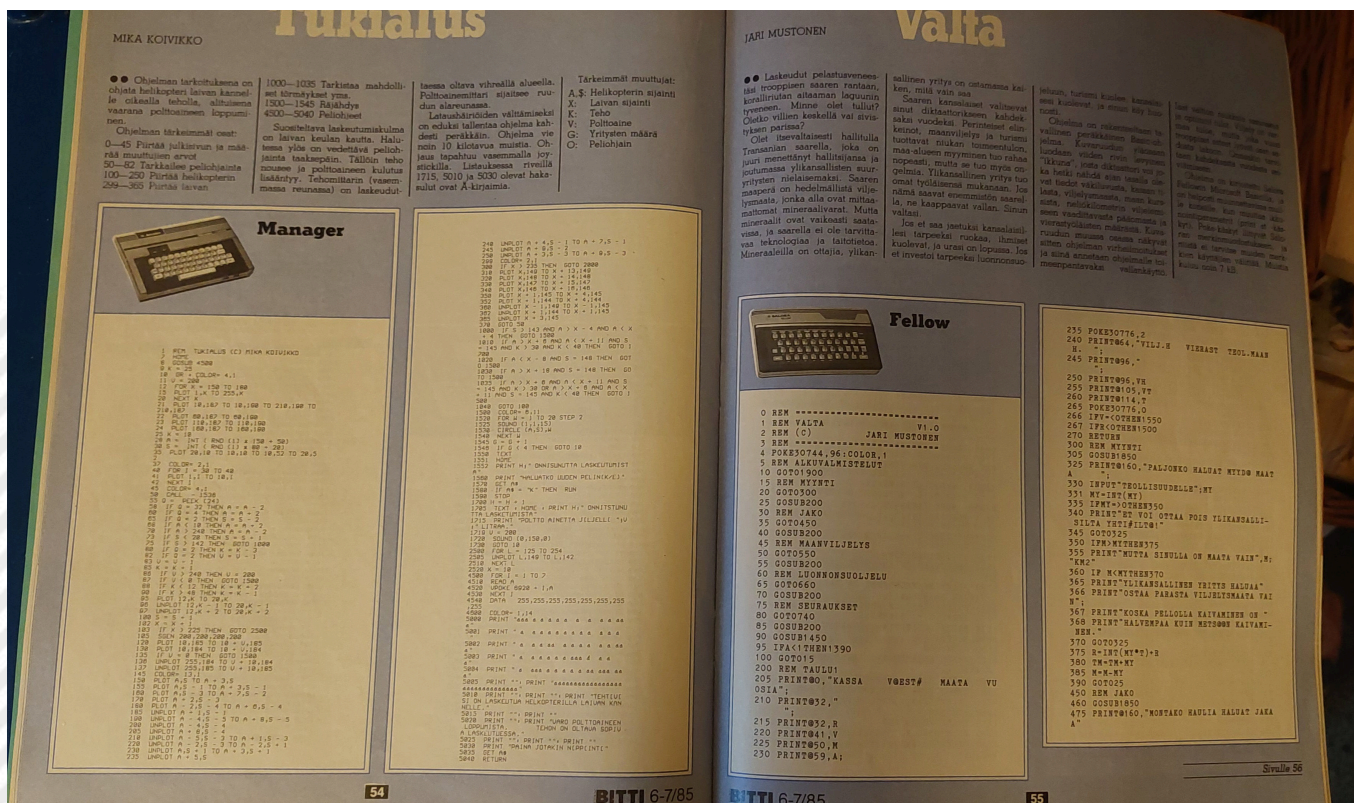


Fig. 4 - Esempio di listati sulla rivista Mikrobitti



Fig. 5 - Espansione di memoria a 16 KB

funzionato. Il computer riconosceva il nome del file, ma i giochi non si avviavano – probabilmente a causa di interferenze del segnale o diafonia stereo dovuta a modem o telefoni.

Un secondo approccio ha funzionato molto meglio: ho bypassato la cassetta e ho collegato direttamente il jack delle cuffie del PC alla porta cassette del Salora Fellow. Ho digitato CLOAD sul Fellow e ho riprodotto un file WAV del gioco Hamburger Sam dal mio PC. Quando il titolo del gioco è apparso sul mio televisore portatile Finlux da 14 pollici alle 3 di notte, ho emesso un forte grido di gioia!

Spero che la mia piccola avventura vi sia piaciuta. Mi sono concentrato sugli aspetti tecnici e sulla mia esperienza personale. Il VZ Basic è un dialetto affascinante da studiare, soprattutto per come i suoi comandi audio differiscano da GW-BASIC o SV-BASIC.

È un hobby di nicchia, ma per un appassionato di

retrocomputing è un viaggio che vale sicuramente la pena di intraprendere.

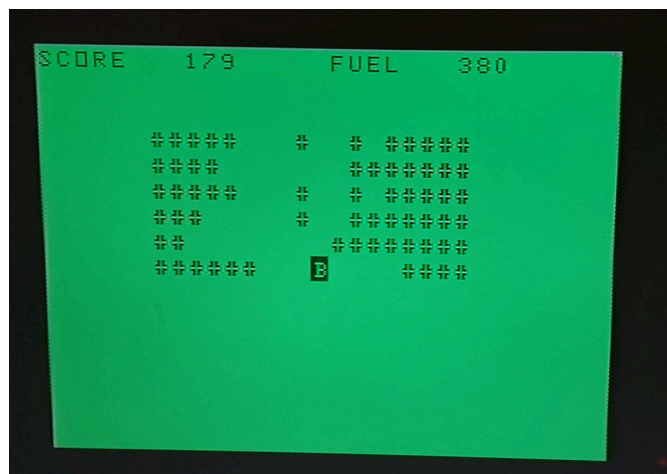


Fig. 6 - Esempio di un gioco

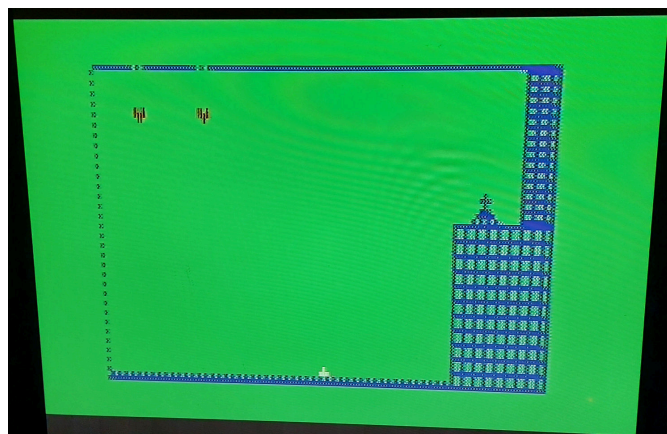


Fig. 7 - Gioco 'Ufoja': lo si potrebbe chiamare 'UFO'. C'è un piccolo punto bianco e si deve sparare agli UFO che cadono dal cielo. Il codice di 'Ufoja' proviene dalla rivista Mikrobitti



Fig. 8 - Peli Loppu, in italiano non lo traduciamo, diciamo semplicemente Game Over. :-)





Sinclair QL: il salto quantico di Sir Clive

di Tommaso Pitturru

Nel 1984, mentre lo ZX Spectrum combatteva coraggiosamente contro il ben più carrozzato Commodore 64, in un mercato degli home computer saldamente ancorato alla tecnologia a otto bit, Sir Clive Sinclair decise di alzare drasticamente il livello dello scontro. Il Sinclair QL (Quantum Leap) non nasceva come un semplice successore — rinunciando peraltro a ogni compatibilità con il passato — ma come un vero e proprio "salto quantico": una workstation professionale proposta a prezzi da home computer, capace di traghettare la potenza dei 32 bit sulla scrivania di ogni utente.

Il QL nacque sotto l'egida di una precisa, quasi ostinata e perfino un po' snob visione filosofica di Sir Clive, che considerava quello dello ZX Spectrum un successo raggiunto per il motivo sbagliato: un sistema concepito per l'alfabetizzazione informatica e la ricerca di base, dotato di un buon processore — lo Zilog Z80, sulla carta ben più prestante del MOS 6510 utilizzato dalla rivale Commodore per il C64 — e di un ottimo interprete BASIC, finito invece nelle mani dei ragazzini che l'hanno utilizzato

alla stregua di una versatile console da gioco. Sir Clive provava una certa ritrosia verso l'industria videoludica, che considerava intellettualmente inferiore rispetto ai settori business, educational e a quello della ricerca. Il QL doveva dunque essere uno strumento che nessuno avrebbe potuto considerare un giocattolo evoluto: un computer professionale e accessibile, dove il divertimento non era contemplato se non come sottoprodotto marginale.

Tali ambizioni si scontrarono purtroppo con scelte progettuali figlie di una filosofia orientata al risparmio estremo. Caratterizzato da un elegante e a suo modo avveniristico design monoblocco, curato da **Rick Dickinson**, il QL aveva un cuore pulsante costituito dalla CPU Motorola 68008 a 7,5 MHz, un processore dall'architettura interna a 32 bit ma castrato da un bus dati a soli 8 bit. Questa architettura ibrida, messa a punto dal capo progettista **David Karlin**, unita all'uso di due chip ULA (ZX8301 e ZX8302) progettati in fretta e furia e afflitti da cronici problemi di surriscaldamento, vanificava gran parte della potenza teorica del sistema. Ironia della sorte, il motivo





principale per cui fu scelto un processore "di compromesso" venne meno poco dopo la frettolosa messa in commercio del QL, quando il ben più potente Motorola 68000 subì un drastico taglio di prezzo, finendo per equipaggiare la quasi totalità degli home computer a 16 bit.

Ma torniamo al QL: persino la memoria era gestita in modo complesso. Con i suoi 128 KB di RAM di base, di cui una parte significativa era occupata dal video, la gestione dei software multitasking del QDOS — il sistema operativo scritto da Tony Tebbs — era qualcosa di possibile sulla carta, ma difficilmente realizzabile nella realtà, nonostante l'eccellente implementazione del SuperBASIC curata da Jan Jones, a detta di molti uno dei migliori interpreti BASIC mai realizzati.

Un aspetto spesso sottovalutato, ma tecnicamente sbalorditivo per l'epoca, era la capacità nativa del QL di operare in rete. Attraverso due semplici prese jack da 3,5 mm, il sistema implementava il **QLAN**, una rete locale capace di collegare fino a 64 computer. Questa intuizione permetteva la condivisione di periferiche e file tra diverse postazioni, anticipando di anni le moderne architetture d'ufficio. Tuttavia, anche in questo caso, l'implementazione soffriva della scarsa robustezza dell'hardware: il protocollo

era gestito direttamente dalla ULA ZX8302 con un sistema di "bit-banging" che caricava pesantemente la CPU e rendeva il segnale estremamente sensibile alle interferenze elettromagnetiche, limitando la velocità teorica di 100 kbps a valori molto più modesti in contesti reali.

La scelta più controversa fu l'adozione dei Microdrive come supporto di memoria standard. Concettualmente affascinanti ed esteticamente futuristici, i Microdrive erano essenzialmente degli anelli di nastro magnetico (derivato dal video tape) che ruotavano ad alta velocità. Teoricamente, il sistema avrebbe potuto raggiungere standard di affidabilità superiori attraverso un controllo più raffinato della velocità del motore e l'uso di nastri con una lubrificazione superiore per ridurre l'usura fisica e lo stiramento, causa principale della perdita di dati. Anche l'adozione di un firmware più evoluto avrebbe potuto implementare sistemi di correzione d'errore (ECC) più robusti e una gestione dei settori che minimizzasse i tempi di latenza durante la rotazione.

Nonostante queste possibili migliorie, il Microdrive non avrebbe mai potuto competere con il floppy disk: l'accesso ai dati era di tipo sequenziale, laddove il floppy disk, grazie alla testina a spostamento radiale, garantiva un accesso





diretto quasi istantaneo. La fragilità intrinseca di un nastro che scorre a contatto con le testine rendeva il supporto suscettibile a un degradamento costante, caratteristica inaccettabile per una macchina business.

Loki, il "SuperSpectrum"

Parallelamente alle difficoltà del QL, nei laboratori di Cambridge prese vita il progetto **Loki**. Concepito come un "SuperSpectrum", il Loki originale era una macchina estrema, pensata da ingegneri del calibro di **John Mathieson** e **Martin Brennan** per annientare la concorrenza con un hardware incredibilmente avanzato. A differenza del QL, il Loki si basava su una versione potenziata dello Z80 (lo **Z80H** con clock a 7 o 8 MHz), ma la vera forza risiedeva nel suo chipset proprietario:

Il Chip Loki (Super ULA): Questo integrato custom era destinato a gestire una risoluzione di 512 x 256 pixel con una palette di 256 colori simultanei. Il chip includeva funzioni di gestione della memoria e accelerazione grafica (una sorta di blitter ante litteram) che avrebbero permesso scorrimenti e animazioni particolarmente fluide, superando i limiti di "color clash" dello Spectrum originale. Il progetto prevedeva ben **128 KB di RAM dedicata esclusivamente al video**, una dotazione identica a quella delle macchine

basate sullo standard **MSX2** presentate poco tempo dopo in Giappone.

Audio Wavetable: Il comparto sonoro sarebbe stato rivoluzionario per l'epoca, con 8 canali audio a 8 bit gestiti via DMA, capaci di riprodurre campionamenti e sintesi complessa, caratteristiche che avrebbero posto il Loki su un piano superiore rispetto al chip Paula di Amiga.

L'idea era di mantenere lo stile compatto e monoblocco, ma con una potenza di calcolo grafica e sonora senza precedenti.

L'evoluzione mancata: Il QL Plus e l'eredità Amstrad

Sulla scia del lavoro svolto per il Loki, si sarebbe potuta delineare una seconda strada, più pragmatica: una fusione tra l'anima professionale del QL e l'eredità ludica dello Spectrum. Questa macchina, un ipotetico QL Plus, avrebbe mantenuto lo chassis monoblocco nero opaco ma avrebbe corretto le storture del 1984 attraverso soluzioni tecniche e software radicali:

Architettura a Doppia CPU: L'integrazione, accanto al Motorola 68000 a 8 MHz con bus a 16 bit, di un coprocessore **Zilog Z80** dedicato per garantire compatibilità hardware nativa con lo ZX Spectrum da una parte e il sistema CP/





M dall'altra. Lo Z80 avrebbe operato in modalità "bridge", accedendo a una porzione della RAM principale, rendendo il QL Plus lo strumento definitivo per il passaggio dall'home computing al personal computing professionale.

L'Interfaccia ICE: Il software di sistema avrebbe finalmente abbandonato l'austera riga di comando per abbracciare l'implementazione dell'ambiente **ICE (Icons, Cursors, Events)**. Questo framework grafico, concepito per dare al QL un'esperienza d'uso simile a quella del Macintosh, avrebbe trasformato il QDOS in un ambiente WIMP maturo. Grazie a ICE, l'interazione con le potenti applicazioni professionali create dalla **Psion** di **David Potter** sarebbe diventata intuitiva, sfruttando puntatori e icone per gestire quel multitasking che il QL possedeva già nel proprio DNA.

Storage Evoluto: L'integrazione di un drive da 3,5 pollici direttamente nello chassis monoblocco, sul lato destro della tastiera, avrebbe eliminato i Microdrive a favore di uno standard finalmente affidabile.

Il Tocco di Amstrad: Qualora Alan Sugar, dopo l'acquisizione dell'aprile 1986, avesse deciso di non soffocare la ricerca Sinclair ma di adattarla alla sua filosofia commerciale, probabilmente avremmo assistito alla nascita di un "Amstrad Sinclair QL 16". Una macchina monoblocco robusta, con il drive frontale e porte standard, pronta a combattere ad armi pari contro l'Atari ST.

La fine di un sogno

Come per altri "What if" di cui abbiamo già parlato, la realtà fu molto più prosaica. La dirigenza di Amstrad scelse la via del profitto immediato e cancellò ogni progetto sperimentale. Il Loki originale finì in un cassetto e la tecnologia sviluppata dai suoi ingegneri prese strade diverse: John Mathieson, Martin Brennan e Ben Cheese, le menti pensanti dietro il progetto Loki, fondarono la **Flare Technology**, alimentando in seguito i progetti del Flare One e l'architettura dell'Atari Jaguar.

Da parte nostra, non possiamo che guardare a questa mancata evoluzione come a un'altra occasione persa. Una macchina che, con il suo design elegante e la sua capacità di far girare sia il software professionale Psion tramite l'interfaccia ICE, sia i classici dello Spectrum, avrebbe rappresentato il perfetto terzo incomodo tra Commodore Amiga e Atari ST. Rimane oggi il ricordo malinconico di un salto tecnologico che la Sinclair avrebbe potuto compiere, se solo avesse avuto il tempo di tramutare le proprie idee in realtà.

Nota dell'Editore: le immagini a corredo di questo articolo sono state generate con l'ausilio di un servizio di IA online e sono frutto della creatività e dell'immaginazione dell'autore dell'articolo.





Accessori dal Retro-mondo NES Zapper

di Carlo Nithaiah Del Mar Pirazzini

In passato, le console hanno avuto diversi accessori per migliorare l'esperienza di gioco. I driving game hanno avuto dei controller speciali progettati come il cruscotto di un'auto, come l'Arcade Racer Joystick per il Sega Saturn e l'Hurricane Steering Wheel per PlayStation 3, gli sparatutto spaziali e i giochi di combattimento aereo hanno avuto dei joystick resi più autentici, come il NES Quick Shot e il Saturn Mission Stick. Ma per il sottoscritto, probabilmente l'accessorio più efficace è la pistola luminosa. È uno dei giocattoli più vecchi, che risale ai primi sistemi di gioco e anche prima. Su NES l'accessorio più iconico è senza dubbio la Pistola Zapper.



Come funziona

Lo Zapper è essenzialmente un controller a pulsante singolo. Quando si preme il grilletto, si invia un segnale alla console per oscurare lo schermo per un breve istante e tutti i bersagli disponibili sullo schermo si trasformano in quadrati bianchi. In quella frazione di secondo, un fotodiodo nella pistola viene attivato e se rileva la luce bianca, il sistema registra un colpo.

Quando ci sono più bersagli, ogni blocco bianco viene contrassegnato in sequenza e, a seconda della luce che si vede, il sistema determina quale bersaglio viene colpito. Alcune "leggende" dicono che sia possibile ingannare la pistola in modo che registri uno sparo mirando a una lampadina e premendo il grilletto, ma non ho mai saputo se funzionasse. Questo è probabilmente per lo stesso



motivo per cui lo Zapper non funziona sui moderni televisori a schermo piatto.

La Zapper ha un design che ricorda i vecchi serial di Flash Gordon. Semplice e leggero, ma sorprendentemente resistente. C'è solo un pulsante, il grilletto, che fa un forte clic quando viene premuto. Senza un pulsante di avvio, i giochi Zapper richiedono di collegare anche un controller normale (per la selezione e lo start). Il modello originale era completamente grigio, ma Nintendo realizzò il modello "arancio" nel 1989.

La precisione era il fiore all'occhiello della Zapper. Non richiedeva nessuna calibrazione o richieste delle pistole successive e moderne. Stando a 2 metri era facile sparare agli oggetti con un ragionevole modello di affidabilità.

Il gioco essenziale

I giochi disponibili per Zapper erano 17. Alcuni

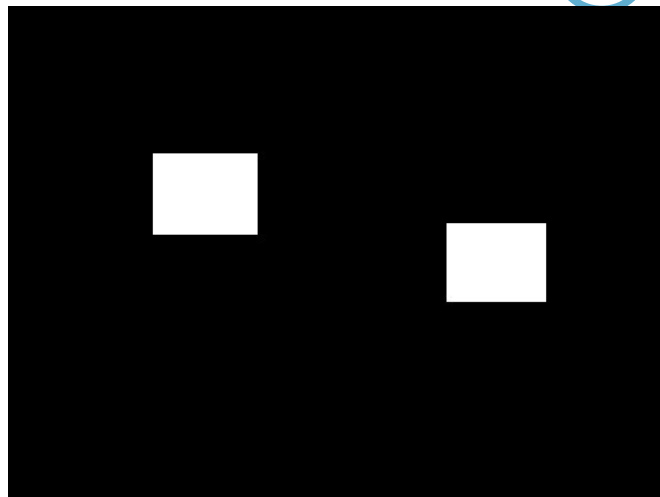




particolarmente divertenti ed originali, altri meno attrattivi. Ma sua maestà è senza dubbio Duck Hunt.

DUCK HUNT – Nintendo 1984

Pubblicato nel 1984 in Giappone, nel '85 in USA e nel 1987 nel resto del globo è stato il titolo di lancio nel mercato statunitense commercializzato nel bundle con la pistola. Ispirato ad un omonimo gioco del 1976 dotato di pistola ottica e proiettore, Duck Hunt aveva tre modalità di gioco. La prima consisteva nel cacciare una semplice anatra, la seconda con due anatre per volta e la terza il tiro al piattello. Un gioco divertentissimo e che fece conoscere al mondo il compare del personaggio giocatore, ovvero il segugio che, quando non si colpiscono gli avversari, ridacchia di noi. Lo ritroveremo come personaggio giocatore nella serie di Super Smash Bros con il nome di “Duo Duck Hunt”.





WaffleCopyPRO-Universal

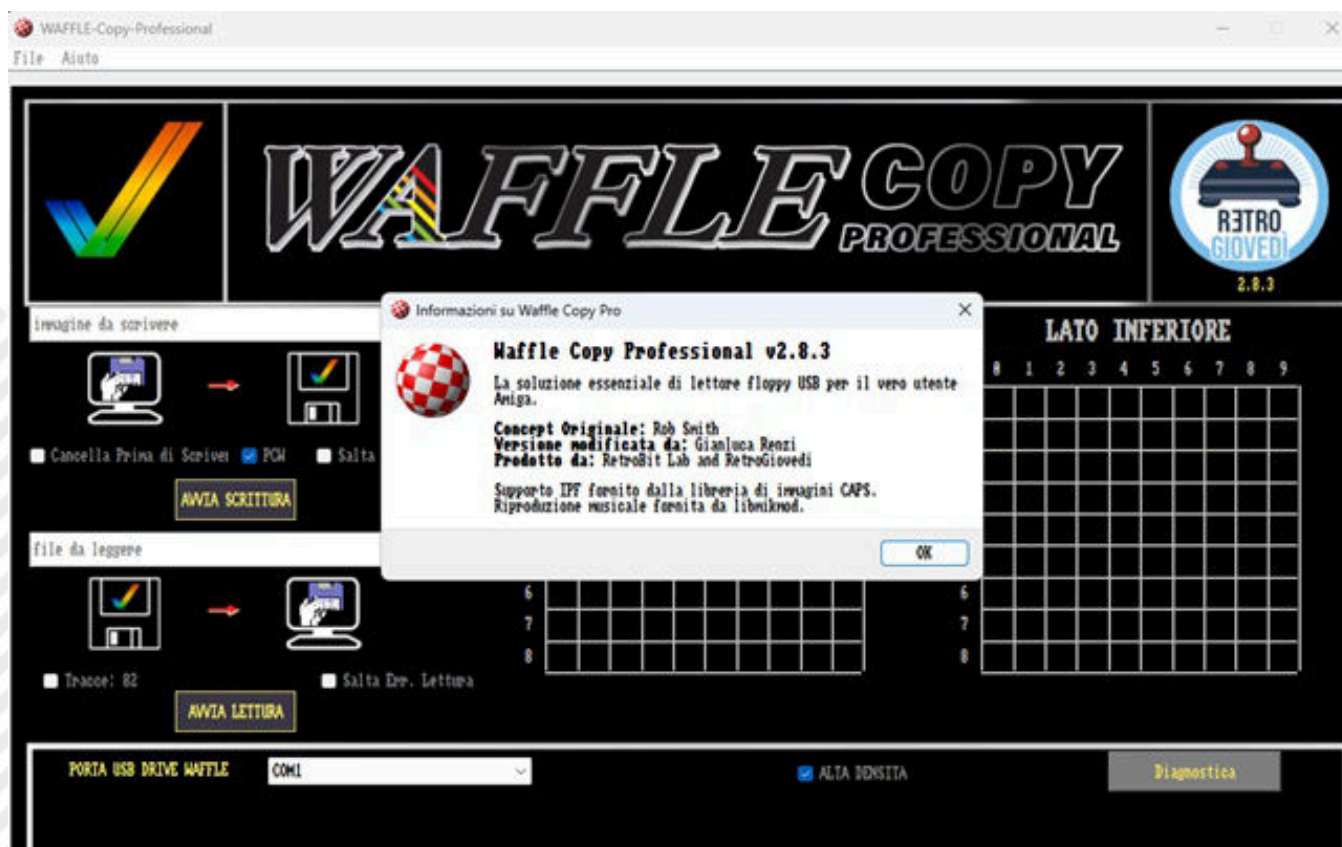
di Carlo Nithaiah Del Mar Pirazzini

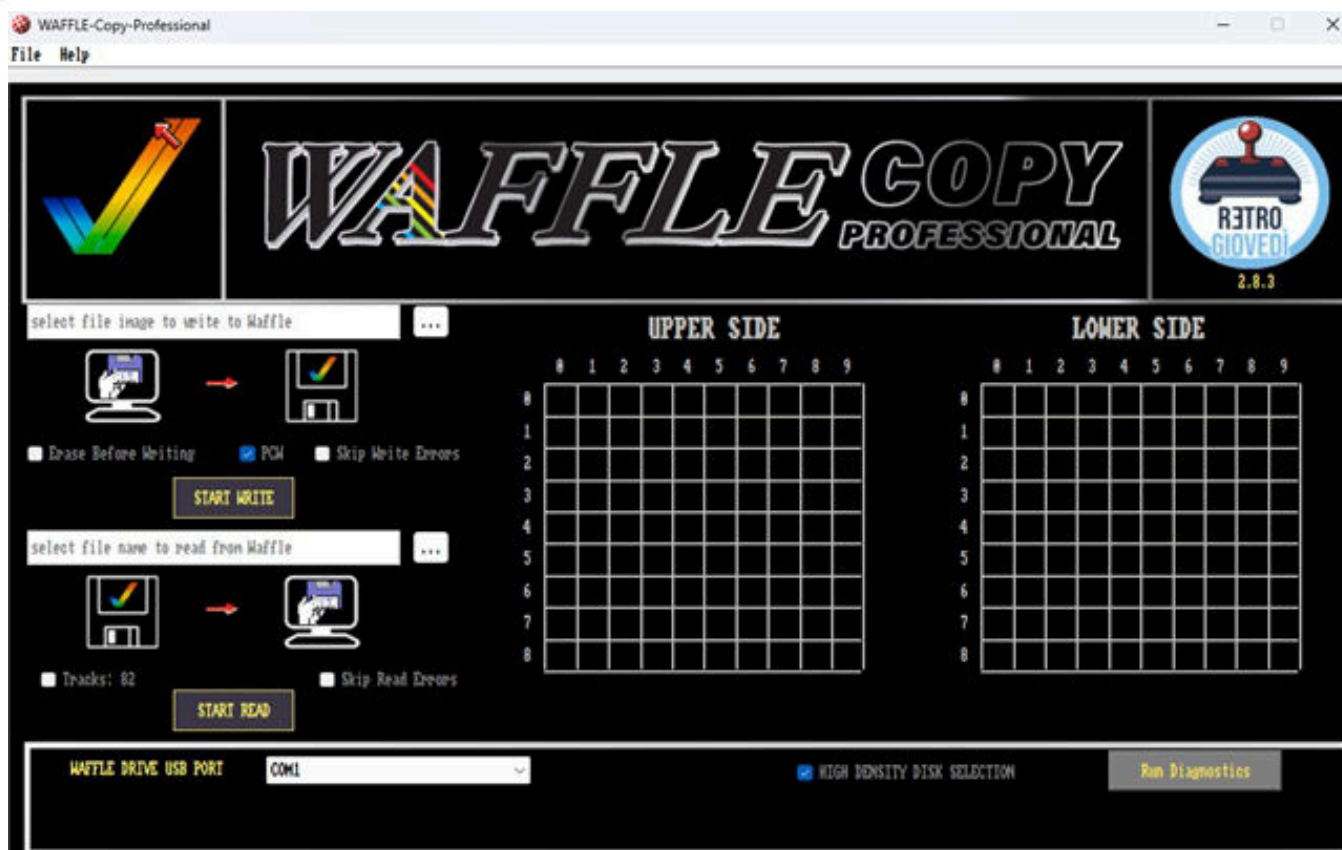
WaffleCopyPRO-Universal è il front-end “definitivo” per chi vuole usare Waffle o DrawBridge come drive floppy universale Amiga (e affini) su PC moderni, con un equilibrio riuscito tra potenza e usabilità. È un'applicazione cross-platform per Windows, macOS (sia Intel che Apple Silicon) e Linux, pensata per leggere e scrivere floppy tramite interfacce come la RetroGiovedì Waffle o l'Arduino DrawBridge, con un focus molto chiaro sui formati Amiga come ADF e IPF. Il target è l'appassionato retro che vuole preservare la propria collezione di dischetti reali e continuare a usarli sia su hardware originale sia in emulazione, evitando di doversi orientare tra tool sparsi, CLI ostiche e workflow artigianali.

Dal punto di vista funzionale, WaffleCopyPRO-Universal si comporta come un hub unico per diversi controller: supporta direttamente Waffle, DrawBridge e altri device Arduino/FTDI compatibili, accentrando in un'unica interfaccia attività che altrimenti richiederebbero più programmi differenti. Gestisce le immagini ADF e integra

staticamente il supporto IPF tramite la CAPS library, eliminando la necessità di plugin o configurazioni esoteriche per lavorare con i formati di preservazione più seri. Il flusso base è volutamente lineare: si seleziona la porta seriale, si inserisce il floppy, quindi si avvia una lettura per creare l'immagine o una scrittura per riversare un ADF/IPF su disco fisico. A questo si affiancano strumenti diagnostici e opzioni avanzate che aiutano a individuare se l'anomalia è dovuta al disco, al drive o al controller, aspetto molto utile in un contesto di supporti magnetici spesso al limite.

L'interfaccia utente ha un look dichiaratamente old school, e ci piace un sacco! Usa il font Amiga Topaz e incorpora elementi visivi da demoscene, come sine-scroller e temi grafici che strizzano l'occhio al mondo Amiga, ma senza sacrificare leggibilità e chiarezza. La UI è pensata per adattarsi tanto a schermi piccoli quanto a monitor più ampi, con particolare attenzione alla visibilità degli stati durante le operazioni di copia, dove la comunicazione di





progresso e stato d'errore è cruciale. Degno di nota è anche il supporto a sedici lingue, italiano incluso, una scelta inusuale per un tool così di nicchia e molto comoda se pensi a un utilizzo in contesti divulgativi o con utenti meno tecnici.

Sul fronte tecnico, il progetto è basato su Qt 5.15.2 e la procedura di build è documentata per Linux, macOS e Windows. Su sistemi Unix-like è sufficiente predisporre le dipendenze (Qt, serialport, libmikmod, libftdi e simili) e procedere con il classico qmake && make, mentre su Windows si appoggia a Visual Studio 2019, Qt e vcpkg per la gestione delle librerie. Il repository risulta attivo e aggiornato. Viene citata una release 2.8.3 che introduce, fra le altre cose, un controllo esplicito della versione firmware (con blocco per revisioni troppo vecchie, inferiori alla 1.8) e miglioramenti sul fronte debug e logging, segno che non si tratta di un esperimento lasciato a metà ma di un progetto vivo. Il fatto che sia open source lo rende inoltre interessante come base di studio o estensione, sia per aggiungere supporto ad altri controller o formati, sia come riferimento per integrare un'estetica retro coerente in un tool tecnico.

Nel complesso, WaffleCopyPRO-Universal si presenta come uno strumento maturo, curato e sorprendentemente accessibile per il tipo di lavoro che svolge. Se state valutando un flusso “leggo e scrivo dischi Amiga reali da un PC moderno” è sicuramente uno dei candidati più solidi e “presentabili” perché combina un'attenzione seria alla preservazione con uno stile dichiaratamente Amiga-centrico che parla direttamente al pubblico retro senza alienare chi cerca un'applicazione funzionale e affidabile.

Link Utili

Repository:

- <https://github.com/gianlucaenzi/ArduinoFloppyDiskReader>





Scopriamo Kyvos di George Sokianos

di Takahiro Yoshihoka, trad. di Carlo Nithaiah Del Mar Pirazzini

Kyvos è un front-end grafico multiplatforma per QEMU, pensato specificamente per creare e gestire macchine virtuali AmigaOS 4 e MorphOS in modo guidato e “chiavi in mano”, senza dover scrivere a mano righe di comando o script complessi.

Obiettivo e architettura generale

Kyvos si presenta come un'applicazione desktop per Linux, macOS x64 e Windows che fa da strato intermedio fra l'utente e QEMU. L'idea tecnica è fare interagire l'utente con una GUI strutturata in schede (hardware, argomenti, gestione VM), e Kyvos genera e lancia per lui i comandi QEMU corretti, occupandosi di path, parametri, periferiche, dischi e driver. Questo approccio riduce moltissimo la complessità d'ingresso per chi vuole usare AmigaOS 4 o MorphOS in emulazione, specialmente con configurazioni modernizzate (ad esempio virtio gpu).

L'applicazione è distribuita come archivio separato per ciascuna piattaforma desktop, accompagnata da un manuale di oltre 114 pagine che descrive in dettaglio installazione, configurazione delle VM, opzioni hardware e problemi noti. Dal punto di vista del ciclo di vita software, Kyvos viene aggiornato regolarmente (l'ultima versione indicata è la 1.8.2 di marzo 2026), con changelog strutturato per versione, che mostra una cura “da progetto maturo” più che da semplice utility amatoriale.

Gestione delle dipendenze e wizard di configurazione

Uno dei punti tecnici di forza è il sistema di onboarding: Kyvos integra un wizard che aiuta l'utente a localizzare o scaricare automaticamente le dipendenze fondamentali, cioè i binari di QEMU per il sistema operativo in uso e l'uso 7 Zip (7zip) per la gestione e l'estrazione degli archivi necessari alle VM

Il wizard controlla la presenza di questi componenti, guida

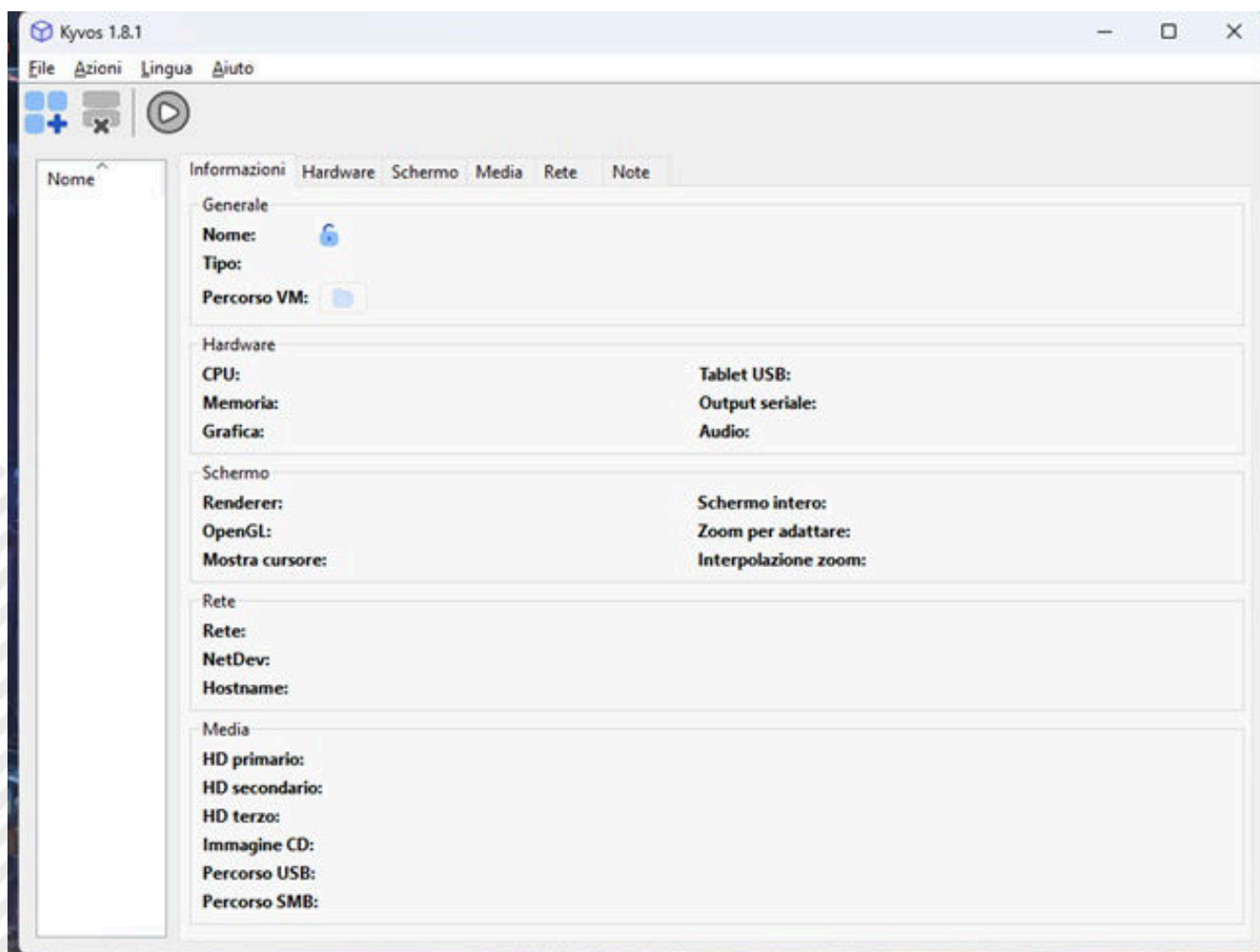


Fig. 1 - Appena installato (è un semplice .exe) appare così



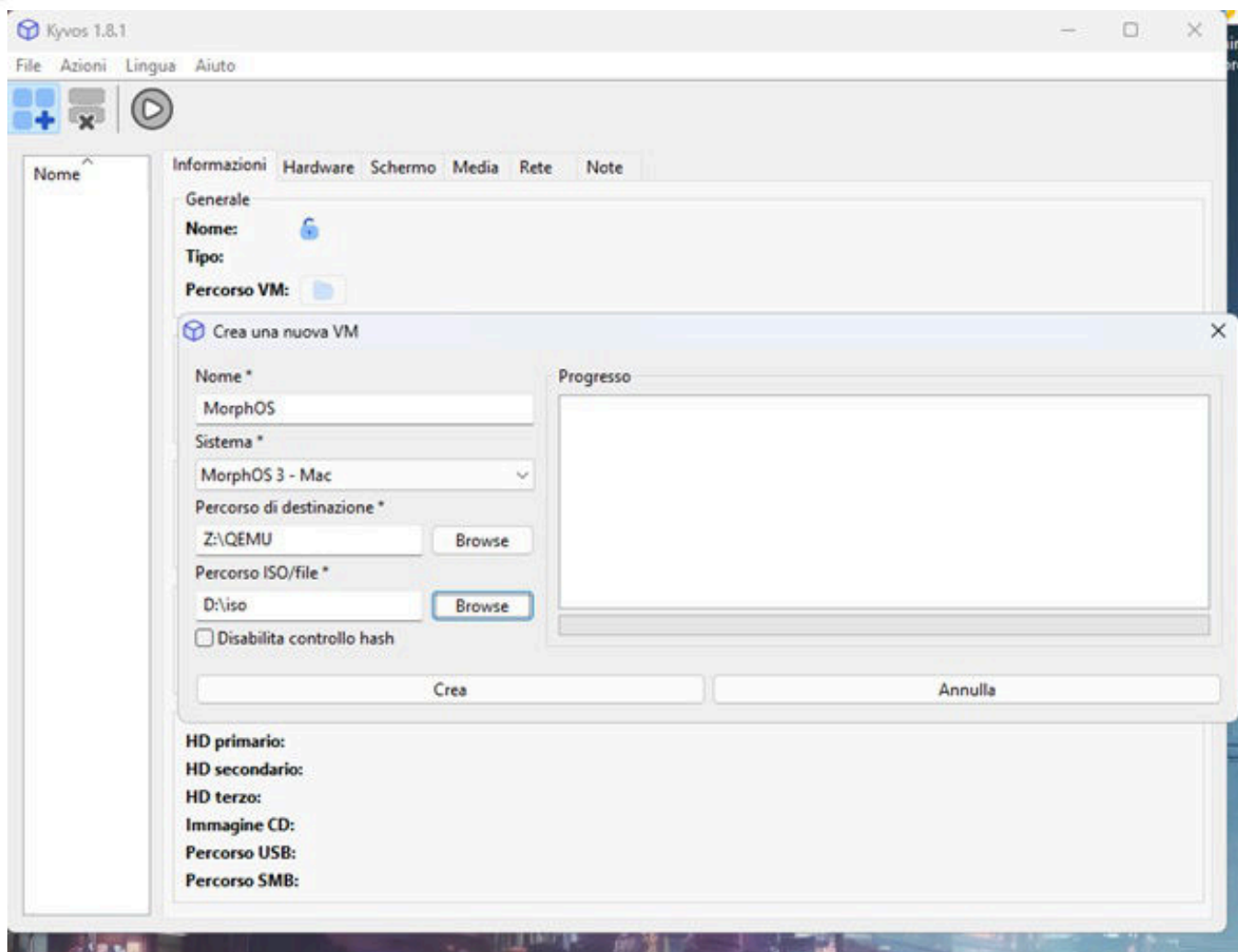


Fig. 2 - Creare una nuova “VM” è piuttosto semplice

al download se mancano e imposta i percorsi all'interno dell'applicazione. Questo evita all'utente di doversi arrangiare a mano con installazioni sparse di QEMU, con il rischio di parametri errati o versioni in conflitto.

Kyvos si posiziona quindi come orchestratore: non fornisce QEMU né i sistemi operativi AmigaOS 4 / MorphOS, ma si assicura che tutto ciò che è esterno (emulatore, tool di archiviazione) sia collegato e configurato correttamente, offrendo una pipeline coerente dall'installazione all'avvio della VM.

Modellazione dell'hardware virtuale e opzioni avanzate

La GUI include una scheda Hardware dedicata, che astrae le impostazioni di QEMU in campi strutturati (CPU, RAM, dischi, GPU, periferiche). Dal changelog emergono alcune funzionalità tecniche interessanti:

- Gestione di più dischi virtuali, con fix specifici per conflitti fra terzo hard disk e CD (segno che Kyvos si occupa direttamente del mapping device/drive in QEMU).
- Campo “Debug level” legato all'output seriale, utile per

chi vuole diagnosticare problemi della VM o del sistema guest (per esempio boot log dettagliati o traccia di errori kernel).

- Controllo sullo stato delle VM all'avvio di Kyvos: se una VM risulta già in esecuzione, l'interfaccia aggiorna l'icona nella lista, così l'utente ha subito il quadro dello stato del parco macchine.

È presente anche una gestione robusta degli errori di file system: è stato introdotto un controllo per gli errori nella cancellazione dei file delle VM, con message box informativa in caso di fallimento. Questo è un dettaglio “da tool serio”, che evita scenari ambigui in cui l'utente pensa di aver cancellato una VM ma i file rimangono sul disco.

Inoltre, è stata aggiunta una scheda dedicata ai parametri personalizzati di QEMU: l'utente avanzato può inserire argomenti extra, che vengono appesi alla linea di comando generata da Kyvos. Questo preserva la flessibilità di QEMU per i power user, pur mantenendo la semplicità per chi non vuole toccare nulla di avanzato.



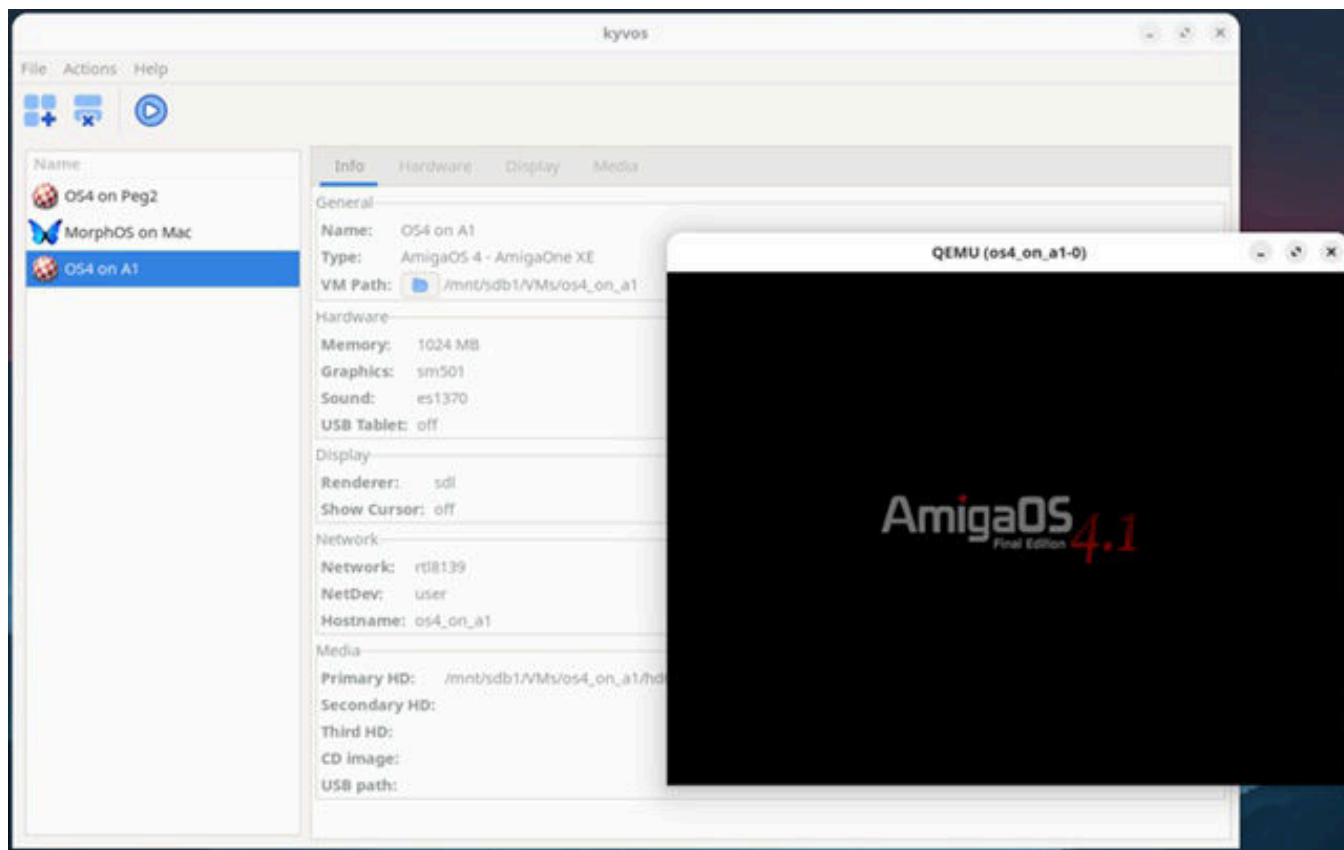


Fig. 3 - Installazione di AmigaOS 4.1 FE in Kyvos

Integrazione con virtio gpu e AmigaOS 4

Kyvos supporta la configurazione di ambienti AmigaOS 4 con driver virtio gpu, una delle direzioni più avanzate nell'emulazione moderna di AmigaOS, ma con una limitazione importante: il driver è ancora in beta privata, distribuito da A Eon, e non è incluso in Kyvos. Tecnicamente, ciò significa che Kyvos fornisce:

- Struttura di configurazione QEMU adatta a usare virtio gpu
- Supporto guidato per chi ha già accesso ai driver
- Documentazione e FAQ su incompatibilità note e peculiarità di QEMU in questo scenario

In pratica, Kyvos diventa il ponte tra l'utente che ha i file corretti (driver, ISO, ROM, ecc.) e una configurazione QEMU non banale da scrivere a mano, come quelle con GPU paravirtualizzata e impostazioni di performance fine tuning.

Manuale, UX e cura del progetto

Il manuale di oltre 100 pagine, con screenshot e guida passo passo, non è un semplice "readme", ma una vera documentazione utente che copre: Installazione su Linux, macOS x64 e Windows, creazione di nuove VM AmigaOS 4 / MorphOS, incompatibilità note di QEMU e loro workaround e l'uso delle varie schede dell'interfaccia,

incluse le opzioni avanzate

Il changelog mostra attenzione continua agli errori reali incontrati dagli utenti: bug fix relativi a campi non inizializzati



Fig. 4 - Il mega manuale di ben 126 pagine in PDF





("Table field 'qemu386' was not initialized!" nella finestra di benvenuto), oggetti mancanti (vmRunAutoInstall), reset indesiderati di percorsi personalizzati durante il setup. Questo indica un ciclo di feedback e manutenzione attivo, con test incrociati fra sistemi e configurazioni.

Dal punto di vista dell'esperienza d'uso, sono state aggiunte voci "Check for updates..." e "Roadmap" nel menu Help, segno che il progetto è pensato per convivere nel tempo con l'utente, non come tool "usa e getta".

Valutazione complessiva

Da un punto di vista tecnico, Kyvos è un front end GUI specializzato, con modello di configurazione interno che traduce impostazioni utente in comandi QEMU robusti. E' multiplatforma e mantenuto attivamente, con versioning chiaro e changelog orientato a bug reali e feature utili (debug seriale, argomenti custom, controllo VM in esecuzione). Fortemente documentato, con manuale e FAQ che coprono l'intero ciclo di vita delle VM AmigaOS 4 e MorphOS. Integrato (lato configurazione) con scenari

avanzati come l'uso di virtio gpu, pur senza distribuire componenti proprietari.

Per chi vuole emulare AmigaOS 4 o MorphOS senza diventare esperto di riga di comando QEMU, Kyvos offre un ambiente strutturato e professionale, che abbassa drasticamente la soglia d'ingresso e, allo stesso tempo, non castra le possibilità di tuning per gli utenti più esperti.

Link Utili

Github del progetto:

- <https://github.com/walkero-gr/kyvos>

Pagina Ko-Fi dello sviluppatore:

- <https://ko-fi.com/s/6476fdadd2>

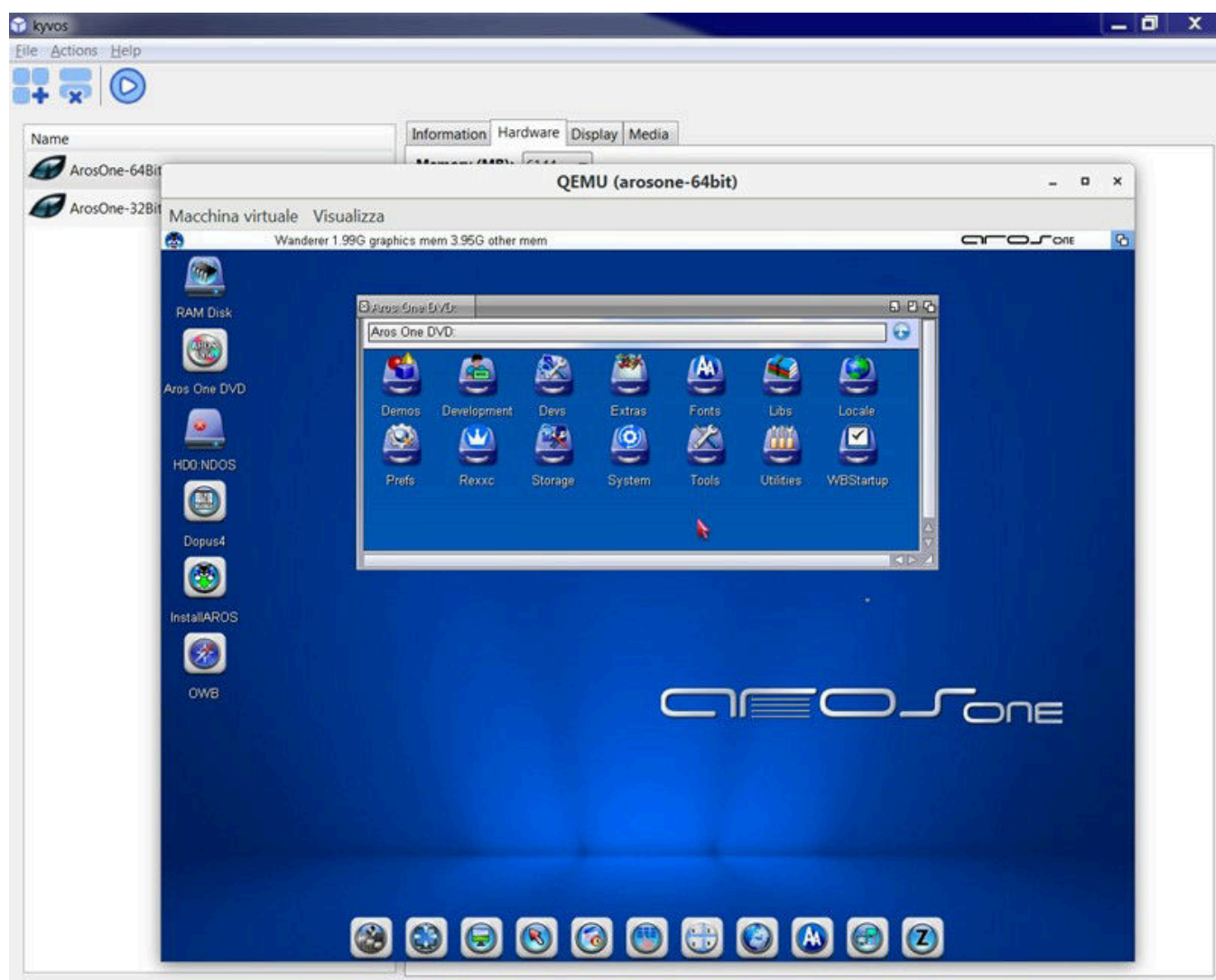


Fig. 5 - Aros emulato perfettamente





Caleidoscopio in varie salse (per C64 per principianti)

di Eugenio Rapella

È ben noto che lo schermo del C64 è “mappato” nelle locazioni di memoria dalla 1024 alla 2023 per quanto riguarda i caratteri, dalla 55296 alla 56295 per quanto riguarda i colori. Questo significa che l'istruzione POKE 1024,18 farà comparire la lettera R nella posizione in alto a sinistra dello schermo (i codici da utilizzare sono riportati nell'APPENDICE E del “Manuale d'uso”) e l'istruzione POKE 55296,1 la colorerà di bianco (APPENDICE G).

Il seguente programma basic (CALEIDO1.BAS) genera un “effetto caleidoscopio” sullo schermo del nostro Commodore:

```
100 print chr$(147):cl=55296
110 poke 53280,0:poke 53281,0
120 for x=0 to 19:for y=0 to 19
130 poke 1024+x+40*y,81
140 poke cl+x+40*y,0
150 next:next
160 w=rnd(-ti)
170 x=int(rnd(1)*10)
180 y=int(rnd(1)*10)
190 c=int(rnd(1)*256)
200 poke cl+x+40*y,c
210 poke cl+y+40*x,c
220 poke cl+19-x+40*y,c
230 poke cl+y+40*(19-x),c
240 poke cl+x+40*(19-y),c
250 poke cl+19-y+40*x,c
260 poke cl+19-x+40*(19-y),c
270 poke cl+19-y+40*(19-x),c
280 goto 170
```

Istruzioni 100-110: pulitura schermo, sfondo e bordo neri. Istr. 120-150: in via preliminare, la parte superiore sinistra viene riempita di un “tappeto” di 20*20 “pallini pieni” (130 POKE ...,81) che vengono immediatamente colorati di nero (istr. 140) per cui si mimetizzano con lo sfondo (ma rimarranno lì per tutto il tempo).

Alla 160 si inizializza il generatore di numeri casuali dopodiché ... si parte: le istruzioni dalla 170 alla 280 costituiscono un loop infinito che animerà il nostro caleidoscopio.

Alla 170, 180, 190 vengono generati tre valori casuali: X e Y sono degli interi tra 0 e 9, C è un intero tra 0 e 255. Alla 200 la POKE CL+X+40*Y,C fa sì che il colore di codice C venga assegnato alla casella dello schermo posizionata nella colonna X+1 e nella riga Y+1. Siccome i “pallini pieni” sono già presenti, l'effetto è quello di colorare il pallino (inizialmente nero, dunque invisibile) con il colore C. Successivamente lo stesso colore viene attribuito ad altri sette pallini che si trovano in posizione simmetrica al primo rispetto alla diagonale principale (si scambia X con Y), rispetto all'asse verticale che separa i primi 10 caratteri dai secondi 10 (si mantiene la riga Y e si sostituisce X con 19-X in POKE CL+19-X+40*Y,C), rispetto all'asse orizzontale ecc. (vedi FIG. 1).

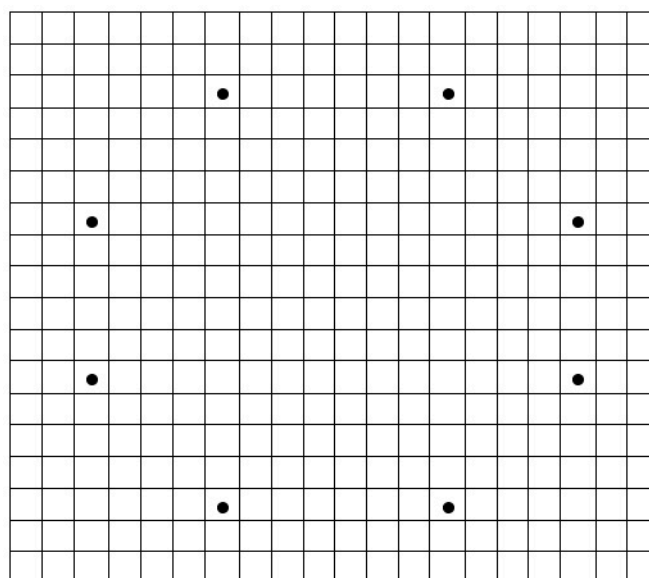


Fig. 1

La 280 rimanda alla 170 e tutto ricomincia: altri otto pallini si colorano di una nuova tinta, poi altri otto e così via.

Si genera un “caleidoscopio animato” di cui potete vedere un paio di frame nelle figure 2 e 3.



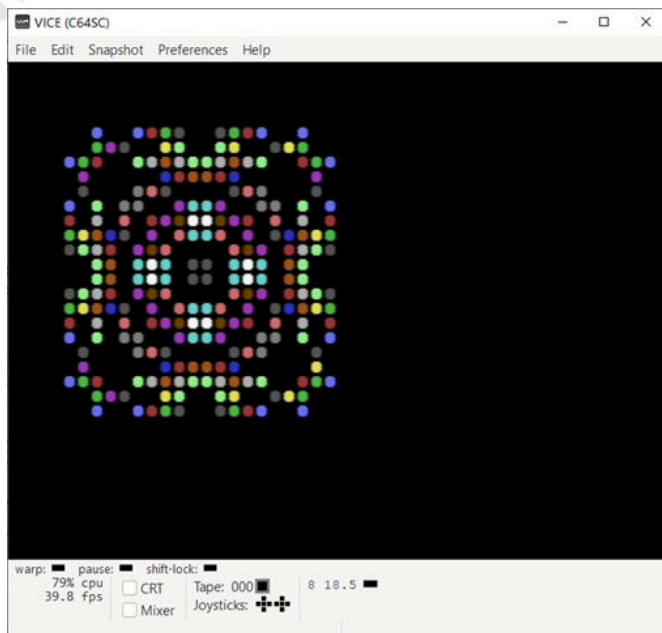


Fig. 2

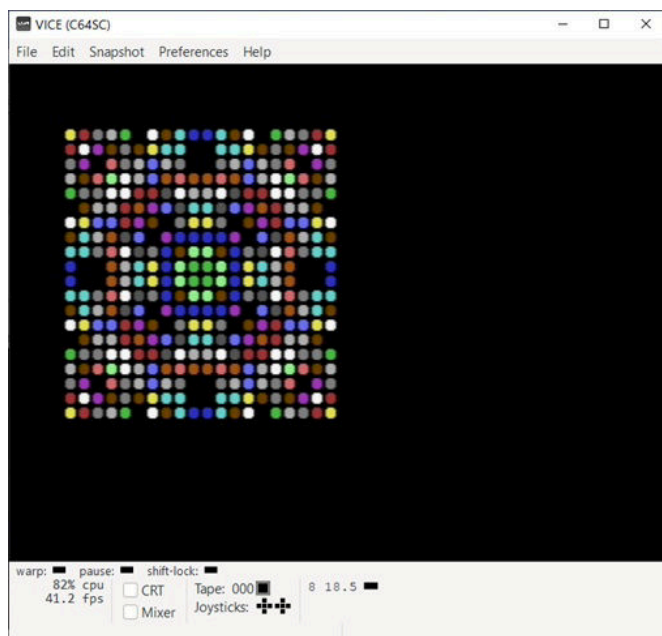


Fig. 3

A mio giudizio, l'effetto ottico è piacevole anche così, però possiamo provare a vedere cosa succede se riusciamo a velocizzare l'esecuzione.

All'interno del LOOP principale (170-280) vengono eseguite addizioni, sottrazioni, moltiplicazioni che il nostro amato Commodore esegue, ahimè, con una certa lentezza. Un possibile miglioramento, in termini di velocità, consiste nell'eseguire alcuni di questi conti in via preliminare al di fuori del LOOP 170-280 e memorizzare i valori in alcuni vettori.

Il programma CALEIDO2.BAS è una variazione di CALEIDO1.BAS:

```
10 dim dma(9),qa(9),qdm(9)
```

```
20 for a=0 to 9:dma(a)=19-a
30 qa(a)=40*a:qdm(a)=40*(19-a)
40 next
100 print chr$(147):cl=55296
110 poke 53280,0:poke 53281,0
120 for x=0 to 19:for y=0 to 19
130 poke 1024+x+40*y,81
140 poke cl+x+40*y,0
150 next:next
160 w=rnd(-ti)
170 x=int(rnd(1)*10)
180 y=int(rnd(1)*10)
190 c=int(rnd(1)*256)
200 poke cl+x+qa(y),c
210 poke cl+y+qa(x),c
220 poke cl+dma(x)+qa(y),c
230 poke cl+y+qdm(x),c
240 poke cl+x+qdm(y),c
250 poke cl+dma(y)+qa(x),c
260 poke cl+dma(x)+qdm(y),c
270 poke cl+dma(y)+qdm(x),c
280 goto 170
```

All'istruzione 10 vengono introdotti tre vettori di dimensione 10 (l'indice va da 0 a 9): DMA(..), QA(..) e QDM(..). Nel ciclo 20-40, questi vettori vengono caricati con i valori 19-A, 40*A e 40*(19-A) per i valori di A da 0 a 9.

CALEIDO2 coincide poi con CALEIDO1 tranne che nelle istruzioni dalla 200 alla 270: anziché eseguire il conto, il valore da associare alla POKE viene prelevato dal vettore.

Così, ad esempio, l'istruzione 270 di CALEIDO1: POKE CL+19-Y+40*(19-X),C

diviene: POKE CL+DMA(Y)+QDM(X),C.

Il tempo impiegato per eseguire i conti viene barattato con il tempo d'accesso agli elementi del vettore. Il guadagno è modesto, ma si nota: lanciando CALEIDO2 l'effetto ottico è un po' più veloce. Ho fatto numerosi esperimenti con programmini di supporto (con tanto di cronometro) e il responso è stato unanime: il risparmio di tempo è del 15 % (uso sempre l'emulatore VICE senza "warp-mode").

Se vogliamo un ulteriore, sensibile, aumento di velocità di esecuzione, potremmo utilizzare un compilatore (ad esempio Blitz Compiler) oppure ... passare all'Assembly.





In relazione all'Assembly del 6502, le mie competenze sono molto modeste. In questa parte mi rivolgo dunque ad un principiante come me: proviamo insieme a redigere un codice utilizzando le istruzioni più elementari.

La parte da velocizzare è quella all'interno del LOOP principale per cui manterremo in Basic la parte di inizializzazione che viene eseguita una sola volta.

Ecco il nostro "piano di lavoro" utilizzando CBM Prg Studio:

1. Creiamo un file Basic (CAL1.BAS) con la parte iniziale.
2. Creiamo il file CAL2.ASM con il codice in Assembly del LOOP principale.
3. Con l'opzione "Tools > Generate BASIC loader" avremo un codice Basic in cui le istruzioni di CAL2.ASM sono trasformate in una serie di DATA automaticamente caricate a partire dalla locazione scelta per il lancio del programma Assembly. Questo "loader" contiene anche l'istruzione SYS che manderà in esecuzione CAL2.ASM.
4. Basterà ora accodare questo codice Basic a CAL1.BAS. Avremo dunque un programma in Basic che esegue sia la parte iniziale sia il loop realizzato in Assembly: CAL3.BAS.

Ecco il file CAL1.BAS:

```
100 print chr$(147):sc=55296
110 poke 53280,0:poke 53281,0
120 for x=0 to 19:for y=0 to 19
130 poke 1024+x+40*y,81
140 poke sc+x+40*y,0
150 next:next
160 for kk=0 to 9
170 v1=19-kk:v2=40*kk:v3=40*v1
180 poke 838+kk,v1
190 h=int(v2/256):l=v2-256*h
200 poke 848+kk,l:poke 858+kk,h
210 h=int(v3/256):l=v3-256*h
220 poke 868+kk,l:poke 878+kk,h
230 next kk
```

Le istruzioni 100-150 sono le stesse di CALEIDO1 e CALEIDO2 (colorazione sfondo e bordo, "tappeto" di pallini colorati di nero); il ciclo 160-230 si incarica, per KK che va da 0 a 9, di inserire i valori 19-KK nelle 10 locazioni a partire dalla numero 838 (avremo dunque 19 in 838, 18 in 839, 17 in 840 ... 10 in 847).

I valori $V2=40*KK$ e $V3=40*(19-KK)$ calcolati alla 170 richiedono DUE byte; ad esempio, per $KK=9$, $V2$ e $V3$

assumono i valori 360 e 400 rispettivamente, entrambi valori che superano 255. L'istruzione 190 si incarica di separare il "byte alto" (H) e il "byte basso" (L) di $V2$. Ad esempio, quando $V2=360$, avremo $H=INT(360/256)=INT(1,40625)=1$ e $L=360-256*1=104$ (da notare che il valore decimale 104, in notazione esadecimale diviene \$68 e, complessivamente, il valore 360, in notazione esadecimale diviene \$0168).

Il "low byte" di $40*KK$ per KK da 0 a 9 viene memorizzato nelle locazioni consecutive a partire dalla numero 848; l' "high byte" a partire dalla 858.

Discorso del tutto analogo per $V3=40*(19-KK)$: alla 210 vengono valutati l'high e il low byte; i low byte vengono inseriti nelle locazioni a partire dalla 868, gli high byte a partire dalla 878.

Riassumendo, per KK che va da 0 a 9 avremo:

19-KK nelle locazioni da 838 a 847

Low byte di $40*KK$ nelle locazioni da 848 a 857

High byte di $40*KK$ nelle locazioni da 858 a 867

Low byte di $40*(19-KK)$ nelle locazioni da 868 a 877

High byte di $40*(19-KK)$ nelle locazioni da 878 a 887

Passiamo ora alla parte in Assembly. Per generare i numeri casuali ho utilizzato un codice che avevo scovato in Internet molto tempo fa. Ne avevo già fatto uso in un contributo su questa rivista (" Il gioco del 'Mini-Lotto' " sul n. 47 di RMW) dove chi è interessato può trovare una sintesi della spiegazione che Marco Pistorio mi aveva fornito in relazione a questo codice.

Il generatore necessita di una "inizializzazione":

```
lda #$ff ;*****
sta $d40e ; inizializzazione
sta $d40f ; del generatore
lda #$80 ; di numeri casuali
sta $d412 ;*****
```

dopodiché, ogni volta che si va a leggere il contenuto della locazione \$D41B, si ottiene un valore casuale nel range 0-255.

Dopo le prime istruzioni (la $*=\$C000$ segnala la locazione di partenza del codice Assembly) ha inizio il LOOP principale: sono le istruzioni comprese tra l'etichetta LOOP e l'istruzione





JMP LOOP. All'interno di questo loop si accede più volte ad una coppia di subroutine che conviene analizzare ora:

SUBROUTINE TETA

; subroutine teta: genera in A

; una cifra casuale tra 0 e 9

teta

alfa lda \$d41b ; in A rnd [0,255]

and #\$0f ; ora in A [0,15]

cmp #\$0a ; è maggiore o uguale a 10?

bcs alfa ; se sì torna a alfa

rts

il numero casuale generato in \$D41B viene caricato nell'accumulatore A; la AND #\$0F esegue un AND logico con il valore \$0F che è 00001111 in binario. L'effetto è quello di azzerare i quattro bit più significativi e di mantenere inalterati i quattro meno significativi. Nell'accumulatore ci sarà ora un valore tra 0 e 15. La CMP #\$0A confronta A con 10 e BCS ALFA rimanda all'etichetta ALFA se A è maggiore o uguale a 10. Il rientro al programma principale (RTS) avviene solo se A è minore di 10, dunque se A è un intero tra 0 e 9.

SUBROUTINE BETA

Come abbiamo visto in CALEIDO1.BAS, nel LOOP principale lo stesso colore va caricato in otto locazioni: $XX+40*YY+55296$, $YY+40*XX+55296$, $(19-XX)+40*YY+55296$ ecc. (qui usiamo XX e YY per evitare confusione con i registri X e Y che entrano in gioco nel codice in Assembly).

Il primo addendo, XX o YY o 19-XX o 19-YY, occupa un solo byte, il secondo ($40*(..)$) ne occupa due. Il terzo addendo, 55296, è \$D800 in esadecimale, dunque il suo low byte è 00 e l'high byte è \$D8.

Il compito della subroutine BETA è quello di eseguire l'addizione e di inserire poi il codice del colore, presente alla locazione 890, al posto giusto. Prima della chiamata, il primo addendo viene caricato nella locazione 891, il low byte del secondo nella 892 e l'high byte nella 893. Dunque, in relazione all'addizione richiesta, abbiamo:

LOW BYTE DELLA SOMMA: contenuto di 891 + contenuto di 892 (il low byte del terzo addendo è zero quindi non va addizionato)

HIGH BYTE DELLA SOMMA: contenuto di 893 + \$D8 (+

l'eventuale riporto originato dall'addizione precedente).

Il low byte della somma viene memorizzato nella locazione 894, l'high byte nella locazione 895.

Per travasare il contenuto dell'Accumulatore in una locazione di memoria a 16 bit, occorre operare attraverso l'indirizzamento indiretto. Con l'istruzione STA(zero page),Y, il contenuto di A viene memorizzato nella locazione di memoria il cui byte meno significativo (LSB) si trova nell'indirizzo "zero page" e quello più significativo (MSB) in quello immediatamente seguente. Una volta recuperato l'indirizzo a 16 bit, il processore addiziona il contenuto del registro Y per ottenere l'indirizzo di destinazione.

Nella routine BETA, l'indirizzo "zero page" è \$FB: dopo aver travasato il contenuto di 894 e 895 in \$FB e \$FC (LDA 894, STA \$FB, LDA 895, STA \$FC) basterà caricare in A il codice del colore in gioco (LDA 890) e riversarlo nella giusta locazione (STA(\$FB),Y) dopo aver azzerato il contenuto del registro Y (LDY #00).

Ecco il codice:

; routine che esegue la somma e piazza

; il colore C nella locazione

beta

clc ; clear the carry flag

lda 891

adc 892

sta 894 ; low della somma

lda #\$d8

adc 893

sta 895 ; high della somma

lda 894

sta \$fb ; zero page

lda 895

sta \$fc ; zero page + 1

lda 890 ; carica codice colore

ldy #00

sta (\$fb),y

rts

Ecco qui di seguito il codice completo del programma CAL2.ASM in cui ci rimane da analizzare il loop principale:

***=\$c000 ;sys 49152**





```
lda #$ff ;*****
sta $d40e ; inizializzazione
sta $d40f ; del generatore
lda #$80 ; di numeri casuali
sta $d412 ;*****
loop lda $d41b ; in A rnd [0,255]
sta 890 ; in 890 codice colore
jsr teta ; ora in A rnd [0,9]
sta 888 ; xx in [0,9] va in 888
jsr teta
sta 889 ; yy in [0,9] va in 889

; xx+40yy (1)
lda 888 ; xx in A
sta 891 ; xx in 891
ldx 889 ; in reg X il valore yy (offset)
lda 848,x ; low byte di 40yy
sta 892 ; va in 892
lda 858,x ; high byte di 40yy
sta 893 ; va in 893
jsr beta

; yy+40xx (2)
lda 889 ; yy in A
sta 891 ; yy in 891
ldx 888 ; in reg X il valore xx (offset)
lda 848,x ; low byte di 40xx
sta 892 ; va in 892
lda 858,x ; high byte di 40xx
sta 893 ; va in 893
jsr beta

; yy+40(19-xx) (3)
lda 889 ; yy in A
sta 891 ; yy in 891
ldx 888 ; in reg X il valore xx (offset)
lda 868,x ; low byte di 40(19-xx)
sta 892 ; va in 892
lda 878,x ; high byte di 40(19-xx)
sta 893 ; va in 893
jsr beta

; xx+40(19-yy) (4)
lda 888 ; xx in A
sta 891 ; xx in 891

ldx 889 ; in reg X il valore yy (offset)
lda 868,x ; low byte di 40(19-yy)
sta 892 ; va in 892
lda 878,x ; high byte di 40(19-yy)
sta 893 ; va in 893
jsr beta

; (19-xx)+40(19-yy) (5)
ldx 888 ; xx in reg X
lda 838,x ; 19-xx in A
sta 891 ; 19-xx in 891
ldx 889 ; in reg X il valore yy (offset)
lda 868,x ; low byte di 40(19-yy)
sta 892 ; va in 892
lda 878,x ; high byte di 40(19-yy)
sta 893 ; va in 893
jsr beta

; (19-yy)+40(19-xx) (6)
ldx 889 ; yy in reg X
lda 838,x ; 19-yy in A
sta 891 ; 19-yy in 891
ldx 888 ; in reg X il valore xx (offset)
lda 868,x ; low byte di 40(19-xx)
sta 892 ; va in 892
lda 878,x ; high byte di 40(19-xx)
sta 893 ; va in 893
jsr beta

; (19-yy)+40xx (7)
ldx 889 ; yy in reg X
lda 838,x ; 19-yy in A
sta 891 ; 19-yy in 891
ldx 888 ; in reg X il valore xx (offset)
lda 848,x ; low byte di 40xx
sta 892 ; va in 892
lda 858,x ; high byte di 40xx
sta 893 ; va in 893
jsr beta

; (19-xx)+40yy (8)
ldx 888 ; xx in reg X
lda 838,x ; 19-xx in A
sta 891 ; 19-xx in 891
ldx 889 ; in reg X il valore yy (offset)
```





```
lda 848,x    ; low byte di 40yy
sta 892      ; va in 892
lda 858,x    ; high byte di 40yy
sta 893      ; va in 893
jsr beta
```

```
jmp loop
```

```
; subroutine teta: genera in A
; una cifra casuale tra 0 e 9
teta
alfa lda $d41b    ; in A rnd [0,255]
    and #$0f      ; ora in A [0,15]
    cmp #$0a      ; è maggiore o uguale a 10?
    bcs alfa      ; se sì torna a alfa
    rts
```

```
; routine che esegue la somma e piazza
; il colore C nella locazione
beta
```

```
clc          ; clear the carry flag
lda 891
adc 892
sta 894      ; low della somma
lda #$d8
adc 893
sta 895      ; high della somma
lda 894
sta $fb      ; zero page
lda 895
sta $fc      ; zero page + 1
lda 890      ; carica codice colore
ldy #00
sta ($fb),y
rts
```

Iniziamo appunto dalle istruzioni a partire dall'etichetta LOOP:

* viene generato il numero casuale, codice del colore da assegnare alle otto caselle; questo valore viene memorizzato nella locazione 890

* grazie alla subroutine TETA vengono generati i due valori XX e YY, due interi casuali tra 0 e 9. XX va nella locazione 888, YY nella 889.

A questo punto, per otto volte, occorre andare a pescare gli elementi giusti dei vettori preparati in CAL1.BAS, elementi da passare alla routine BETA che posizionerà il colore.

Le singole istruzioni sono commentate sinteticamente all'interno del codice, vediamo comunque un paio di esempi in dettaglio per capire meglio il meccanismo:

Esempio 1: $XX + 40 * YY$

```
lda 888: il valore di XX va in A (l'Accumulatore)
sta 891: ora XX va in 891, pronto per la routine BETA
ldx 889 nel registro X va il valore YY
lda 848,x ora in A c'è l'elemento contenuto in 848+YY
        ovvero il low byte di 40*YY
sta 892 il low byte di 40*YY va nella locazione 892
        pronto per la routine BETA
lda 858,x ora in A c'è l'elemento contenuto in 858+YY
        ovvero l'high byte di 40*YY
sta 893 l'high byte di 40*YY va nella locazione 893
        pronto per la routine BETA
```

I parametri necessari alla routine BETA sono in posizione; siamo pronti per ...

```
jsr BETA
```

Esempio 2: $(19 - YY) + 40 * XX$

```
ldx 889 il valore di YY va nel registro X
lda 838,x in A va il valore contenuto in 838+YY
        ovvero il valore 19-YY
sta 891 (19-YY) va in 891
        pronto per la routine BETA
ldx 888 nel registro X va ora XX
lda 848,x ...
da qui si prosegue come nell'esempio 1.
```

Dopo gli otto segmenti di codice, si giunge al JMP LOOP per una nuova coppia di coordinate e un nuovo colore.

Siamo alla fine: utilizziamo il Tool "Generate BASIC loader", chiudiamo la finestra "Assembly Dump" e scegliamo come "Start Line Number" un numero superiore ai numeri di linea utilizzati in CAL1.BAS. Io ho scelto 1000 con un incremento di 20. Il risultato andrà copiato e incollato al programma CAL1.BAS.

Ecco qui sotto il programma finale, CAL3.BAS, che si ottiene. Potete copiarlo, incollarlo ed eseguirlo come un





nuovo file Basic di CBM Prg Studio senza operare personalmente attraverso "Generate BASIC loader".

CAL3.BAS:

```

100 print chr$(147):sc=55296
110 poke 53280,0:poke 53281,0
120 for x=0 to 19:for y=0 to 19
130 poke 1024+x+40*y,81
140 poke sc+x+40*y,0
150 next:next
160 for kk=0 to 9
170 v1=19-kk:v2=40*kk:v3=40*v1
180 poke 838+kk,v1
190 h=int(v2/256):l=v2-256*h
200 poke 848+kk,l:poke 858+kk,h
210 h=int(v3/256):l=v3-256*h
220 poke 868+kk,l:poke 878+kk,h
230 next kk
1000 REM GENERATED ML LOADER
1020 SA = 49152
1040 FOR N = 0 TO 283
1060 READ A% : POKE SA+N,A%: NEXT N
1080 SYS SA
1100 END
1120 DATA 169,255,141,14,212,141,15,212
1140 DATA 169,128,141,18,212,173,27,212
1160 DATA 141,122,3,32,238,192,141,120
1180 DATA 3,32,238,192,141,121,3,173
1200 DATA 120,3,141,123,3,174,121,3
1220 DATA 189,80,3,141,124,3,189,90
1240 DATA 3,141,125,3,32,248,192,173
1260 DATA 121,3,141,123,3,174,120,3
1280 DATA 189,80,3,141,124,3,189,90
1300 DATA 3,141,125,3,32,248,192,173
1320 DATA 121,3,141,123,3,174,120,3
1340 DATA 189,100,3,141,124,3,189,110
1360 DATA 3,141,125,3,32,248,192,173
1380 DATA 120,3,141,123,3,174,121,3
1400 DATA 189,100,3,141,124,3,189,110
1420 DATA 3,141,125,3,32,248,192,174
1440 DATA 120,3,189,70,3,141,123,3
1460 DATA 174,121,3,189,100,3,141,124
1480 DATA 3,189,110,3,141,125,3,32
1500 DATA 248,192,174,121,3,189,70,3
1520 DATA 141,123,3,174,120,3,189,100

```

```

1540 DATA 3,141,124,3,189,110,3,141
1560 DATA 125,3,32,248,192,174,121,3
1580 DATA 189,70,3,141,123,3,174,120
1600 DATA 3,189,80,3,141,124,3,189
1620 DATA 90,3,141,125,3,32,248,192
1640 DATA 174,120,3,189,70,3,141,123
1660 DATA 3,174,121,3,189,80,3,141
1680 DATA 124,3,189,90,3,141,125,3
1700 DATA 32,248,192,76,13,192,173,27
1720 DATA 212,41,15,201,10,176,247,96
1740 DATA 24,173,123,3,109,124,3,141
1760 DATA 126,3,169,216,109,125,3,141
1780 DATA 127,3,173,126,3,133,251,173
1800 DATA 127,3,133,252,173,122,3,160
1820 DATA 0,145,251,96

```

Ora, per finire, ancora un paio di versioni. Ecco quella per lo splendido ugBASIC di Marco Spedaletti:

OPTION DEFAULT TYPE BYTE

RANDOMIZE TIMER

POKE 53272,21

CLS:w\$=CHR\$(209)

COLOR BACKGROUND 0

COLOR BORDER 0

x1=10:y1=10

DO

x=RND(x1):y=RND(y1)

c=RND(15):PEN c

LOCATE x,y:PRINT w\$

LOCATE 19- x,y:PRINT w\$

LOCATE y,x:PRINT w\$

LOCATE y,19-x:PRINT w\$

LOCATE x,19-y:PRINT w\$

LOCATE 19-x,19-y:PRINT w\$

LOCATE 19-y,x:PRINT w\$

LOCATE 19-y,19-x:PRINT w\$

LOOP

In questa versione non viene utilizzato il POKE; il cursore viene posizionato tramite il comando LOCATE prima della stampa del "pallino pieno" (è CHR\$(209) se preceduto da POKE 53272,21) e il colore C viene assegnato tramite il comando PEN C (le KEYWORDS sono descritte dettagliatamente nello "User Manual").





Un bel po' di anni fa avevo scoperto "Turbo Rascal Syntax Error" che era da poco comparso sulla scena; naturalmente, l'avevo subito messo alla prova con il mio CALEIDOSCOPIO.

In relazione al mio programmino, avevo scambiato alcune mail con Nicolaas Groeneboom, l'autore di TRSE, che mi aveva proposto di inserirlo tra i tutorials.

È presente anche oggi, in una versione leggermente modificata da Nicolaas.

Lo trovate in:

C64 Tutorials > easy > TR 19 caleidoscope.ras

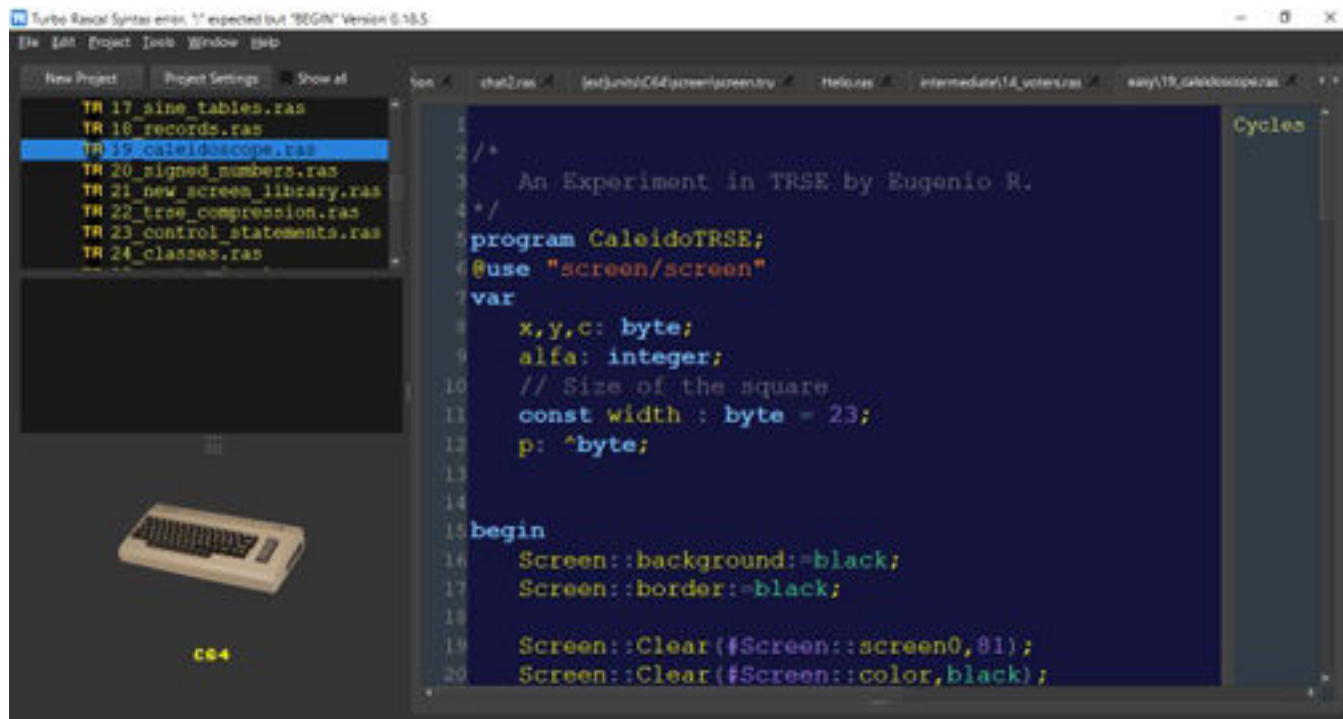


Fig. 4 - C64 Tutorials > easy > TR 19 caleidoscope.ras

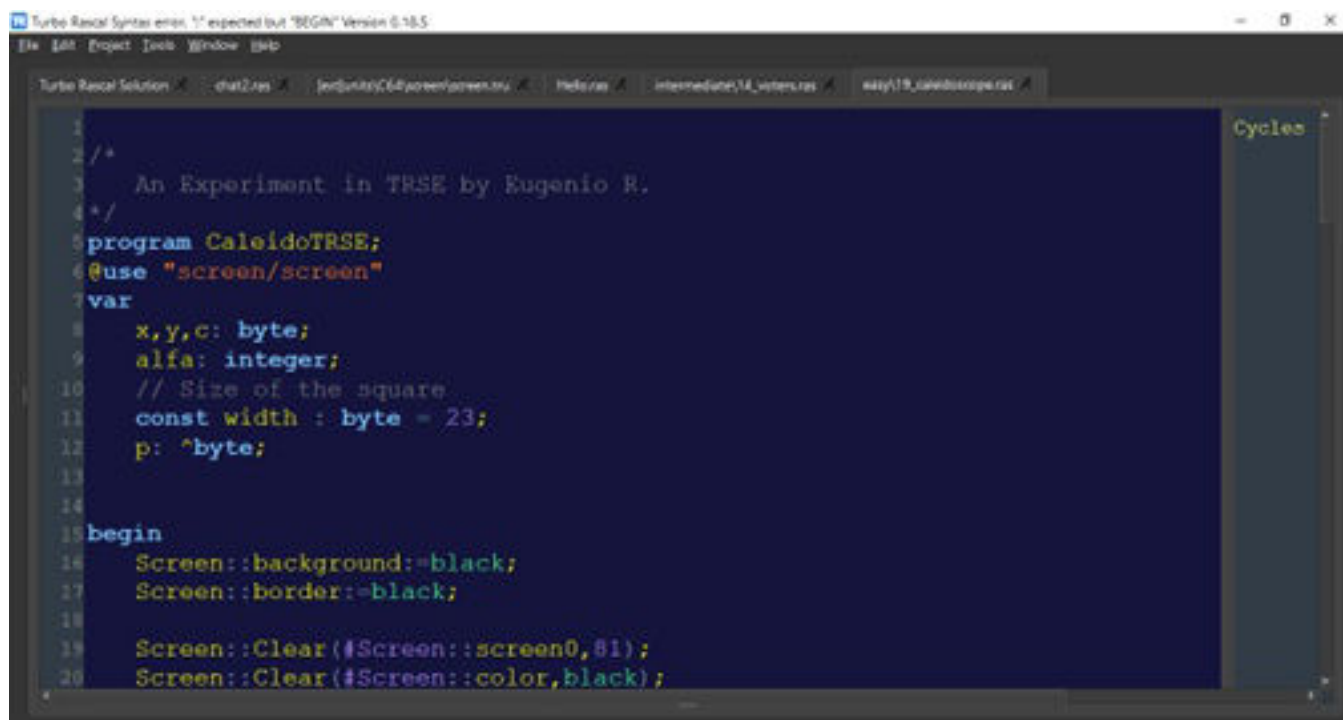


Fig. 5 - Il programma caleidoscope.ras caricato nell'editor del TRSE





Un gioco di prestigio (per C64 per principianti)

di Eugenio Rapella

Su YouTube potete trovare decine e decine di video relativi a giochi di prestigio con le carte da gioco e non solo. Solitamente alla presentazione del gioco segue un "tutorial" in cui viene svelato il trucco. C'è un tipo davvero simpatico, Matt McGurk, che presenta numerosi giochi. Eccone uno: Formate un mazzo scegliendo 15 carte da un mazzo di 52 carte, date un'occhiata all'ultima carta del mazzo che avete creato, memorizzatela e invitate uno spettatore, chiamiamolo Francesco, a partecipare al gioco. Tanto per cominciare, recitate la parola "MAGIC" lettera per lettera; ad ogni lettera dello spelling, spostate una carta dalla cima del vostro mazzo e create un mazzetto di cinque carte di fronte e voi, poi lasciate cadere il resto del mazzo sopra quello che avete creato (naturalmente le carte sono sempre coperte). A questo punto, in gran segreto, scrivete su un foglietto la carta che avete memorizzato, e mettete da parte il foglietto facendo notare a Francesco che quanto avete scritto non potrà più essere modificato. Ora chiedete a Francesco di scegliere una delle 52 carte del mazzo e, sempre facendo lo spelling, contegiate le carte del vostro mazzo a partire dall'alto creandone uno nuovo. Alla fine lasciate cadere le carte rimaste sopra il mazzetto che avete creato. Ad esempio, se Francesco ha scelto la donna di cuori, farete lo spelling di "QUEEN OF HEARTS" facendo scorrere dunque 13 carte (per ora manteniamo la lingua inglese, in accordo con il video; vedremo dopo come modificare il tutto per eseguire il gioco in lingua italiana). Durante questa fase, potete far notare a Francesco che il numero di carte spostate dipende dalla carta scelta, ad esempio "TWO OF CLUBS" comporta lo spostamento di 10 carte e non 13 come nell'esempio precedente e ... il foglietto è già stato scritto!.

Siamo quasi alla fine: ora fate lo spelling della parola "TRICK" e, come prima, create un nuovo mazzetto di cinque carte sopra il quale lascerete cadere in blocco le rimanenti.

Recuperate il foglietto: la vostra previsione coinciderà magicamente con la carta in cima al mazzo!

Come è possibile? Per svelare il mistero ci viene in aiuto

il nostro C64. Consideriamo N carte impilate coperte, ovvero con il dorso rivolto allo spettatore, e sia $S1$ la posizione di una carta a partire dal basso. Ora immaginiamo di prendere K carte (con K compreso tra 1 e N) partendo dall'alto, creare con queste un nuovo mazzetto e sovrapporre in blocco il mazzo delle carte rimaste. Qual è ora la nuova posizione $S2$ (sempre dal basso) della carta che era in posizione $S1$?

Se K è minore od uguale a $N-S1$, le K carte spostate vanno a finire sul fondo del nuovo mazzo e lo spostamento in blocco del resto del mazzo porta la carta in posizione $S1$ a salire di K posti per cui $S2=K+S1$. Ad esempio, se $N=20$, $S1=3$ e $K=5$, quando si crea il nuovo mazzo, le cinque carte prese dall'alto finiscono sul fondo e quando si sovrappone il blocco rimasto, la carta che era in terza posizione dal basso, va a finire in ottava posizione (sempre dal basso, naturalmente).

Diverso è il discorso se K è maggiore di $N-S1$ perché in questo caso lo spostamento delle K carte coinvolge anche quella in posizione $S1$. Le prime $N-S1$ carte dall'alto diventano le prime $N-S1$ dal basso nel nuovo mazzo dopodiché arriva la carta che era in posizione $S1$.

Ecco dunque che avremo $S2=N-S1+1$. Ad esempio, se $N=20$, $S1=12$ e $K=15$, dopo aver spostato le prime $20-12=8$ carte, toccherà alla carta che era in posizione $S1=12$ che andrà ad occupare la nona posizione dal basso: $8+1=20-12+1=N-S1+1$.

Nel Basic del C64 le espressioni logiche come $(A<B)$ hanno valore "-1" se vere, valore "0" se false per cui possiamo definire una funzione che, a partire dal numero X di carte che si intendono spostare per creare il nuovo mazzetto, fornisca la nuova posizione $NP(X)$ di una carta inizialmente in posizione $S1$. Nell'istruzione 10 del programma che segue, se X è minore o uguale a $N-S1$, il valore di $(X \leq N-S1)$ è "-1" mentre quello di $(X > N-S1)$ è 0 per cui la funzione $FN NP(X)$ assume il valore di $X+S1$. In caso contrario è $(X \leq N-S1)=0$, $(X > N-S1)=-1$ e $FN NP(X)$ diviene $N-S1+1$.



Nel gioco di prestigio, così come è stato descritto, si utilizza un mazzo formato da $N=15$ carte; all'inizio la "carta segreta" da segnare sul foglietto è l'ultima delle 15 carte, dunque in posizione $S1=1$ a partire dal fondo. Per comodità, chiamiamo con $ALFA(Y)$ la procedura "scorri un certo numero Y di carte per creare un mazzetto e sovrapponi le rimanenti"; $ALFA(Y)$ viene ripetuta tre volte: la prima con $Y=5$ (con lo spelling di MAGIC), la seconda con un numero Y variabile (ad esempio $Y=10$ nel caso di ACE OF CLUBS) e la terza di nuovo con $Y=5$ (con lo spelling di TRICK).

Il programma MCGURK gestisce la situazione più in generale. Il suo obiettivo è quello di vedere come cambia la posizione finale di una certa carta al variare dei vari parametri, soprattutto al variare del numero di carte della seconda ripetizione della procedura che abbiamo descritto:

Ecco il listato di MCGURK:

```

10 def fn np(x)=- (x<=n-s1)*(x+s1)-(x>n-s1)*(n-s1+1)
20 print chr$(5):input"totale carte ";n
30 input"posizione carta segreta";s1
40 input"n. carte mazzetto";m
50 sw=s1:print
60 for k=1 to n
70 a=fn np(m):s1=a:b=fn np(k):s1=b:c=fn np(m)
80 print k,a;b;c
90 s1=sw:next

```

Alla 10 viene definita la funzione che, data una carta in posizione $S1$, ne fornisce la nuova posizione $NP(X)$ quando si esegue la procedura $ALFA(X)$. Alla 20 si chiede il numero delle carte (variabile N) che si utilizzano per il gioco, alla 30 la posizione iniziale della carta da identificare (variabile $S1$). Alla 40 si chiede quante carte entrano in gioco la PRIMA volta e la TERZA volta che si esegue la procedura $ALFA$ (variabile M). Nel gioco, così come è stato proposto, questo valore è 5 ovvero il numero di lettere della parola **MAGIC** e della parola **TRICK**.

A questo punto, con N, S1, M fissati, il C64 va a vedere cosa capita al variare del numero di carte della SECONDA volta che entra in gioco la procedura ALFA perché è questo il valore che dipende dalla carta scelta dallo spettatore per il secondo spelling.

La posizione della “carta segreta” viene travasata nella variabile SW e ... si parte:

Nel ciclo che inizia alla 60, per K che va da 1 a N:

* A diviene la posizione della carta in S1 dopo la procedura ALFA(M)

* Il valore di A diviene la nuova S1, perché quella è la nuova posizione di partenza, prima del secondo spelling

* B diviene la posizione della “carta segreta” dopo il secondo spelling; qui il numero di carte spostate è K

* Il valore di B diviene la nuova S1, perché quella è la nuova posizione di partenza, prima del terzo spelling

* C diviene la posizione della “carta segreta” dopo il terzo spelling.

Alla 80 vengono stampati il valore di K (numero di carte utilizzate per il secondo mazzetto) e la posizione della “carta segreta” dopo ciascuna delle tre procedure ALFA. Alla 90 viene ripristinato il valore iniziale S1 per il prossimo K.

Il meccanismo risulta più chiaro se lanciamo il programma con i valori del gioco di prestigio così come proposto da Matt McGurk:

NUMERO TOTALE DELLE CARTE è 15

POSIZIONE CARTA SEGRETA è 1 (la prima dal basso)

N. CARTE MAZZETTO è 5 (numero di carte utilizzato per il primo – MAGIC - e terzo – TRICK – spelling).

Ecco il risultato:

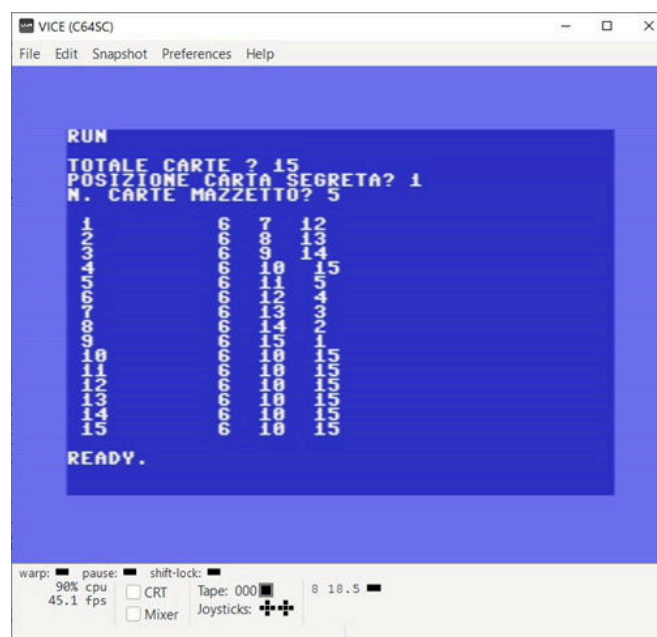


Fig. 1 - Schermata numero 1





Guardiamo, ad esempio, i valori della settima riga: 6, 13, 3 che si ottengono per $K=7$. Il significato è questo: se la posizione iniziale della “carta segreta” è la prima dal basso, dopo la prima procedura ALFA (spelling di MAGIC), la “carta segreta” passa in posizione 6, dopo la procedura ALFA con spostamento di $K=7$ carte andrà in posizione 13 e infine, dopo una nuova procedura ALFA (spelling di TRICK) con 5 carte, si troverà in posizione 3, ovvero sarà la terza carta dal basso.

Potete provare: prendete un mazzetto di 15 carte con il Joker in fondo al mazzo. Eseguite tre volte la procedura ALFA con 5, 7, 5 carte e vedrete che il Joker finirà nella terza posizione dal basso. Se invece eseguite la procedura ALFA con la sequenza 5, 2, 5 troverete il Joker nella tredicesima posizione a partire dal basso come previsto dalla seconda riga della schermata.

Ora fissate l'attenzione sulle ultime 5 righe della SCHERMATA 1: il valore finale è 15 qualunque sia il valore di K da 10 a 15. Questo significa che, se eseguite la procedura ALFA con la sequenza 5, K , 5 otterrete sempre la carta segreta in cima al mazzo (che corrisponde a 15 dal basso) se K è 10 o 11 o 12 o 13 o 14 oppure 15.

Certo, potreste chiedere allo spettatore di scegliere un numero tra 10 e 15 per la seconda procedura ALFA, ma il trucco proposto da Matt è più sottile: chiede allo spettatore di scegliere una delle 52 carte del mazzo e poi ... esegue lo spelling.

Apparentemente lo spettatore ha molte più possibilità di scelta, ma ... vediamo cosa può succedere.

Elenchiamo i possibili valori delle carte accompagnati dal numero di lettere che esprime il valore in lingua inglese. Avremo (A,3) perché A=ACE è formato da 3 lettere; (2,3) in quanto 2=TWO è formato da 3 lettere e così via:

(A,3), (2,3), (3,5), (4,4), (5,4), (6,3), (7,5), (8,5), (9,4), (10,3), (J,4), (Q,5), (K,4).

Il numero di lettere va da un minimo di 3 (ACE, TWO, SIX, TEN) ad un Massimo di 5 lettere (THREE, SEVEN, EIGHT, QUEEN).

Per quanto riguarda i semi, abbiamo HEARTS di 6 lettere, DIAMONDS di 8 lettere, CLUBS di 5 lettere e SPADES di 6 lettere: un minimo di 5 lettere e un massimo di 8.

Quando si esegue lo spelling delle carte, avremo anche due lettere per la preposizione OF, così. Ad esempio, con FIVE OF SPADES, si ottengono, complessivamente $4+2+6=12$ lettere.

Per quanto detto prima, qualunque sia la carta scelta dallo spettatore, lo spelling della seconda procedura ALFA andrà da un minimo di 10 lettere (ad esempio SIX OF CLUBS) ad un massimo di 15 (ad esempio con QUEEN OF DIAMONDS) che sono proprio i valori che portano tutti alla stessa posizione finale della “carta segreta”.

È il momento per un piccolo “Intermezzo”.

Guardando la SCHERMATA 1, può sembrare strano che la posizione finale (15) non dipenda dalla procedura ALFA(K) per ben sei diversi valori di K (da 10 a 15). In fondo lo spostamento di carte è diverso e questo dovrebbe, in qualche modo, modificare la posizione finale della nostra “carta segreta”.

Naturalmente il prestigiatore punta proprio su questo aspetto: la manovra ha tutta l'aria di mescolare il mazzo in modo del tutto casuale.

In realtà, nella procedura ALFA, quando la “carta segreta” è già stata calata sul nuovo mazzetto, poco importa che le successive vengano sovrapposte per via dello spelling o lasciate cadere in blocco: la nuova posizione della carta segreta non cambierà (se fate qualche esperimento con le carte ve ne rendete conto facilmente).

Ma torniamo al gioco di prestigio: funzionerà anche se proposto in lingua italiana?

Per quanto riguarda i valori delle 13 carte abbiamo che ASSO è di 4 lettere, DUE è di 3 lettere ecc. Questa volta il valore della carta con il minor numero di lettere è il RE che è formato da sole 2 lettere mentre il maggior numero di lettere si ottiene per il QUATTRO: ben 7 lettere (il FANTE origina 5 lettere o 4 lettere se indicato come JACK; la DONNA fornisce 5 lettere oppure 6 lettere se chiamata REGINA. Queste scelte non hanno importanza perché non modificano il valore minimo e massimo del numero di lettere). Per quanto riguarda i semi abbiamo FIORI: 5 lettere, QUADRI: 6 lettere, PICCHE: 6 lettere e CUORI: 5 lettere. Anche in italiano occorre aggiungere esattamente 2 lettere per la preposizione DI per cui, qualunque sia la carta scelta da Francesco, si andrà da un minimo di 9 lettere (RE DI FIORI oppure RE DI CUORI) ad un massimo di 15 (QUATTRO DI QUADRI oppure QUATTRO DI PICCHE).

Ohi ohi, siamo nei guai. La SCHERMATA 1 ci dice che per $K=9$ la posizione finale non sarà in cima al mazzo.





Che fare? Il programma MCGURK ci aiuta a trovare altre soluzioni. Proviamo a modificare il numero di carte del primo e del terzo mazzetto e portiamo a 6 questo valore. Ecco il risultato:

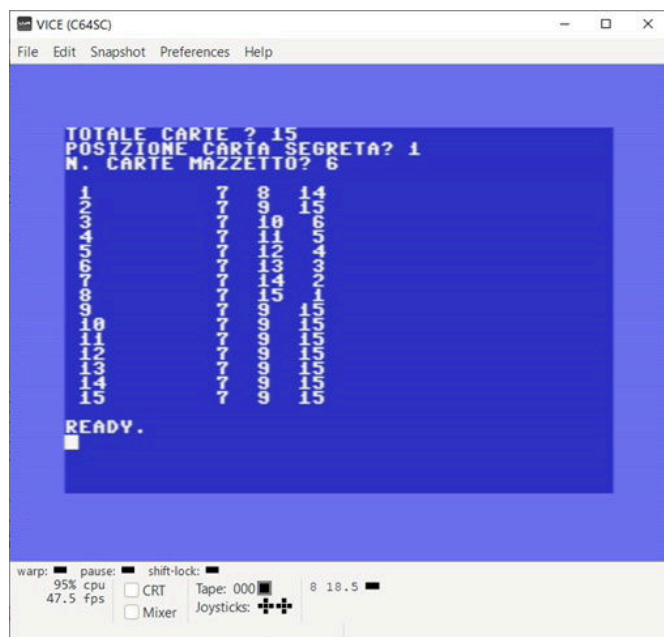


Fig. 2 - Schermata numero 2

Ce l'abbiamo fatta! Per tutti i K da 9 a 15 il terzo valore è 15. Il gioco funzionerà anche in italiano se anziché usare la coppia (MAGIC, TRICK) ne utilizzeremo una con parole di 6 lettere, ad esempio (PAROLA, MAGICA).

Finora abbiamo utilizzato il programma MCGURK per analizzare il gioco proposto nel video. Possiamo però lanciarlo con nuovi valori di N, S1 e M e inventarci così nuovi giochi di prestigio!

Diamo il RUN e procediamo, ad esempio, con:

TOTALE CARTE: 20

POSIZIONE CARTA SEGRETA: 10

N. CARTE MAZZETTO: 6

Il nostro fidato Commodore ci informa che la carta segreta finirà in posizione 11 per tutti i valori di K compresi tra 5 e 20. Ecco allora un gioco di prestigio nuovo di zecca (funziona davvero, provate!):

1. Dite a Francesco, il nostro spettatore, che vi serve un mazzo di 20 carte. Prendete le prime 20 da un mazzo ben mescolato e approfittate per dare un'occhiata alla carta in fondo al mazzo. Memorizzatela: sarà la vostra previsione. Nei suoi video Matt mostra anche come si può mescolare le carte e tagliare il mazzo senza modificare la posizione

dell'ultima carta, queste azioni abbelliscono il gioco, ma non sono indispensabili.

2. Dite a Francesco che rimescolerete il mazzo in vari modi e, tanto per cominciare, create un mazzetto di 9 carte prendendo tre gruppi di 3 carte alla volta a partire dall'alto e lasciate poi cadere in blocco il resto del mazzo sopra il mazzetto delle 9 carte. Questo porterà la carta che avete memorizzato in posizione 10 (a partire dal fondo).

3. Prendete un foglietto e, in gran segreto, scrivete la vostra previsione. Il foglietto viene messo da parte e non potrà più essere modificato.

4. Eseguite la procedura ALFA(6): farete lo spelling del termine PAROLA creando un mazzetto di 6 carte e sovrapponetevi in blocco le rimanenti.

5. A questo punto chiedete a Francesco il "nome e cognome" di una persona a sua scelta ed eseguite la procedura ALFA, Se, ad esempio, Francesco dice "IVY FIORENTINI", conterete 13 carte per formare il nuovo mazzetto e lascerete cadere in blocco le carte rimanenti. Qui avete ampie possibilità di scelta in quanto è sufficiente che il valore associato alla manovra sia tra 5 e 20. Ad esempio, dopo aver formato il mazzetto delle 13 carte, potreste dire a Francesco: "Se vuoi posso aggiungere al tuo mazzetto 0, 1, 2 o 3 carte. Cosa vuoi che faccia?" Questo per dare l'impressione di rimescolare ulteriormente il mazzo.

6. Come al punto 4, eseguite la procedura ALFA(6) utilizzando il termine MAGICA.

7. A questo punto, la "carta segreta" si trova in posizione 11 a partire dal basso, ovvero in 10 posizione dall'alto. Potete procedere come meglio credete scartando nove carte e lasciando la carta in cima oppure facendo lo spelling di una frase di nove lettere, ad esempio "eccola qui".

8. Prima di girare la carta, chiedete a Francesco di leggere cosa è scritto sul foglietto e ... il gioco è fatto.

Naturalmente, al punto 5, potete sostituire il "nome e cognome" con qualcosa di alternativo; inoltre, nel video, Matt suggerisce alcune manovre diversive per rendere il gioco più interessante (iscrivetevi al suo canale: Matt è davvero simpatico!).

Dite la verità, non vi aspettavate che il C64 vi avrebbe aiutato a diventare dei maghi!



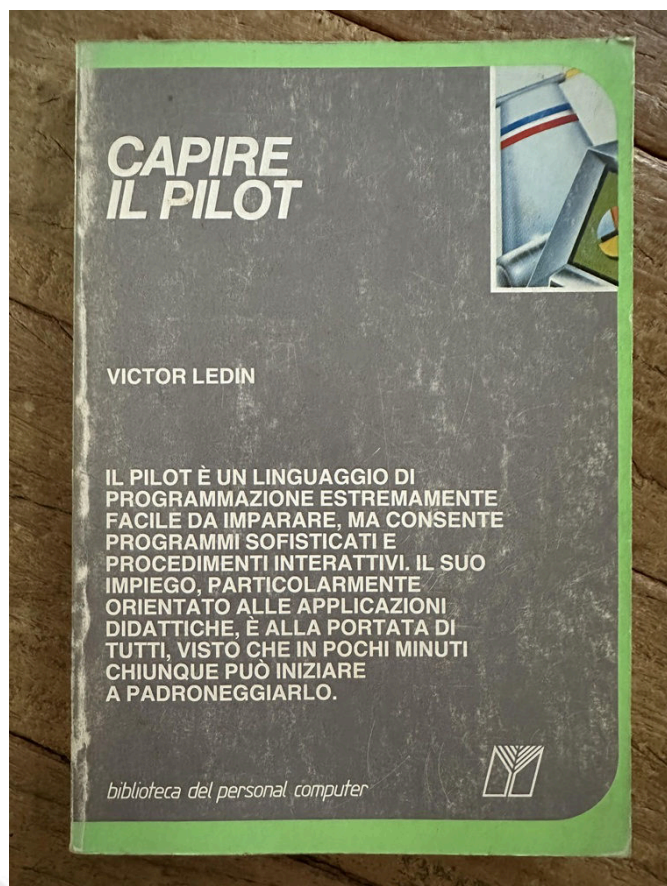


Super Turtle Pilot - storia di un avventuroso recupero

di Francesco Fiorentini

È cominciato tutto da questo libro: **Capire il Pilot** di **Victor Ledin**. Sono appassionato collezionista di libri dedicati all'informatica degli anni precedenti il 2000 e questo libro era decisamente uno che meritava un posto nella mia ormai ben nutrita collezione.

Ho già altri libri di Franco Muzzio Editore e devo ammettere che, prima di riceverlo, mi aspettavo un libro un po' più grande del formato quasi "tascabile" di questo testo.



IL PILOT

Il Pilot, acronimo di **Programmed Inquiry, Learning or Teaching** (traducibile in italiano come: Domanda, Apprendimento o Insegnamento Programmato) è un linguaggio ideato nel 1968 da John Starkweather del Medical Center dell'Università della California a San Francisco. Essendo un linguaggio di natura non algebrica, risulta più semplice da apprendere di molti altri linguaggi dell'epoca come il Fortran, il Basic ed il Pascal. La sua struttura lineare, sebbene differente da quello degli altri linguaggi, risulta ideale per un primo approccio al computer da parte dei neofiti della programmazione.

Ovviamente, il linguaggio Pilot è a disposizione in molte delle nostre retro-macchine ed il Commodore 64 non fa certo eccezione. Spinto dalla curiosità mi sono quindi messo alla ricerca delle versioni a disposizione per l'8 bit di casa Commodore.

Ecco una lista ripresa dal sito: <https://telarity.com/~dan/cbm/languages.html#PILOT> - (filtrata per C64)

PILOT [C64] (1983, CBM)

Tutte le funzionalità di Common PILOT, oltre alle interfacce per lettori di videodischi e nastri. Disponibile per l'acquisto presso The Computer Bargain Store.

PILOT [C64,VIC,PET] (Michael Tinglof, COMPUTE! Dec. 1982, Vol.4,No.12,Iss.31)

Un interprete PILOT essenziale scritto in BASIC, quindi non molto veloce. Probabilmente funziona anche su C128 e 264. È stato inoltre portato su KIM-1.

PILOT II [C64] (1983, Tamarack Software)

Include 25 comandi di editing, 19 comandi per la grafica a tartaruga e 23 comandi di programmazione con diversi sottocomandi e opzioni. Supporta grafica ad alta e bassa risoluzione e sprite, e include un editor di sprite.

Vanilla PILOT [C64,PET] (1982, Tamarack Software, Brøderbund)

Supporta i comandi "turtle graphics", "trova/sostituisci", e "trace and dump" utili per system diagnostics e debugging.

Come da copione, il Commodore 64 non delude mai, pensate ad un software e il nostro amato 8 bit sicuramente ne avrà avuto una implementazione. Incredibile.

Per non lasciare comunque nulla al caso ho deciso di chiedere all'intelligenza artificiale di DeepSeek di segnalarmi altre versioni di Pilot per il C64 e puntualmente l'IA mi ha indicato delle varianti non presenti nella lista sopra:

- **Commodore PILOT (1982/1983):** La versione ufficiale venduta dalla stessa Commodore su dischetto. Era accompagnata da un manuale e includeva utilità per la creazione di sprite e la sintesi del suono, mettendo in evidenza le sue capacità che andavano ben oltre il semplice testo.





- **Super Turtle PILOT (1987)**: Una versione potente pubblicata sulla rivista COMPUTE!. Ha migliorato significativamente il linguaggio aggiungendo comandi dedicati per la grafica a tartaruga, la manipolazione degli sprite, la musica e persino la gestione dei file su disco. Forse la sua caratteristica più notevole era la possibilità di incorporare comandi BASIC direttamente all'interno di un programma PILOT, consentendo loro di condividere variabili e combinare i punti di forza di entrambi i linguaggi.

- **Pilot-Plus/64**: un altro interprete commerciale lodato per il suo ampio e utile set di utilità di programmazione. Includeva comandi come RENUMBER, FIND, AUTO (per la numerazione automatica delle righe) e PLIST (per stampare l'elenco di un programma), che erano di gran lunga più avanzati rispetto all'editor BASIC standard dell'epoca.

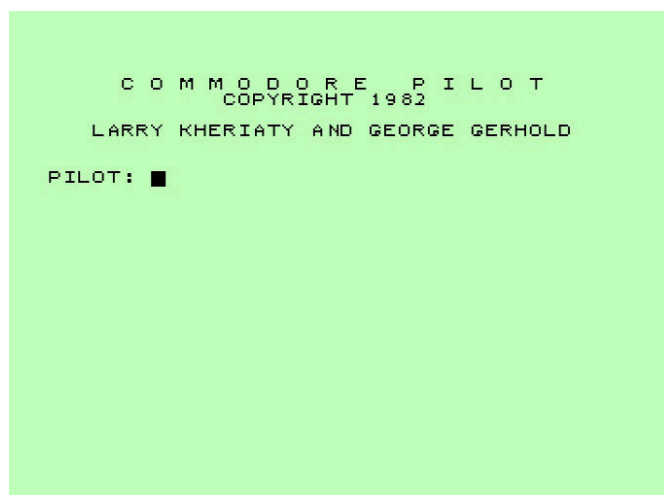


Fig. 2 - Commodore PILOT del 1982

Come sapete, per estrarre il massimo dal Commodore 64 occorre accedere direttamente alla sua memoria (comandi PEEK e POKE del Basic) e quindi ho fatto una ulteriore domanda a Deepseek per scoprire quali versioni di Pilot avessero questa possibilità'.

La risposta di Deepseek mi ha immediatamente indicato il Super Turtle Pilot come l'unico in grado di incorporare comandi Basic come PEEK e POKE; DeepSeek rincara la dose sul Commodore Pilot adducendo che il linguaggio era stato progettato per il dialogo e l'insegnamento, non per la programmazione di sistemi. Non disponeva di comandi integrati come PEEK o POKE (probabilmente adotta un altro meccanismo per accedere alla memoria, ma non ho ancora indagato; materiale per un prossimo articolo... :-)).

A questo punto il nerd che vive dentro di me, nemmeno

tanto sopito, si è subito incuriosito e ha pensato bene di andare a cercare questo fatidico Super Turtle Pilot.

Ebbene, dopo un po' di ricerche e' apparso subito chiaro che il Super Turtle Pilot non fosse disponibile al download da nessuna parte. Sacrilegio!

Compute!

Nella risposta di DeepSeek pero' c'era un importante indizio. Il Super Turtle Pilot (lo abbrevierò in STP) era un software divulgato dalla rivista Compute!, una rivista americana pubblicata dal 1979 al 1994 e contenente centinaia di listati di notevole interesse per gli utenti home computer degli anni '80.

Quindi non dovevo far altro che cercare il numero che conteneva il listato dell'STP e, con i potenti mezzi oggi a nostra disposizione, recuperare l'antico vaso perduto (ah no, questa era un'altra storia...).

Trovare la rivista che conteneva il listato e' stato semplice; dopo una veloce ricerca su Google e' subito saltato fuori il link ad Archive.org del **numero 89 di Ottobre 1987**.

A pagina 74 fa bella mostra di se' l'articolo Super Turtle Pilot ed a pagina 77 comincia il listato.

Primo ostacolo: il listato e' sicuramente in Assembly, pero' in formato esadecimale, pronto per essere dato in pasto ad una utility distribuita in un numero precedente di Compute! dal nome di **MLX**, un programma di immissione di codice in linguaggio macchina che permette di scrivere il listato direttamente nelle locazioni memoria del

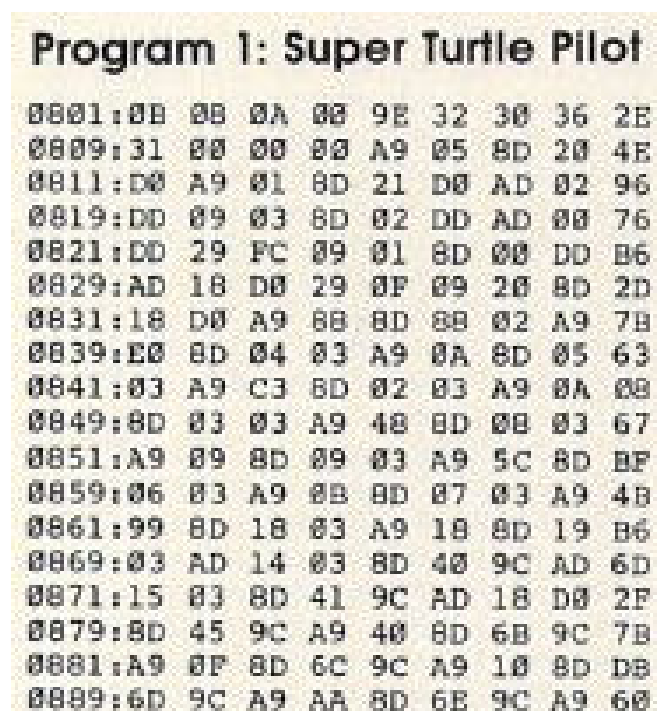


Fig. 3 - Il listato di STP su Compute!





Commodore 64 controllando al tempo stesso che non ci siano errori nei dati immessi.

Ovviamente recuperare il MLX e poi copiare manualmente le righe del codice (sono 781...) era fuori di discussione. Decido comunque di recuperare il codice, intanto da averlo in formato elettronico, pronto per ogni evenienza.

Faccio un po' di screenshot e li passo a DeepSeek per fargli estrarre il listato e, in pochi minuti, ho il codice su Scite. Decido quindi di chiamare **Marco Pistorio**, grande esperto di Assembly del C64 e decidere con lui una strategia.

Le righe DATA

Dopo una breve chiacchierata decidiamo che possiamo tentare di creare un programmino in Basic che, utilizzando delle righe DATA vada a scrivere in memoria il codice. Sapevamo l'indirizzo di partenza e quello finale, quindi avevamo le informazioni per farlo.

Ma ovviamente non e' cosi' semplice. Dopo un attento esame risulta chiaro che i nove byte che compongo ogni riga sono 8 byte di codice ed uno di controllo; inoltre il listato di Compute! deve essere convertito da esadecimale a decimale.

La rimozione del nono byte e' piuttosto semplice, Scite ha la possibilita' di selezionare non solo in orizzontale, ma anche in verticale, quindi con pochi clic rimuovo il byte di controllo da ogni riga.

Per la trasformazione da esadecimale a decimale decido ancora di avvalermi dell'aiuto dell'AI. Chiedo quindi a DeepSeek di convertire tutti i byte del listato e di inserirli in una riga DATA (partendo da 1000) separandoli con una virgola. DeepSeek in pochi istanti compie la magia. Abbiamo le righe DATA da inserire in un programma Basic per essere scritte in memoria.

Creiamo il programmino, assicurandoci di spostare la memoria Basic e di lasciare abbastanza spazio in memoria per il codice e diamo quindi il RUN... E riceviamo un messaggio di errore. Il programmino si blocca incasinando la memoria Basic dell'emulatore.

Le mie conoscenze tecniche del Commodore 64 non mi aiutano, ma quelle di Marco invece arrivano a capire il problema. Il programma deve essere eseguito in due step, prima la parte che libera la memoria e poi la parte che va a scrivere il codice. Marco, riesce a eseguire il tutto e... Di nuovo un altro fallimento. Una volta "iniettato" il codice in memoria, il programma parte ma non fa niente a parte

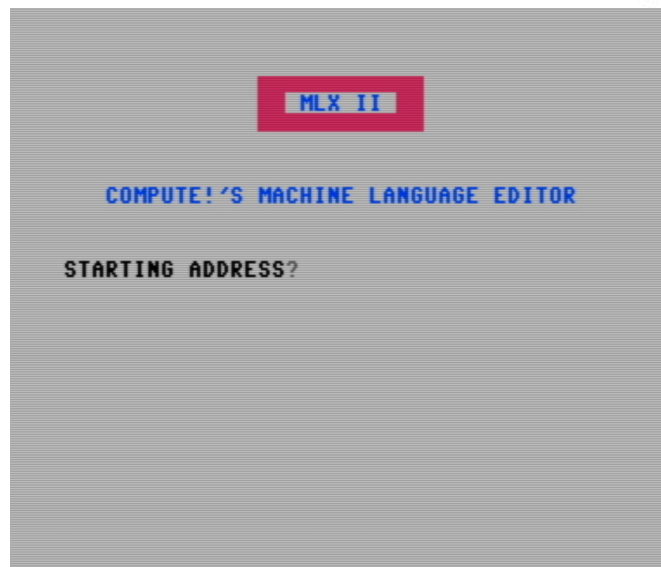


Fig. 4 - MLX II - Compute! Machine Language Editor

colorare lo sfondo di verde.

Marco esamina il codice Assembly in memoria nell'emulatore, e dopo alcune righe risulta evidente che alcune istruzioni sono errate!

Il fatto che il codice di Compute! sia errato non ci sfiora neanche lontanamente, piu' probabile che DeepSeek abbia frainteso qualche byte durante il recupero del codice.

La conferma ci viene non appena individuiamo una delle prime righe incriminate, un byte della riga recuperato da DeepSeek e' diverso da quello stampato nella rivista...

Ci arrendiamo quindi? Giammai! Super Turtle Pilot sembra essere un programma interessante. Leggendo l'articolo su Compute! si intuisce chiaramente che si tratti di un'estensione del Basic V2 e non di un linguaggio a se stante; nonostante questo pero' STP non e' presente da nessun parte e sarebbe un peccato abbandonarlo all'oblio. Va recuperato, ma al tempo stesso non posso controllare 6248 byte, uno per uno, confrontando la rivista ed il mio listato... Decisione drastica: recupero MLX ed immetto tutte le righe tramite copia e incolla nel programma, sperando che il byte di controllo faccia il suo lavoro.

MLX - Compute! Machine Language Editor

Ero gia' pronto a mettermi al lavoro per recuperare il codice di MLX quando mi viene un'idea! Perché non cercare questo programma su internet? Magari qualcuno, data l'importanza e la diffusione della rivista Compute!, ha gia' fatto il lavoro sporco. Una rapida ricerca e... Si'! MLX e' gia' disponibile! E vai!

Accedo al link fornitomi da Google e trovo MLX disponibile al download pronto per essere utilizzato sull'emulatore.





Ma, controllando meglio, questo e' MLX versione 1, nella rivista numero 89 di Compute! si fa riferimento alla versione II... Comincio a perdere le speranze, Super Turtle Pilot si difende con le unghie e con i denti. Pero' mi viene un dubbio, chiunque abbia recuperato la versione 1 magari ha fatto lo stesso con la versione successiva. Per fortuna il sito Gaming Alexandria ha una comoda funzione di ricerca: provo a cercare "MLX II" e... Si'! C'e' anche la versione successiva.

Non perdo neanche un minuto per scaricarla e lanciarla sull'emulatore. Funziona. Adesso non devo far altro che incollare tutto il mio codice e vedere dove si blocca...

Copio tutto il listato, lo incollo a su MLX e si blocca subito. Il tempo necessario a MLX per calcolare il byte di controllo e presentare la riga successiva, fa si' che MLX non accetti piu' di una riga per volta...

Fatto 30, facciamo 31!

Arrivato a questo punto, sento il profumo della vittoria. Decido stoicamente di imbarcarmi nell'impresa di copiare riga per riga il codice su MLX, sperando che il gioco valga la candela.

Sembra un lavoro lungo e tedioso, e lo e', ma dopo un paio di copia e incolla scopro come posso velocizzare leggermente il tutto. Con un doppio click sulla riga di Scite riesco a copiare l'intera riga, mentre con un singolo click del pulsante destro sul VICE riesco a incollarla su MLX. Il lavoro e' sempre tanto, ma con qualche ora di fatica me la posso cavare.

Ve la faccio breve, ci sono volute circa 3 ore a mezza, spezzate in due giorni, per incollare tutto il codice e correggere gli errori che di tanto in tanto MLX mi segnalava. Circa un 10% delle righe aveva almeno un byte errato. Grazie a MLX, che calcola il byte di controllo non soltanto a partire dagli 8 byte precedenti, ma tiene conto anche della locazione di memoria nella quale il codice deve essere scritto, adesso ho il codice Assembly di Super Turtle Plot in memoria. Non devo far altro che salvare il file ed eseguirlo.

Salvo il file, esco da MLX, riavvio VICE e lancio il file... EUREKA! La schermata del Super Turtle Pilot mi avverte che il programma e' in esecuzione e pronto ad accettare i miei comandi. L'antico vaso era stata salvato... Ah no, Super Turtle Pilot era finalmente stato recuperato e pronto per essere messo a disposizione di tutti sul nostro sito!



Fig. 5 - Super Turtle Pilot by Alan Poole

Cos'e' Super Turtle Pilot

STP e' una delle centinaia di estensioni Basic fiorite negli anni '80 e '90. E' particolarmente interessante per la peculiarita' di aggiungere comandi PILOT e Logo (Turtle) al Basic del Commodore 64. E' possibile combinare i comandi Basic ai nuovi comandi aggiunti dal STP ma con una limitazione. Non e' possibile inserire piu' di un comando per riga perche' il PILOT usa ":" (i due punti) come parte dei suoi comandi.

Vi rimando alla pagina del nostro sito: <https://www.retromagazine.net/super-turtle-pilot-commodore-64-assembly/> per un'approfondita analisi dei comandi del STP e per un esempio di programmazione.

Parleremo ancora di STP nei nostri articoli, ma per il momento e' tutto.

Link Utili

Commodore PILOT (1982)

- <https://www.commodoreserver.com/PublicDiskDetails.asp?DID=EE24139BC8494912B5800597C17A36AD&SnapshotID=1> (richiede registrazione)

Manuale Commodore PILOT

- https://www.retro-commodore.eu/c64-software/?rceu_page=2

Rivista Compute! - Numero: 89 - Ottobre 1987

- <https://archive.org/details/1987-10-compute-magazine>

MLX - Compute! Machine Language Editor

- <https://www.gamingalexandria.com/wp/2025/12/mlx-ii-v1/>

Super Turtle Pilot - download dal sito di RetroMagazine

- <https://www.retromagazine.net/super-turtle-pilot-commodore-64-assembly/>





Tutti i modi grafici non-standard del VIC-II

di Stefano Tognon aka "Ice00"

Introduzione

Sebbene il VIC-II fornisca nativamente 8 modalità grafiche (di cui 3 "invalide", quindi inutilizzabili se non per ottenere uno schermo totalmente nero), l'astuzia dei programmatori, grazie ad una conoscenza profonda del chip video acquisita nel corso del tempo, ha permesso di arrivare a più di 35 nuove varianti, rendendo tangibile ciò che sembrava irrealizzabile. Per comprendere come questo sia possibile serve conoscere alcuni principi base di funzionamento del chip impiegato nel Commodore 64.

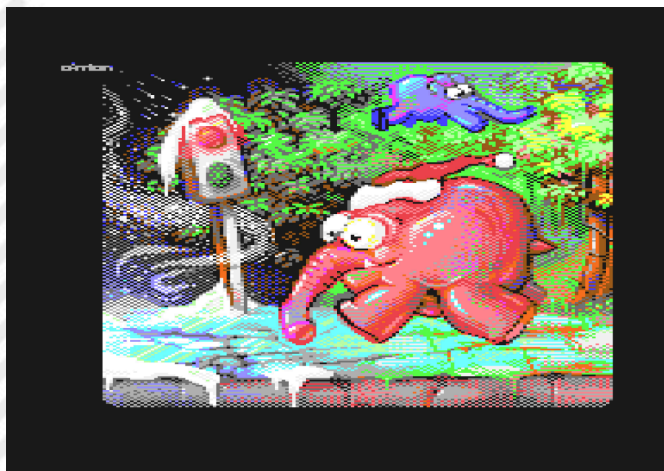
Il VIC-II disegna lo schermo pixel per pixel pilotando il segnale del raster (o pennello elettronico) della TV/monitor da sinistra a destra e dall'alto in basso, disegnando un bordo colorato attorno all'area grafica dello schermo. Nel momento in cui deve mostrare l'immagine grafica (che è sempre strutturata in blocchi da 8x8 pixel), il chip preleva i dati dal banco di memoria, interrompendo quindi l'attività del processore 6510, che quindi avrà a disposizione meno dei 63 cicli di clock (normalmente forniti da un sistema PAL) per linea a disposizione per eseguire codice.

Il VIC-II permette di generare un segnale di interrupt quando raggiunge una specifica linea dello schermo e, condizione essenziale ai fini di generare nuove modalità grafiche, il programmatore deve intervenire colmando in modo preciso il ritardo (che può andare da 0 a 7 cicli) con cui il processore riceve la notifica. A questo punto si

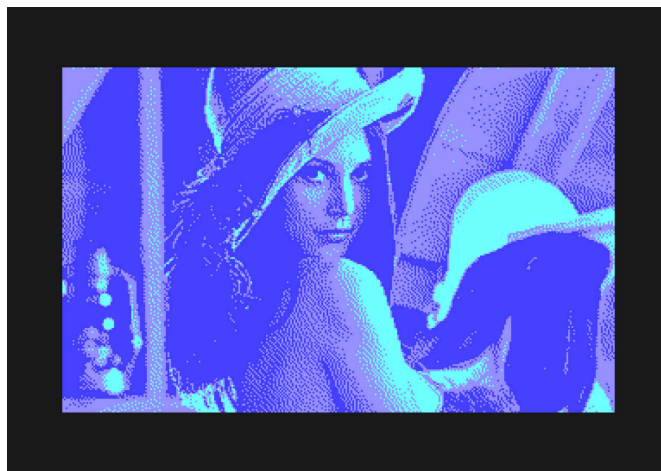
possono impiegare varie tecniche per ottenere questo risultato, come il "doppio interrupt" o il ricorso al timer contatempo del chip CIA, in modo da essere certi di colmare con precisione il ritardo generato. Ma il punto chiave è che possiamo sapere cosa sta eseguendo il processore ad una determinata linea in un preciso ciclo.

Inoltre, sebbene il VIC-II veda (cioè può accedere) solo 16KB della memoria (tranne per la RAM colore e la ROM caratteri che sono casi particolari), il processore centrale può indirizzarlo istantaneamente in 1 dei 4 banchi differenti dei 64KB disponibili, permettendo quindi di realizzare modi grafici differenti utilizzando metodi avanzati che in alcuni casi necessitano di ben 40KB di RAM per funzionare.

Per procedere con la descrizione dei modi grafici standard e non standard del VIC-II, ricordiamo che il VIC-II dispone di una palette di 16 colori fissi e che offre 8 sprite hardware indipendenti di risoluzione 24x21 pixel, monocromatici o multicolore (in questo caso i pixel "orizzontali" sono dimezzati in numero ma hanno grandezza doppia), le cui dimensioni orizzontali e verticali possono raddoppiare. Gli 8 sprite si possono anche riutilizzare più volte sullo schermo (tramite la tecnica del multiplexing) a patto di non averne più di 8 sulla stessa rasterline.



FLI (20KB): "Screw It Santa! I'm Gonna Be Rudolphant Anyway!" by Carrion (2013)



AH (17KB): "Example 3" by Algorithm, Algotech (2006)





NUFLI (26KB): "Forgotten" by Mufflon prg, Crest/Metalvotze (2010)

Modalità standard

Vediamo brevemente le modalità video standard offerte nativamente dal VIC-II.

SCM (Standard Character Mode): visualizza 40x25 caratteri a 2 colori (sfondo e colore carattere), scelti fra 256 caratteri predefiniti. Se si usano i caratteri della ROM di sistema, l'immagine derivante è chiamata PETSCII.

MCM (Multicolor Character Mode): visualizza 40x25 caratteri, monocromatici (2 colori) o multicolore (con 2 colori comuni aggiuntivi). Nel caso multicolore i pixel in orizzontale sono la metà e sono larghi il doppio e si possono usare solo i primi 8 colori della palette di sistema.

HBM (High-resolution Bitmap Mode): grafica bitmap (cioè formata da una mappa di bit che rappresentano i punti) da 320x200 pixel a 2 colori (sfondo e primo piano) per ogni blocco da 8x8 pixel.

MBM (Multicolor Bitmap Mode): grafica bitmap multicolore da 160x200 pixel a 4 colori. I pixel in orizzontale sono grandi il doppio, quindi il modo grafico è equivalente a quello bitmap 320x200 con aspect ratio di 2:1. Due colori sono comuni a tutti i blocchi 8x8, uno è indipendente e l'ultimo è il colore di sfondo.

ECM (Extended Background Color Mode): visualizza 40x25 caratteri a 2 colori. I caratteri sono scelti da soli 64 caratteri predefiniti. La differenza rispetto alla modalità carattere standard sta nel fatto che il colore di sfondo è scelto tra 4 possibili.



IXFLI (29KB): "Digital Magic demo" by Dane, Crest (2002)

Non-modalità

Può il VIC-II generare immagini anche con la grafica disattivata? Certo!

PRS (Permanent Raster Split): con questa tecnica si cambia il colore del bordo in ogni momento e grazie al fatto che usiamo una routine sincronizzata al singolo ciclo, l'immagine è stabile. Otteniamo quindi 384x272 pixel con celle di dimensione 32x1 con 3 colori differenti sulla stessa riga.

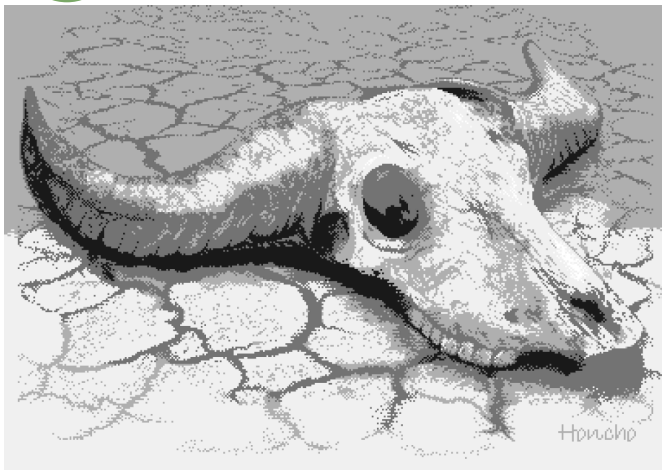
Apertura dei bordi

I primi modi grafici alternativi nascono quando si scopre come "aprire" i bordi, ovvero ingannare il VIC-II in modo che non disegni più il bordo, ma continui a mostrare gli sprite che sono posizionati sul bordo stesso. Il VIC-II infatti può operare a 38x24 celle invece che 40x25 in modo da poter far scorrere lo schermo di 8 pixel sia in orizzontale (asse X) che in verticale (asse Y), tecnica che tutti i giochi a scorrimento utilizzano.

Aprire il bordo superiore e inferiore è relativamente semplice: quando il VIC-II verifica se è giunto il momento di disegnare il bordo, si commuta da 25 a 24 righe (e poi viceversa) in modo da ingannare il chip video e non far scattare la condizione di disegno del bordo. Naturalmente occorre attuare questo trucco su due precise rasterline (una per il bordo superiore ed una per il bordo inferiore).

Con lo stesso principio si possono aprire i bordi laterali: si commuta da 40 a 38 colonne quando il VIC-II si trova in un preciso ciclo della rasterline (e da 38 a 40 prima





Full screen tramite HBM+Sprite (14KB): "COP26" by Honcho, Atlantis (2021)

che arrivi dalla parte opposta). Ciò implica che il trucco dev'essere attuato su ogni linea e per tutte le 200 linee, pertanto è d'obbligo mettere in campo una routine sincronizzata al singolo ciclo!

La tecnica di apertura dei bordi ci fornisce queste modalità alternative:

ESCOS (Expanded Screen CONstruction Set): tutto lo schermo, inclusi i bordi, è tutto tappezzato (grazie al multiplexing) da sprite espansi in orizzontale, in modalità multicolore e, a seconda dell'implementazione scelta, espansi o meno anche in verticale. La tecnica prevede che sullo schermo non arrivi mai i dati della mappa grafica, pur essendo perfettamente attiva, grazie ad una routine perfettamente sincronizzata al ciclo. Si ottiene quindi un'immagine con risoluzione 384x272 pixel a 4 colori, formata solo da tanti sprite opportunamente disegnati e con un aspect ratio di 4:1 (o 4:2 se li espandiamo anche in verticale).

FLESCOS (Full Screen All Borders Unexpanded Graphics - Flexible Line Sprite CONstruction Set): tutto lo schermo, compresi i bordi, è riempito con sprite monocromatici non espansi, risoluzione 384x272 pixel con 2 colori. Cosa? 8 sprite al massimo in orizzontale non espansi per 24 pixel ciascuno forniscono in totale solo 192 pixel, non certo i 384 pixel promessi! Qualcosa non torna... ma come insegnano gli illusionisti, il trucco c'è ma non si vede! Infatti, la tecnica usata per questo modo grafico prevede che si disegni una fila di sprite nella parte sinistra dello schermo per coprire i primi 192 pixel, poi che si lasci il



ESCOS (4KB/7KB): "Face" by OPM, Dutch USA-Team (1987)

"vuoto" che prosegue fino alla riga successiva dove si riprende a disegnare altri 8 sprite dopo aver raggiunto la metà dello schermo, quindi solo a destra. L'occhio dell'utente non si accorge dello spazio vuoto a destra e percepisce le due file di sprite come se fossero unite. Semplicemente geniale!

Full Screen: diverse modalità grafiche utilizzano la bitmap standard o la multicolor per lo schermo e anche gli sprite che coprono i bordi aperti e/o aggiunti per fornire maggiore dettaglio in qualche punto dell'immagine principale. Alcune modalità optano per lo schermo pieno, mentre altre aprono solo i bordi laterali o solo i bordi superiore ed inferiore.

Sprite per aggiungere colori

Gli sprite possono essere sapientemente usati anche per incrementare il numero di colori di un blocco 8x8, impiegandoli come sfondo (o anche in primo piano) e tappezzando in multiplexing tutta l'area dello schermo.

AH (Advanced Hires): si utilizzano sprite monocolori espansi in orizzontale posti dietro alla grafica bitmap ad alta definizione. Il risultato che si ottiene è un'immagine 320x200 pixel con 3 colori per cella 8x8 e aspect ratio di 1:1 (pur utilizzando sprite che hanno un aspect ratio di 2:1).

SH (Super Hires): si sacrifica la dimensione dell'immagine in modalità alta definizione, portandola a soli 96x200 pixel ma piazzando gli 8 sprite a due a due sovrapposti in 4 colonne: otteniamo così 4 colori per blocco 8x8 (con





2 colori che essendo dello sprite coprono un'area di 24x21 pixel). Per semplicità implementativa, spesso l'immagine è ridotta a 96x167 pixel sfruttando quindi solo 7 cambi di righe di sprite.

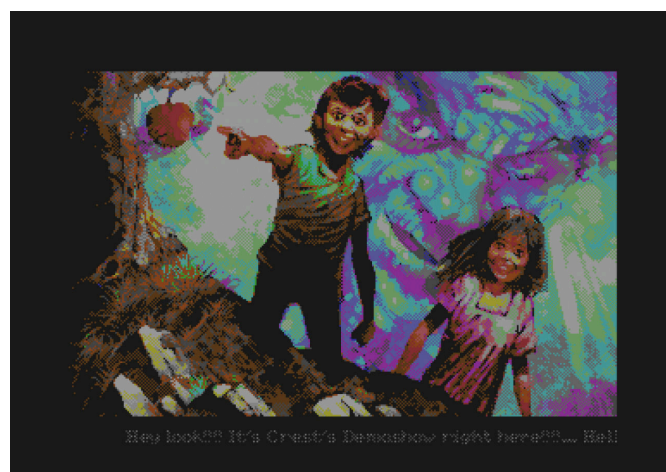
MUCSU Hires (MULTiColor Sprite Underlay Hires): in questa modalità s'impiegano alcuni sprite multicolore espansi in orizzontale sotto ad una immagine in modalità alta risoluzione (in pratica si tratta della versione multicolore equivalente alla modalità AH vista in precedenza).

Più colori per blocco

Una delle innovazioni più importanti è la tecnica chiamata Flexible Line Interpretation (FLI). Il VIC-II legge i dati grafici una linea ogni 8, ma invece di inibire la lettura come avviene nella modalità ESCOS, la si impone ad ogni rasterline (sempre mediante una routine che opera precisamente al singolo ciclo). Il risultato che otteniamo è la possibilità di variare i colori in modo indipendente riga per riga, dal momento che il VIC-II avrà nuovi dati da prelevare perchè li legge su ogni linea.

In questa modalità rimane poco tempo alla CPU (perché bloccata dal VIC-II impegnato a leggere i dati) ed il timing è talmente risicato che i primi 24 pixel non si ottengono dai dati grafici letti come al solito, ma da valori sempre pari a \$FF. Ciò porta alla visualizzazione, ad esempio, di un colore fisso, ma si può rimediare nascondendolo, eventualmente ricorrendo alla copertura con sprite che abbiano lo stesso colore dello sfondo.

La tecnica FLI consente di utilizzare le seguenti modalità:



ASSLACE (23KB): "Don't meet Crest"
by Crest (2004)

FLI (Flexible Line Interpretation): la tecnica di base è usata con la grafica bitmap multicolore e si ottiene un'area di 296x200 pixel con la possibilità di variare per ogni riga i 2 colori multicolore, perciò potenzialmente si possono usare tutti e 16 i colori nel blocco 8x8.

AFLI (Advanced Flexible Line Interpretation): di base si utilizza la grafica bitmap standard, perciò si può variare sia il colore di sfondo sia il principale su ogni riga del blocco arrivando anche in questo caso a potenzialmente 16 colori rappresentati in contemporanea.

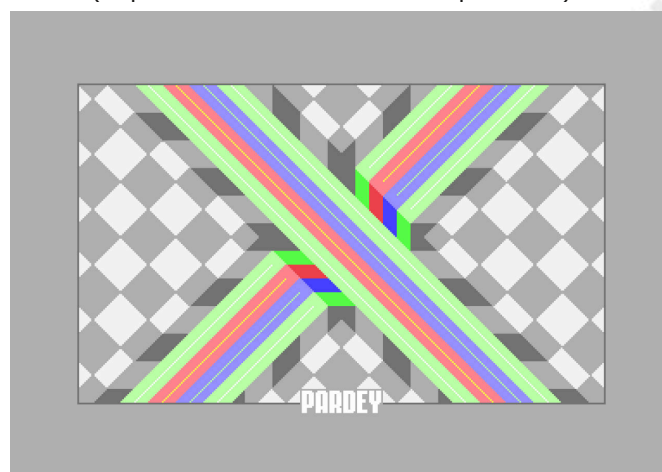
UFLI (Underlayed Flexible Line Interpretation): questa tecnica utilizza uno strato di sprite come sottofondo per dare ancora più colori alla modalità FLI (quindi si ottiene un'immagine in modalità AH ma con la tecnica FLI).

NUFLI (New Underlayed Flexible Line Interpretation): l'impiego di questa tecnica estende ulteriormente la modalità UFLI mediante l'uso di sprite che compensano l'area del bug FLI portando la risoluzione effettiva a 320x200.

XFLI (eXtended Flexible Line Interpretation): utilizza uno strato di 8 sprite multiplexato senza espansione in modo mirato, dunque è in grado di generare immagini di risoluzione ridotta a 192x168 pixel.

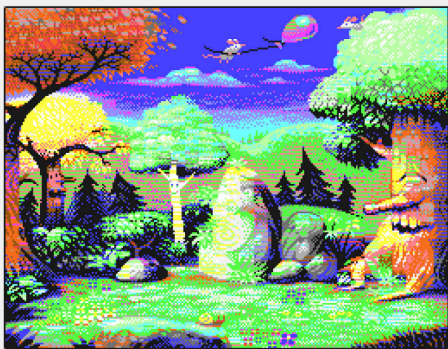
MUFLI (Multicolor Underlayed Flexible Line Interpretation): in questa modalità si utilizzano sprite multicolore come strato sottostante ad una immagine FLI (in pratica la versione con sprite multicolore della UFLI).

SHFLI (Super Hires Flexible Line Interpretation): ad una



ECM (2,4KB): "Xpardey"
by AgentE (2024)





AFLI (19KB): "Test Flight" by Mermaid, Genesis Project/Vision (2016)

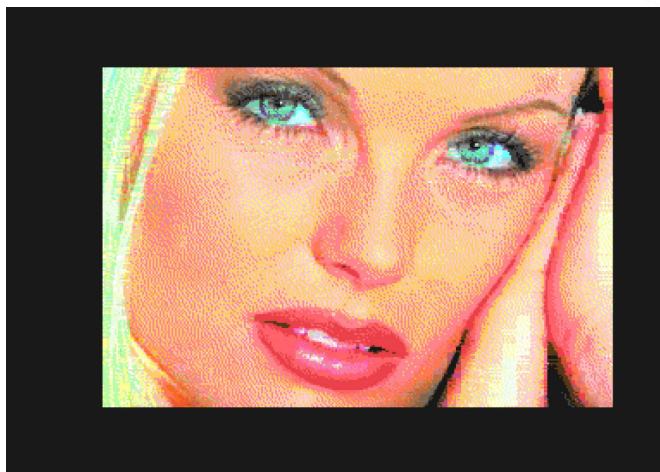
immagine Super Hires viene applicata la tecnica FLI, ottenendo quindi un aumento nei colori utilizzabili, dal momento che si può cambiare quello principale ad ogni riga (la risoluzione rimane confinata a 96x167).

SHFXL (Super Hires Flexible Line Interpretation eXtra Large, detta anche SOH-FLI): estende la modalità SH portandola a 144x168 pixel perché si usa una fila di 6 sprite multiplexati sottostanti. La dimensione maggiore si ottiene a discapito di un colore della fila di sprite rimossa.

MUCSU-FLI (MULTiColor Sprite Underlay Flexible Line Interpretation): qui si utilizza uno strato da 5 sprite multiplexati multicolore con dimensione orizzontale raddoppiata (240x160) sotto ad una grafica alta definizione da 272x160 pixel (la zona periferica risulta meno dettagliata, pertanto va sapientemente realizzata l'immagine in modo da tenere in considerazione questo effetto collaterale).

Interlacciamento

Sicuramente pochi si ricorderanno delle modalità grafiche interlacciate che i vecchi monitor VGA per PC avevano al fine di aumentare la risoluzione grafica senza aumentare il numero di pixel. Non serve avere un hardware apposito per riprodurre questa tecnica perché si può programmare il VIC-II per applicare lo stesso principio. Quando proiettiamo una immagine sulla TV e al frame successivo ne applichiamo una seconda che differisce (in modo opportuno) dalla prima, la persistenza della immagine sulla retina fornisce la percezione di un numero superiore di colori o di una risoluzione maggiore (anche se sarebbe meglio chiamare questo effetto con l'espressione "aspect ratio dei pixel



IAFLI (27KB): "Hiflint2-diff example" by MixCol Hi-Fli Converter, Algothech (2006)

ridotto").

Il modo più semplice ed immediato per ottenere l'interlacciamento è quello di mostrare un'immagine nel primo frame e poi di traslare di un pixel a destra l'immagine nel secondo frame. Ciò che si ottiene è la percezione di ben 128 colori teorici rispetto ai 16 della palette (anche se più realisticamente si arriva ad un massimo di 70, togliendo i colori troppo simili). Se prendiamo una immagine multicolore, i pixel in orizzontale sono grandi il doppio rispetto alla dimensione verticale, per cui quando si trasla di un pixel, metà dei pixel multicolore si sovrappongono all'altra metà, ossia ciascun pixel finisce su quello a destra. Ecco dunque che l'aspect ratio da 2:1 diventa 1:1 con una risoluzione reale effettiva sempre di 320x200.

Se applichiamo lo stesso trucco nella direzione verticale, otteniamo proprio l'effetto che realizzavano le modalità interlacciate dei primi monitor: se prendiamo un'immagine 320x400 pixel e visualizziamo in un frame le righe pari e nel successivo le righe dispari, la fusione visiva sembra mostrarci in un'area di 320x200 quanto era contenuto nell'immagine originale.

Tra le modalità che utilizzano l'interlacciamento annoveriamo:

- MCI (Multicolor interlace): si alternano due immagini bitmap multicolore.
- IH (Interlaced Hires): si alternano due immagini bitmap ad alta risoluzione standard.
- IFLI (Interlaced Flexible Line Interpretation): si alternano due immagini FLI.





- UIFLI (Underlaid Interlaced Flexible Line Interpretation): si alternano due immagini UFLI.

- XIFLI (eXtended Interlaced Flexible Line Interpretation): si alternano due immagini XFLI.

- MUIFLI (Multicolor Underlaid Interlaced Flexible Line Interpretation): si alternano due immagini MUFLI.

- SHIFLI (SuperHires Interlace Flexible Line Interpretation): si alternano due immagini SHFLI.

- SHI (SuperHires Interlace): si alternano due immagini SH ottenendo sempre una risoluzione di 96x167 ma idealmente con una palette di 128 colori (con 10 colori max per ogni blocco 8x8).

- SHIFXL (Super HiRes Interlaced Flexible Line Interpretation eXtra Large): si alternano due immagini SHFXL.

- IAFLI (Interlaced Advanced Flexible Line Interpretation): si alternano due immagini AFLI

- HMFLI (HiRes Mode Flexible Line Interpretation): si alternano un'immagine FLI e una AFLI. Questo permette di percepire dettagli ad alta definizione pur mantenendo alto il numero di colori.

- MEGATEXT: sono 4 immagini in modalità testo interlacciate. Qui il flickering è molto più evidente degli altri metodi, sia per l'utilizzo di soli caratteri, sia per via della sovrapposizione di 4 immagini differenti.

- TRIFLI (anche detto TRILACE): si alternano tre immagini FLI. Nonostante il flicker sia potenzialmente maggiore degli altri metodi con 2 immagini, con l'uso di dithering opportuno si riesce a compensare abbastanza bene. I colori potenziali visibili per sovrapposizione raggiungono i 2048!

- MRFLI (Multi-Resolution Flexible Line Interpretation): alterna due immagini di cui le linee pari sono FLI e le dispari AFLI o viceversa.

- ECMI (Extended Background Color Mode Interlaced): usato solo nei demo, alterna due immagini ECM.

Interlacciamento avanzato

Il problema della modalità interlacciata è il flickering: l'immagine completa viene percepita dall'occhio umano con una frequenza di soli 25Hz nel caso di 2 immagini, per cui quando ci sono colori che hanno una grossa differenza cromatica, si ha l'impressione che l'immagine non sia stabile e sembra traballare.

ASSLACE è un metodo che mitiga questo problema: si utilizza una immagine MCI a cui si sovrappongono uno schermo pieno di sprite con pattern a buchi invertiti tra un frame e l'altro: la tonalità complessiva diventa più scura a causa del colore degli sprite sovrapposti che prevalgono, ma il flickering percepito si riduce in modo sensibile!

Metodi particolari

Rimangono ancora da classificare alcuni metodi particolari che sono stati utilizzati:

X-SHBM (X-Scroll High Bitmap Mode): l'immagine è una standard bitmap, ma ad ogni linea viene applicato uno shift nella direzione orizzontale (X) secondo un opportuno pattern ciclico (serve quindi una routine sincronizzata al singolo ciclo di macchina). L'effetto è di aumentare virtualmente i colori a disposizione di ogni blocco 8x8 grazie a questa sfasatura.

QCH (Quarter Char Hires): questa è una modalità FLI per

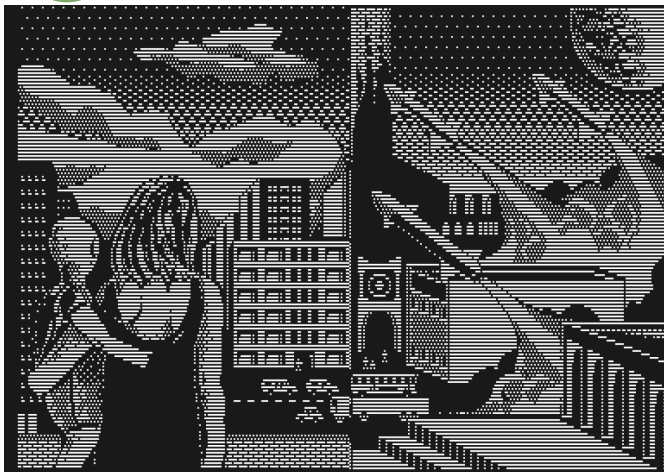


TRIFLI (40KB): "T3 scene 2"
by Algorithm, Algotech (2010)



X-SHBM (9KB): "Huhuu"
by Duce, Extend (2016)





FLESCOS (7KB): "13:37 Example 1" by The Sarge/Soya/Trident, Fairlight (2004)

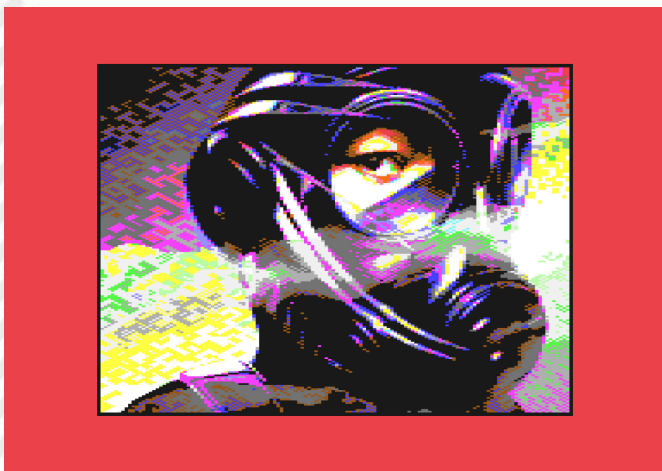
il bitmap ad alta definizione in cui le "bad line" sono eseguite una riga sì e una no, rendendo di fatto i colori indipendenti ogni 8x2 pixel.

HCB (Half Char Bitmap): si tratta di una modalità FLI per bitmap multicolore, in cui le "bad line" avvengono ogni 4 linee. Con i cicli macchina risparmiati si riesce anche a variare la memoria colore, che normalmente non viene toccata.

Sprite + Rasterbar: con questa tecnica si riesce a dare più colore ad una immagine composta da soli sprite se si cambia il colore dello sfondo come avveniva per il PRS. La routine è pertanto a singolo ciclo ed il processore 6510 viene fermato solo durante la lettura dei dati relativi agli sprite.

È tutto?

In realtà no, perché nella pratica esistono altri trucchi tecnici che possono dar luogo ad altri formati.



HCB (8KB): "Edge of Disgrace demo" by Booze Design (2008)



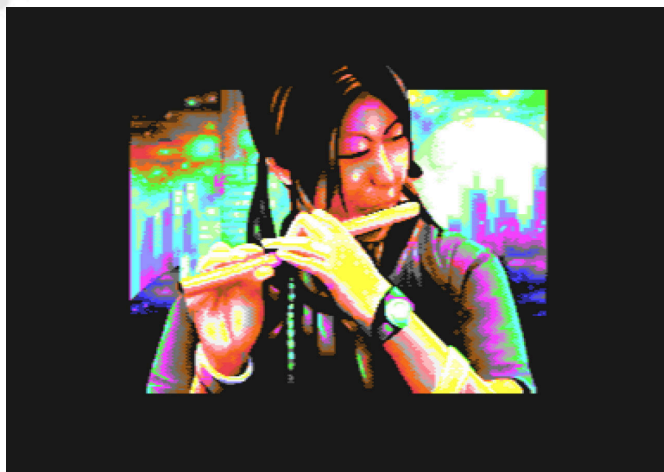
HBM (9KB): "Goblin Towers" by Mermaid, Genesis Project (2016)

Ad esempio, il cosiddetto "ghost byte" (\$3FFF) può essere usato per immettere grafica e farla visualizzare al VIC-II in diverse aree dello schermo, in modo analogo a come il valore \$FF finisce per coprire i pixel nell'area del bug FLI. Altro esempio sono i "gray dot", che si verificano in alcune particolari revisioni del chip VIC-II e producono dei punti grigi mentre si riscrive lo stesso colore che il VIC-II sta leggendo.

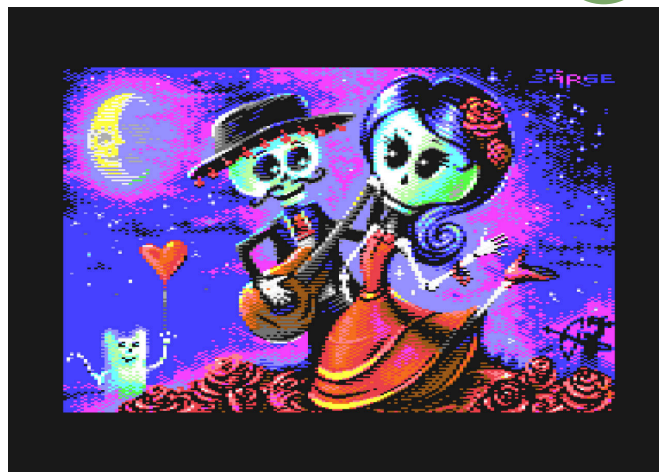
Indubbiamente ci sono molti altri metodi particolari che sono stati usati in alcuni demo, la fantasia dei coder è sterminata in questo senso, ma non essendo (ufficialmente) documentati, probabilmente aspettano ancora di essere rintracciati e studiati!

Per gli interessati ad approfondire tutti questi metodi in maniera più dettagliata, uscirà verosimilmente a Dicembre un libro intitolato "Pixels & Cycles: The Art of VIC-II Graphics". Si potrà quindi capire e apprezzare come avviene la sinergia tra la Cpu e il VIC-II per permettere a lato pratico di attuare tutti i meccanismi che sono stati accennati sopra. Ad oggi non si trovano pubblicazioni così mirate su questo argomento in letteratura, pertanto l'impronta data al libro è di poter essere comunque accessibile a tutti, in modo da far comprendere come avviene "la magia". Ma non mancano di certo i listati assembler dei punti cardine, essenziali per capire la logica dei cicli nella rasterline. Infine l'utente super evoluto potrà mettere mano alle routine di tutti i metodi trattati dato che saranno messi a disposizione tutti i sorgenti che implementano ogni singolo metodo.





IFLI (36KB): "Cosmic Fragility" by Jailbird, Booze Design (2006)



MBM (10KB): "Habanero" by The Sarge, Fairlight (2021)



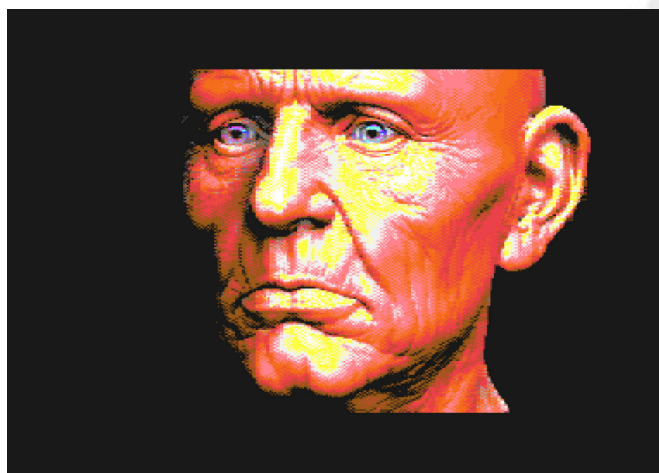
PETSCII (2KB): "Hermit" by redcrab, Genesis Project (2023)



SCM (4KB): "Oh, No! What Logo?!" by jmin (2024)



MCI (19KB): "Sweet Chili O'Mine" by Duce, Extend (2007)



HMFLI (28KB): "HMFLI" by Banshee/Peacemaker, Hitmen (2016)





Costruzione di mappe di gioco

di Marco Bergomi

Immaginate di voler realizzare un videogame con ambientazioni fantasy, un rogue game oppure un RPG insomma in tutti questi casi è necessario realizzare un'area di gioco.

Una delle parti più dispendiose di risorse è la costruzione di schermate fisse o mappe di gioco; quindi ho pensato ad un metodo per risparmiare memoria e semplificare il compito, lasciando al nostro C64 il compito di disegnare la schermata, seguendo semplici e particolari indicazioni. Ovviamente questo non è l'unico modo per ottenere lo stesso risultato, ma è quello che maggiormente si adatta ai miei scopi, ovvero realizzare un gioco a schermata fissa. Fissiamo gli step di questo progetto:

- 1) definire i codici numerici che verranno usati per comporre le schermate/mappe;
 - 2) associare i codici ai caratteri PETSCII;
 - 3) definire le dimensioni della schermata di gioco;
 - 4) disegnare la mappa con i codici scelti;
 - 5) scrivere la routine di decodifica codici e stampa a video.
- Pronti? Via!

Punto 1)

Iniziamo scegliendo i codici PETSCII da impiegare nella mappa:

0 = spazio: lo spazio vuoto dove non vi è nulla, per questo carattere ho deciso di usare il codice video 32 (tabella POKE), di colore nero (tabella colori = 0);

1 = erba: si potrebbe pensare ad un quadrato di colore verde per definire il prato, codice 160 di colore verde (13);

2 = acqua: come per l'erba ma di colore azzurro o blu codice 102, colore (14);

3 = le mura: questo potrebbe essere una porzione di muro, di pietra o mattoni, codice 104 di colore grigio (15);

4 = una chiave, oppure un pugnale, insomma un oggetto: codice 30, mentre per il colore possiamo scegliere un 10, rosso chiaro0

Si potrebbe continuare così per 256 elementi (come massimo), ovvero tutta la mappa caratteri, magari ridisegnando tutti i caratteri per ottenere un effetto a video migliore.

Punto 2)

Abbiamo già deciso quali caratteri della mappa verranno utilizzati per disegnare la schermata/mappa, vediamo di riassumerli:

000 spazio	codice 32	colore 0	Nero
001 erba	codice 160	colore 13	verde chiaro
002 acqua	codice 102	colore 14	azzurro
003 mura	codice 104	colore 15	grigio chiaro
004 oggetto	codice 30	colore 10	rosso chiaro

ecc...

Definiti questi codici identificativi, si inizia a disegnare la prima schermata usando i codici sopra definiti.

Punto 3)

Per semplicità disegneremo una schermata/mappa di 10x10 caratteri, così da esser veloci, ma ovviamente potete sfruttare tutta l'area caratteri 40x25 oppure andare oltre impiegando tutta la memoria che ritenete necessario per lo scopo.

	Colonna									
Riga	0	1	2	3	4	5	6	7	8	9
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0

Come dicevamo, creeremo una piccola schermata di 10x10 che andremo a stivare nell'area di memoria a partire dall'indirizzo: 49152, dove troviamo 4kByte di RAM liberi e pronti allo scopo.

Punto 4)

Adesso procediamo con il disegno vero e proprio della schermata:

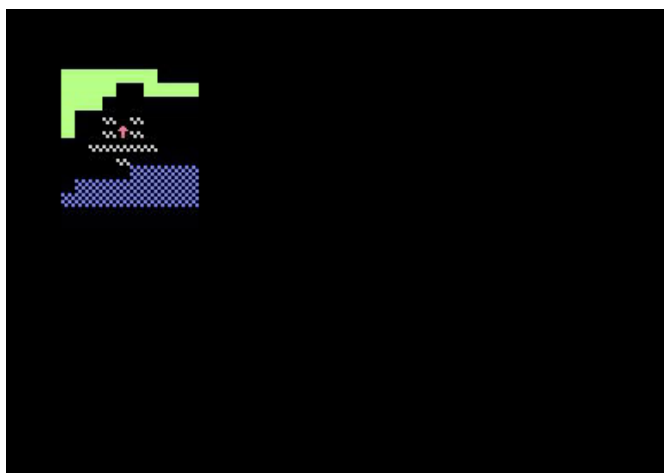




Colonna

Riga	0	1	2	3	4	5	6	7	8	9
1	1	1	1	1	1	1	1	0	0	0
1	1	1	1	0	0	1	1	1	1	1
1	1	1	0	0	0	0	0	0	0	0
1	0	0	3	0	3	0	0	0	0	0
1	0	0	3	4	3	0	0	0	0	0
0	0	3	3	3	3	3	0	0	0	0
0	0	0	0	3	0	0	0	0	0	0
0	0	0	0	0	2	2	2	2	2	2
0	2	2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2	2	2

Questo dovrebbe essere il risultato finale:



Una mappa semplice per non perdere di vista l'obiettivo: le routine.

Ecco una soluzione per salvare in memoria la mappa appena disegnata:

```
1 CLR:PRINTCHR$(147)"LETTURA MAPPE"
5 M=49152:M1=99
10 FORI=MTOM+M1:READX:POKEI,X:NEXT
20 END
100 DATA 1,1,1,1,1,1,1,0,0,0
101 DATA 1,1,1,1,0,0,1,1,1,1
102 DATA 1,1,1,0,0,0,0,0,0,0
103 DATA 1,0,0,3,0,3,0,0,0,0
104 DATA 1,0,0,3,4,3,0,0,0,0
105 DATA 0,0,3,3,3,3,0,0,0,0
106 DATA 0,0,0,0,3,0,0,0,0,0
107 DATA 0,0,0,0,0,2,2,2,2,2
108 DATA 0,2,2,2,2,2,2,2,2,2
109 DATA 2,2,2,2,2,2,2,2,2,2
```

Le uniche due righe degne di nota sono la 5 e la 10, la prima serve per settare due variabili M=49152 come

indirizzo di memoria iniziale, ed M1=99 (100-1=99 si parte contando anche lo zero) dimensione della mappa (10x10); la riga 10 è un ciclo FOR per caricare i valori presenti nei DATA e stivarli dall'indirizzo 49152 in poi.

Ora che abbiamo inserito in memoria la nostra prima mappa, vediamo come disegnare a video la nostra mappa: il piccolo programma che segue legge i dati, li traduce in codice carattere e colore, poi li salva in memoria video e colore per la visualizzazione:

```
1 clr:poke53280,0:poke53281,0:printchr$(147)
2 mv=1024:mc=55296:i=0:t=99:ri=0
3 sc=49152:jp=0:nc=5
5 dim og(4,1)
10 for i=0tonc-1
12 read og(i,0), og(i,1)
13 next
99 rem disegna mappa
100 sc=sc+jp*t
101 fori=0tot
102 pokemv+ri,og(peek(sc+i),0)
103 pokemc+ri,og(peek(sc+i),1)
104 ri=ri+1
105 if ri=10 then110
106 next:goto200
110 ri=0:mv=mv+40:mc=mc+40:goto106
200 geta$:ifa$=""then200
205 end
1000 data 32,0,160,13,102,14,104,15,30,10
```

Analizziamo il codice:

nella prima riga si esegue un reset delle variabili, viene settato lo schermo ed il bordo di colore nero. Le righe 2-3-5 servono per dichiarare le variabili, associandoli il valore opportuno:

mv = memoria video, questa variabile andrà modificata in base a dove vogliamo piazzare la nostra mappa;
mc = memoria colore, anche per questa come la precedente, andrà calcolata la posizione corretta.

Nell'esempio mv=1024, mc=55296 sono il primo carattere in cima a sinistra dello schermo del C64.

Mentre la variabile i = 0 viene usata come contatore; t=99 è la dimensione della mappa (come detto precedentemente da 0 99 = 100); ri = riga indice, praticamente tiene la posizione nella riga corrente, in





sostanza la colonna dove disegnare;

sc = 49152 è la sorgente di dati, dove si trova la mappa;
jp = 0 viene usata per il calcolo della posizione in memoria,
ovvero se avessimo più di una mappa, jp terrà la posizione
attuale, inizialmente avendo una sola mappa il valore di
jp rimarrà uguale a 0.

In riga 5 viene dimensionato un vettore og (oggetti) con
dimensione 5x2 (ricordo che il primo elemento del vettore
è lo zero, quindi avrà per dimensioni 4x1).

Nelle righe 10-12-13 viene letta la lista dei caratteri
impiegati nella mappa, è un normale ciclo FOR da 0 a nc-
1 (numero caratteri -1), le coppie di valori (codice carattere
+ colore) saranno stivati della matrice og().

Alla riga 100 viene calcolato l'indirizzo di partenza della
mappa attuale, partendo dalla prima mappa jp = 0

sc = sc + jp*t che equivale a: sc = 49152 + 0*99;

dalla riga 101 alla 106 viene eseguito il ciclo di lettura
della mappa dati, che si trova all'indirizzo sc, il valore sc+i
viene usato per leggere la memoria con il comando PEEK,
questo valore appena estratto verrà usato come indice
riga della matrice og, in colonna 0, tale valore verrà usato
come argomento da inviare alla memoria colore attraverso
il POKE all'indirizzo MV+RI.

La stessa cosa viene fatta alla riga 103 per la mappa
colore, ovvero viene estratto dalla matrice og() con lo
stesso indice sc+i, ma in colonna 1, il colore del codice
del carattere da stampare a video. Questo parametro
verrà passato alla memoria colore attraverso il POKE.

Seguono gli incrementi della posizione nella mappa video
e colore, in posizione 105 si verifica che non sia arrivato
alla fine della riga della mappa (in questo esempio 10
caratteri), nel caso si salta alla riga 110 dove viene
azzerata la colonna ed incrementata la riga, sommando 40.
Si esegue lo stesso ciclo per tutti i numeri di riga, raggiunta
la fine della mappa il programma termina attendendo la
pressione di un tasto.

Benché il programma sia molto semplice funziona in modo
soddisfacente, ma essendo scritto in basic ha i suoi
“problemi” legati alla lentezza.

Vediamo come si potrebbe realizzare lo stesso programmino,
ma sfruttando l'assembly:

```
*=49152                ;inizio del programma
```

```
lda #147                ;cancella lo schermo
jsr $f1ca                ;

lda #0                  ;setta bordo e schermo
sta 53280                ;di colore nero
sta 53281                ;

jsr reset                ;resetta tutte
le variabili
jsr setmappa            ;setta i puntatori
della mappa
loop1
jsr leggimappa          ;inizia lettura
primo byte della mappa
jsr stampa              ;procedura di
stampa del carattere letto
inc index                ;incrementa
l'indice del i-esimo carattere letto
lda index                ;confronta con
la fine della tabella
cmp tabella              ;
bne loop1                ;Se non uguale
torna a loop1
jsr premiuntasto        ;Attende la pressione
di un tasto
rts                      ;Chiude il programma
;=====
reset
lda #0                  ;Azzerare le variabili
sta $fa                  ;
sta $fb                  ;
sta $fc                  ;
sta $fd                  ;
sta $fe                  ;
sta $ff                  ;
sta pos                  ;
sta index                ;
lda #10                  ;Carica 10 come
dimensione massima di colonna
sta colon                ;
lda #2                  ;Setta a zero il
livello di gioco - step 2
sta liv                  ;
rts                      ;Ritorna al chiamante
;=====
```





```

                                clc                                ;Somma alla
                                posizione attuale
premiuntasto                    lda $fa                                ;40 caratteri
                                (40 colonne video)
                                adc #40                                ;per partire con
                                una nuova linea
                                sta $fa                                ;
                                lda $fb                                ;
                                adc #0                                ;
                                sta $fb                                ;
                                clc                                ;Stesso
leggimappa                      ;procedimento per la memoria colore
                                lda $fc                                ;
                                adc #40                                ;40 colonne
                                sta $fc                                ;
                                lda $fd                                ;
                                adc #0                                ;
                                sta $fd                                ;
                                rts
                                ;=====
                                setmappa
                                lda video                                ;Setta i puntatori
                                della
                                sta $fa                                ;pagina zero,
                                per gestire
                                lda video+1                            ;memoria video e memoria
                                sta $fb                                ;colore.
                                lda color                                ;
                                sta $fc                                ;
                                lda color+1                            ;
                                sta $fd                                ;
                                ldx liv                                ;Setta il puntatore
                                lda addmap,x                            ;alle mappe.
                                sta $fe                                ;
                                inx                                    ;
                                lda addmap,x                            ;
                                sta $ff                                ;
                                rts
                                ;=====
                                ;Definizione delle variabili
                                video !word 1024
                                color !word 55296
                                objj !byte 0
                                objc !byte 0

                                jsr $ffe4                                ;Kernal routine
                                del $GETIN
                                cmp #$00                                ;confronta con 0
                                beq premiuntasto                        ;Se uguale a 0 allora
                                non è stato premuto un tasto
                                rts                                    ;Altrimenti salta
                                a premiuntasto
                                ;=====
                                leggimappa
                                ldy index                                ;setta posizione
                                dell'indice di lettura mappa
                                lda ($fe),y                            ;legge il primo
                                carattere
                                tax                                    ;lo trasferisce in X
                                lda defobj,x                            ;preleva dalla
                                tabella DEFOBJ il carattere X-esimo
                                sta objj                                ;salvalo in OBJJ
                                lda defcol,x                            ;preleva il
                                carattere X-esimo
                                sta objc                                ;salvalo in OBJC
                                rts
                                ;=====
                                stampa
                                lda objj                                ;carica il codice
                                del disegno
                                ldy pos                                ;legge la posizione
                                sta ($fa),y                            ;setta in memoria video
                                lda objc                                ;carica il codice colore
                                sta ($fc),y                            ;setta in memoria colore
                                iny                                    ;incrementa
                                indice di posizione
                                sty pos                                ;salva la posizione
                                attuale nella variabile POS
                                cpy colon                                ;confronta con
                                COLON (limite della schermata)
                                beq nuriga                                ;se raggiunto il
                                limite, salta alla nuova riga
                                rts
                                ;=====
                                nuriga
                                lda #0                                ;setta 0 in pos
                                (posizione iniziale)
                                sta pos                                ;

```





```

index    !byte 0
tabella  !byte 100
colon    !byte 10
pos       !byte 0
liv       !byte 0
defobj   !byte 32,160,102,104,30
defcol   !byte 0,13,14,15,10
addmap   !word map1,map2

map1     !byte 1,1,1,1,1,1,1,0,0,0
          !byte 1,1,1,1,0,0,1,1,1,1
          !byte 1,1,1,0,0,0,0,0,0,0
          !byte 1,0,0,3,0,3,0,0,0,0
          !byte 1,0,0,3,4,3,0,0,0,0
          !byte 0,0,3,3,3,3,3,0,0,0
          !byte 0,0,0,0,3,0,0,0,0,0
          !byte 0,0,0,0,0,2,2,2,2,2
          !byte 0,2,2,2,2,2,2,2,2,2
          !byte 2,2,2,2,2,2,2,2,2,2

map2     !byte 1,1,1,1,1,1,1,1,1,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,0,0,0,0,0,0,0,0,1
          !byte 1,1,1,1,1,1,1,1,1,1

```

ho commentato le linee di codice per spiegare come funziona il programma, che può essere schematizzato in questo modo:

- leggere il primo byte della mappa;
- stampare il carattere decodificato secondo lo schema prefissato;
- incrementare l'indice usato per la lettura della tabella;
- se non abbiamo raggiunto la fine della tabella, ripetiamo il ciclo dall'inizio.

Questo programma se lanciato, stampa in un tempo quasi nullo tutta la mappa; ottimo se vogliamo dare al nostro gioco un aspetto professionale. Ma sappiamo tutti che utilizzare l'assembly non è così semplice, l'errore è sempre

in agguato; non è un linguaggio molto elastico, quindi possiamo scendere a compromessi, ovvero usare un linguaggio compilato, così da poter avere uno strumento di sviluppo più semplice da utilizzare, ma molto potente; sto parlando del Vision Basic versione 1.1.

Vediamo subito il listato:

```

1 clr:cls:colors 1,0,0
5   mmv=1024:mmc=55296:scr=0:liv=0:oc=7:i=0:zz=0
6 x=0:xx=0:yy=0:oo=0:ogl=0:ocl=0:tabella=100
10 dim ogx(4),clx(4)
11 dim livscr(4)
35 (=====)
36 (definizione decoder)
40 i=0
41 read ogx(i),clx(i):[inc i]:do41,5
43 (=====)
44 (set indirizzi delle mappe)
45 livscr(0)=49152
48 (=====)
51 (lettura primo schermo)
52 zz=livscr(liv)-1:x=0
55 add xx=x+mmv:add yy=x+mmc:add zz=zz+1
57 oo=peek(zz)
60 ogl=ogx(oo):ocl=clx(oo)
62 pokexx,ogl:pokeyy,ocl
64 [inc x]
65 if x=10 then x=0:add mmv=mmv+40:add mmc=mmc+40
66 do55,tabella
70 keypress:end
75 (=====)
1000 data32,0,160,13,102,14,104,15,30,10

```

Il listato comprende pochissimi comandi Vision Basic per tenere una linea con la versione Basic V2; vediamo in dettaglio:

1) resettare le variabili, cancellare lo schermo e settare come colore di bordo e schermo il nero, il colore del testo: bianco.

5-6-10-11) set delle variabili, i vettori oggetto e colore (purtroppo il Vision Basic non gestisce le matrici), ed il numero di elementi della tabella.

41) loop di lettura codici carattere e colore per stampare a video la mappa.

45) ho inserito solo l'indirizzo di partenza della prima mappa da stampare, gli schermi aggiuntivi sono da





aggiungere al vettore LIVSCR.

52-55) funzioni di calcolo degli indirizzi della mappa video (XX) e colore (YY); la variabile ZZ contiene l'indirizzo di lettura della mappa da decodificare a video.

57) leggere un byte dalla mappa da decodificare

60) estrarre i carattere ed il suo colore in funzione del codice letto dalla mappa.

62) stampare nella memoria video e colore i valori decodificati

64) incrementare l'indice che tiene conto di quanti byte abbiamo letto dalla mappa da decodificare.

65) verifico di non essere arrivato al termine della riga (che in questo esempio è di 10 caratteri), nel caso bisogna aggiornare le variabili (mmv) ed (mmc) saltando alla riga successiva.

66) ripetere il ciclo un numero di volte, pari alla dimensione della mappa (10x10 = 100 elementi)

70) attendere la pressione di un tasto, quindi terminare il programma

la versione Vision Basic che nel peggiore dei casi è 6 volte più rapida del fratello V2; al primo posto si colloca la versione in assembly: questa viene eseguita quasi in tempo zero, ma è anche la più complessa e lunga da implementare.

A scopo informativo vi è anche la possibilità di sfruttare il Basic compilato, ottenendo un incremento di prestazioni di almeno 4 volte rispetto alla controparte interpretata. Sta a noi decidere come implementare questo programma, quindi a voi la scelta e buona programmazione.

*Permettetemi due parole di ringraziamento a Marco Bergomi per avere aderito alla mia richiesta di scrivere un articolo sul suo lavoro sulle mappe dei giochi. Marco e' l'amministratore del gruppo Facebook **MOS-Commando**, dove pubblica spesso post interessanti, giochi e consigli di lettura. Se non conoscete il gruppo, dategli un'occhiata,, ne vale la pena.*

Francesco F.

Mettendo a confronto queste 3 versioni, la cosa che colpisce maggiormente è la velocità d'esecuzione del programma; la maglia nera spetta al Basic classico; segue

```
DATA 1,1,1,1,1,1,1,0,0,0
DATA 1,1,1,1,0,0,1,1,1,1
DATA 1,1,1,0,0,0,0,0,0,0
DATA 1,0,0,3,0,3,0,0,0,0
DATA 1,0,0,3,4,3,0,0,0,0
DATA 0,0,3,3,3,3,3,0,0,0
DATA 0,0,0,0,3,0,0,0,0,0
DATA 0,0,0,0,0,2,2,2,2,2
DATA 0,2,2,2,2,2,2,2,2,2
DATA 2,2,2,2,2,2,2,2,2,2
```

<u>Pos.0</u>	<u>Pos.1</u>	<u>Pos.2</u>	<u>Pos.3</u>	<u>Pos.4</u>	
32	160	102	104	30	Codice video
0	13	14	15	10	Codice colore

Fig. 1 - Con il valore 3 letto nella mappa, viene piazzato in memoria video il codice 104 di colore 15





Preservazione digitale con miglioramento del codice

di David "Cercamon" La Monaca

Una delle attività che RetroMagazine World porta avanti già da diverso tempo, è il progetto denominato RetroLiPS. Questo progetto si propone di recuperare, digitalizzare e preservare dall'oblio software scritto per le macchine retro, in particolare tutti quei programmi che fin dai primi anni '80 comparivano su libri e riviste sotto forma di listati prevalentemente scritti in BASIC, il linguaggio che tutti i computer dell'epoca adottavano come strumento di default per interagire con le risorse tecniche.

Unendo il desiderio di contribuire a RetroLiPS e cedendo ad una personale sensazione di nostalgia, che, ne sono certo, tutti i fan del retrocomputing avranno provato almeno una volta, di recente ho preso fra le mani una delle poche riviste cartacee che ancora fanno parte della mia retro-biblioteca. Si tratta di Commodore Computer Club n. 14 dell'ottobre 1984, una rivista italiana che coniugava notizie, articoli tecnici e di programmazione per le macchine dello storico marchio americano, lasciando poco spazio a giochi e recensioni commerciali. Fra i ragazzini italiani degli anni '80, CCC rappresentava un valido punto di riferimento ed un'opportunità concreta per conoscere e studiare l'hardware ed i sistemi operativi degli home computer, nonché approfondire metodi e tecniche per programmarli al meglio.

Il numero 14 per certi versi può essere ricordato fra i più importanti di tutta la storia della rivista, soprattutto per via della pubblicazione integrale delle famose routine grafiche II di Danilo Toma (a partire da pag. 40),

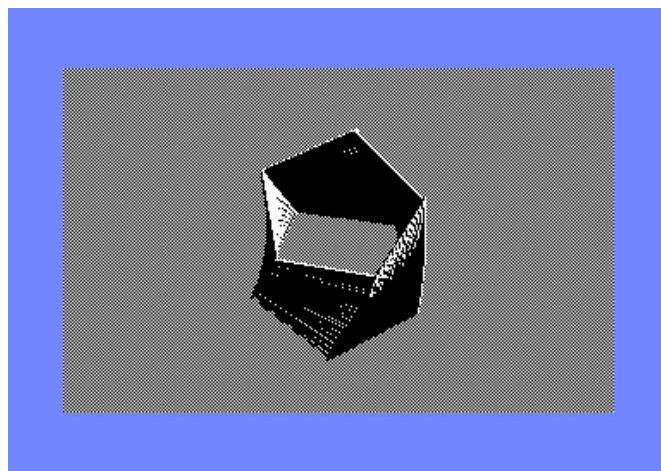


Fig. 2 - solidi - prisma con torsione

un'estensione composta di ben 9 nuovi comandi grafici per il BASIC v2, notoriamente carente in quanto a istruzioni dedicate alla gestione della grafica bitmap multicolore e in alta risoluzione. Le routine grafiche II, al tempo, rappresentarono per molti appassionati italiani del C64 una soluzione comoda ed efficiente per porre rimedio alla mancanza del BASIC v2.

La prima versione delle routine grafiche di Toma era stata pubblicata pochi mesi prima sul numero 10 della stessa rivista nell'aprile 1984. Si tratta di una versione base ed essenziale che comprende 5 routine in linguaggio macchina richiamabili dai programmi BASIC con il comando SYS. Le routine consentono di attivare e pulire la grafica bitmap hires (320x200 pixel), plottare punti e disegnare linee, scegliere i due colori concessi e tornare al modo testo. In soli 717 byte, senza invadere l'area di memoria riservata

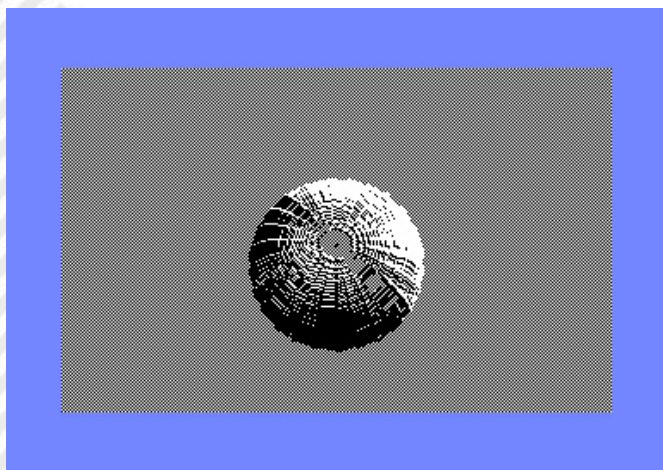


Fig. 1 - solidi - sfera

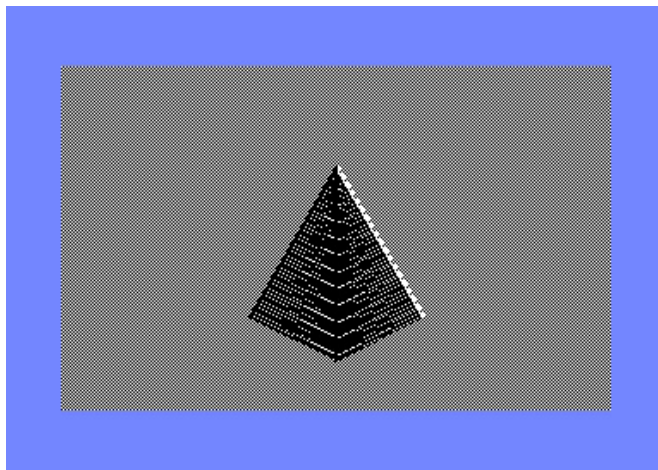


Fig. 3 - solidi - piramide





Listato Solidi originale - codifica CBM Prg Studio 64

```

0 rem "O{E*28}P"
1 rem "{cm g} {W*7}s o l i d i {W*7} {cm m}"
2 rem "{cm g} {cm m}"
3 rem "{cm g} versione c 64 {cm m}"
4 rem "{cm g} {cm m}"
5 rem "{cm g} di luca galizzi {cm m}"
6 rem "{cm g} {cm m}"
7 rem "{cm g} via ninfea {cm m}"
8 rem "{cm g} {cm m}"
9 rem "{cm g} cusano milanino (mi) {cm m}"
10 rem "{cm g} {cm m}"
11 rem "{cm g} tel: 02/6187588 {cm m}"
12 rem "L{R*28}{sh @}"
13 rem "{cm +*30}"
14 rem "{cm +}{space*28}{cm +}"
15 rem "{cm +} caricare le routines gra- {cm +}"
16 rem "{cm +} fiche di danilo toma ccc10{cm +}"
17 rem "{cm +}{space*28}{cm +}"
18 rem "{cm +*30}"
24 rem *** type-in/testing by dlm/rmw ***
29 clr:x%=0:y%=0:x1%=0:y1%=0
30 a=0:b=100:c=1:d=50:e=3:f=.25:g=.5:h=2:gr=atn(1)/45:w=200
40 pl=49693:dr=49728:cl=49763:co=49785:mo=49821
50 rem *** input parametri ***
60 poke248,0:sysmo:print "{clear}{down*3}"
70 print "{right*6}figura :"
75 print "{down}{right}{reverse on}1{reverse off} prisma":print "{down}{right}{reverse on}2{reverse off}
piramide"
77 print "{down}{right}{reverse on}3{reverse off} sfera"
78 print "{down}{right}{reverse on}4{reverse off} emisfera":print "{down}{right}{reverse on}5{reverse
off} iperboloide"
80 input "{down*3}quale scegli ";tp
85 iftp<1or tp>5 then print "{up*5}":goto 80
90 input "{down}quanti lati ";lf
100 iflf<1 then 90
110 input "{down}angolo di incidenza ";vr
111 ifvr<1or vr>90 then print "{up*3}":goto 110
120 input "{down}angolo di rotazione ";ar
130 input "{down}angolo di torsione ";at
140 gosub 1000
150 rem *** disegno del solido ***
160 ar=gr*ar:al=100*sin(vr*gr):y0=100+al/2:x0=160
170 cs=cos(vr*gr):se=8*atn(1)/lf
180 forj=atob:ontpgosub250,260,270,280,290
190 ar=ar+at*gr/b
200 rem *** disegno della figura piana ***
205 an=8*atn(1)+ar+.001:ry=di*cs
210 x%=c+x0+di*sin(ar):y%=w-(c+y0+ry*cos(ar))
215 fori=artoanstepse
220 y1%=w-(c+y0+ry*cos(i)):x1%=c+x0+di*sin(i)
225 poke2,c:sysdr:x%=x%-c:y%=y%-c:x1%=x1%-c:y1%=y1%-c
230 poke2,a:sysdr:x%=x1%:y%=y1%:nexti
240 y0=y0-al/b:nextj:goto300
250 di=d:return
260 di=d*(c-j/b):return

```





```
270 di=d*sqr(c-(c-j/d)^h):return
280 di=d*sqr(c-(j/d)^h):ifj=dthenj=110
285 return
290 di=d*sqr(f+e*((j/b)-g)^h):return
300 geta$:ifa$=""then300
310 goto14
999 *** reticolato di sfondo ***
1000 poke248,c:pokeh,a:poke251,c:poke252,a
1002 sysco:syscl:sysmo
1005 p=57344:h=170:i9=85:i8=1
1010 forx=pto65343step2
1020 pokex,h:pokex+i8,i9:nextx
1035 h=2
1040 return
2003 rem"{E*30}"
2010 rem prova ad introdurre questi va-
2020 rem lori; rispettivamente alle ri-
2030 rem chieste di figura, lati, inci-
2040 rem denza, rotazione, torsione.
2050 rem      3      25      65      50      40
2060 rem      2       4      60       0       0
2070 rem      5       5      50       0       0
2080 rem      3      50      45      60      30
2090 rem      5       6      60       0       0
2100 rem"{cm p*30}"
2110 rem *** type/in/testing by dlm/rmw
```

ai programmi BASIC, le routine grafiche I svolgono egregiamente, e con una buona rapidità, il proprio compito, tanto che la redazione della Systems Editoriale (la casa editrice della rivista CCC) da quel momento le adottarono proponendole a tutti i lettori per scrivere programmi in grado di sfruttare le risorse grafiche offerte dal VIC-II. Proprio nel numero 14 apparve un programma sorprendentemente valido ed efficiente per disegnare in alta risoluzione alcune figure geometriche solide, basandosi esplicitamente sulle routine grafiche I del n. 10. L'articolo dal titolo "Solidi ad alta risoluzione", a firma di Luca Galluzzi, si trova a pag. 14 della rivista ed è accompagnato dal listato completo in grado di disegnare prismi, piramidi,

sfere, semisfere ed iperboloidi secondo alcuni parametri di base utilizzando le chiamate SYS alle routine grafiche sopra citate.

Una delle cose interessanti di quest'articolo è il metodo con il quale si raggiunge l'obiettivo di creare uno spazio pseudo-3D per visualizzare le figure solide. L'effetto tridimensionale viene infatti ottenuto mediante la tecnica dell'ombreggiatura, che il programma calcola per ogni singola linea o punto del solido tracciato sullo schermo. Qui interviene un problema tecnico, dal momento che la grafica hires del C64 non consente l'uso di più di due colori (uno per lo sfondo ed uno per il disegno), ma per tracciare i solidi in modo efficace ne servono almeno tre: il colore di sfondo, il colore di primo piano (o di disegno) ed il colore dell'ombreggiatura. Per superare questo limite, il programma usa i due colori base (bianco e nero) per disegno ed ombreggiatura ed un reticolato (fatto di pixel bianchi e neri accostati secondo una griglia 8x8) per creare un pattern che viene ripetuto per tutto lo schermo, dando così l'impressione di usare un terzo colore (grigio) per lo sfondo. Questa soluzione funziona bene: sul pattern di pixel bianchi e neri dello sfondo, i due colori base tracciano le linee delle diverse figure geometriche previste dal programma con un effetto finale che risulta molto

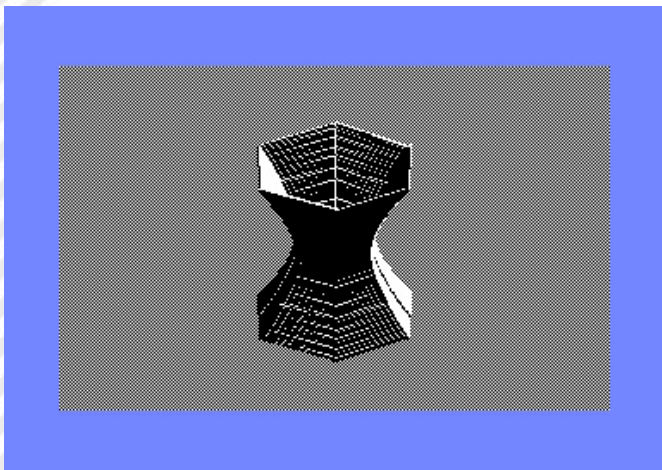


Fig. 4 - solidi - iperboloide





Listato assembly routine Hires Bitmap Fill

```

* = C2D0                ; inizio routine - sys 49872

LDA #$00                ; $E000 inizio hires
STA $FD                 ; in byte basso e alto
LDA #$E0                ; nelle locazioni
STA $FE                 ; $FD e $FE
LDX #$1E                ; ciclo esterno
loop1 LDY #$00           ; $1E=31 - 31*256 = primi 7936 bytes
loop2 LDA #$55           ; $55 = 0101 0101 reticolato di bit
      STA ($FD),Y
      DEY
      ROL                ; da $55 a $AA = 1010 1010
      STA ($FD),Y
      DEY
      BNE loop2
      INC $FE            ; incrementa byte alto
      DEX
      BPL loop1         ; continua il ciclo
      LDY #$40           ; ultimi 64 bytes (8 blocchi 8x8)
loop3 LDA #$55
      STA ($FD),Y
      DEY
      ROL                ; raddoppia $55=85 -> 170=$AA
      STA ($FD),Y
      DEY
      BPL loop3
      RTS

```

gradevole (rif. figg. 1-4).

È proprio su questo “terzo colore” e sul metodo con il quale riempie l'area di sfondo della pagina grafica dello schermo, che si fonda l'idea di questo nostro umile contributo. Sì, perché, come detto in precedenza, il programma “Solidi ad alta risoluzione” esegue bene il suo lavoro ma ovviamente il disegno è relativamente lento per quelli che sono gli standard odierni, anche se assolutamente normali per l'epoca di pubblicazione. Ciò che davvero risulta molto lenta è la tecnica di riempimento dello sfondo perché viene implementata da un ciclo FOR/NEXT che riempie le 8000 locazioni dello schermo hires con i valori alternati \$AA (170 decimale) e \$55 (85 decimale), creando così una scacchiera di punti bianchi e neri. Ma perché proprio quei due byte? Beh, basta pensare alla rappresentazione binaria dei valori \$55 e \$170 (fig. 5). Le righe 1005-1020 del listato (riquadro 1) mostra come il ciclo FOR/NEXT parta dalla locazione \$E000 (57344) di inizio schermo hires, locazione che viene modificata dalle routine grafiche I dalla posizione di default \$2000 (8192).

Dopo aver digitalizzato il listato del programma “Solidi

ad alta risoluzione”, ho pensato ad un piccolo intervento sul listato in BASIC per sostituire il ciclo di riempimento dello sfondo con una piccola routine in linguaggio macchina capace di svolgere lo stesso compito. Ciò consente di “pulire” lo schermo o, meglio, di inizializzarlo con i valori che costruiscono il reticolato, in una frazione del tempo rispetto al ciclo FOR/NEXT del BASIC. La routine aggiunta (riquadro 2), che parte da \$C2D0, in coda alle routine grafiche che vanno da \$C000 a \$C2CC, non è altro che la sequenza di due loop che percorrono le locazioni da \$E000 a \$F3FF (57344-65343) memorizzando i byte che formano il reticolato. Il primo loop riguarda i primi 7936 byte (31x256) mentre il secondo concerne gli ultimi 64 byte, sempre alternando i valori \$55 e \$AA. La routine di fill dello schermo hires fa uso delle locazioni in pagina zero \$FD e \$FE per puntare all'inizio dell'area di memoria \$E000, riservata allo schermo in modalità grafica hires. Poi utilizza l'indirizzamento indicizzato indiretto per scandire le locazioni da \$E000 in avanti. Come detto, il secondo ciclo chiude il gap di 64 byte mancanti per giungere sino alla locazione di fine hires 65343.

L'ultimo problema da affrontare è quello di trasformare





Listato BASIC modificato - codifica per CBM Prg Studio

```
10 rem ** solidi (routine toma 1)
12 rem **
14 forzz=49872to49872+41:readvv:pokezz,vv:next
18 data 169,0,133,253,169,224,133,254,162,30,160,0,169,170,145
22 data 253,136,42,145,253,136,208,245,230,254,202,16,238
26 data 160,64,169,170,145,253,136,42,145,253,136,16,245,96
28 rem **
29 clr:x%=0:y%=0:x1%=0:y1%=0
30 a=0:b=100:c=1:d=50:e=3:f=.25:g=.5:h=2:gr=atn(1)/45:w=200
40 pl=49693:dr=49728:cl=49763:co=49785:mo=49821
50 rem *** input parametri ***
60 poke248,0:sysmo:print"{clear}{down*3}"
70 print"{right*6}figura : "
75 print"{down}{right}{reverse on}1{reverse off} prisma":print"{down}{right}{reverse on}2{reverse off}
piramide"
77 print"{down}{right}{reverse on}3{reverse off} sfera"
78 print"{down}{right}{reverse on}4{reverse off} emisfera":print"{down}{right}{reverse on}5{reverse
off} iperboloide"
80 input"{down*3}quale scegli ";tp
85 iftp<1or tp>5thenprint"{up*5}":goto80
90 input"{down}quanti lati ";lf
100 iflf<1then90
110 input"{down}angolo di incidenza ";vr
111 ifvr<1orvr>90thenprint"{up*3}":goto110
120 input"{down}angolo di rotazione ";ar
130 input"{down}angolo di torsione ";at
140 gosub 1000
150 rem *** disegno del solido ***
160 ar=gr*ar:al=100*sin(vr*gr):y0=100+al/2:x0=160
170 cs=cos(vr*gr):se=8*atn(1)/lf
180 forj=atob:ontpgosub250,260,270,280,290
190 ar=ar+at*gr/b
200 rem *** disegno della figura piana ***
205 an=8*atn(1)+ar+.001:ry=di*cs
210 x%=c+x0+di*sin(ar):y%=w-(c+y0+ry*cos(ar))
215 fori=artoanstepse
220 y1%=w-(c+y0+ry*cos(i)):x1%=c+x0+di*sin(i)
225 poke2,c:sysdr:x%=x%-c:y%=y%-c:x1%=x1%-c:y1%=y1%-c
230 poke2,a:sysdr:x%=x1%:y%=y1%:nexti
240 y0=y0-al/b:nextj:goto300
250 di=d:return
260 di=d*(c-j/b):return
270 di=d*sqr(c-(c-j/d)^h):return
280 di=d*sqr(c-(j/d)^h):ifj=dthenj=110
285 return
290 di=d*sqr(f+e*((j/b)-g)^h):return
300 geta$:ifa$=""then300
310 goto28
999 *** reticolato di sfondo ***
1000 poke248,c:pokeh,a:poke251,c:poke252,a
1010 sysco:syscl:sysmo
1020 sys49872:rem ** fill hires screen
1030 h=2
1040 return
```





in linee DATA da inserire nel programma BASIC per sostituire il lento ciclo FOR/NEXT. Problema che sembra banale ma in realtà nasconde, nel caso specifico, qualche insidia. Non è efficiente rimpiazzare il ciclo di riempimento originale con un altro ciclo nella stessa posizione per leggere e memorizzare i valori delle linee DATA: il caricamento in memoria della routine sarebbe eseguita ad ogni ripartenza del programma. Quindi meglio posizionare il ciclo di lettura DATA/scrittura in RAM della routine di fill prima dell'inizio del programma alla linea 29. Ma per come sono strutturate le routine grafiche I di Toma non è possibile utilizzare alcuna variabile permanente prima delle variabili intere (come X%, Y%, ecc.) che vengono usate per le coordinate da passare alle routine di PLOT e DRAW. Perciò si esegue il caricamento della routine una sola volta prima della linea 29, in modo che la routine sia memorizzata e sia disponibile per tutta la durata dell'esecuzione del programma.

Una curiosità: nel codice della routine di riempimento dello sfondo abbiamo usato l'istruzione ROL (fig. 6) per cambiare il valore del registro accumulatore della CPU da \$55 a \$AA, cioè al posto di LDA #\$AA. Shiftando verso sinistra (ROL sta per Rotate Left) i bit dell'accumulatore (\$55 = 0101 0101) si ottiene proprio il valore \$AA = 1010 1010 (il bit 0 viene riempito con uno 0, proveniente dal bit di carry). Non è un gran risparmio di byte (solo 2), soprattutto in una routine che di per sé è già breve e compatta, ma chi sviluppa in assembly è abituato a ottimizzare sempre il codice, ove possibile.

Fra i riferimenti potete trovare il link per scaricare un dischetto immagine **solidi-rgt-final.d64** che contiene tutto l'occorrente per far girare il programma nella versione originale e in quella modificata con la routine di riempimento bitmap in linguaggio macchina. Consigliamo di confrontare le due versioni e godersi il disegno dei solidi su C64

```
01010101
10101010
01010101
10101010
01010101
10101010
01010101
10101010
```

Fig. 5 - blocco 8x8 reticolato di bit

originale, C64U (anche aumentando la frequenza della CPU) oppure su emulatore (ad es. VICE, sfruttando la modalità Warp, ALT+W).

Come emerge da questo articolo, e questo è il reale obiettivo del nostro contributo, anche la semplice attività di digitazione e conservazione di listati BASIC originariamente pubblicati su libri e riviste può sfociare in qualcosa d'interessante e, soprattutto, può fornire spunti di riflessione e materiale concreto da studiare e, se possibile, migliorare.

*Un ringraziamento speciale va al nostro **Marco Pistorio/RMW** e a **Phaze101** per la loro consueta disponibilità.*

Riferimenti

RetroLiPS – articolo sul sito web ufficiale:

- <https://www.retromagazine.net/solidi-in-alta-risoluzione-c64-basic-rgt/>

Raccolta scansioni rivista CCC:

- <https://ready64.org/download/categorie.php?cat=51>

Dischetto immagine solidi-rgt-final.d64:

- <https://www.retromagazine.net/download/rlips/solidi-rgt-final.d64>

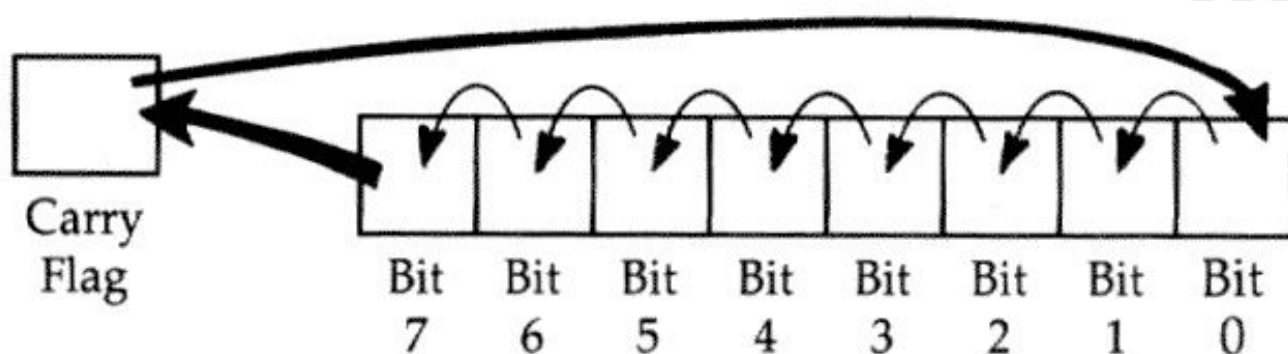


Fig. 6 - istruzione ROL





AmigaOS 3.3 – L'evoluzione del sistema operativo

di Giampaolo Moraschi

AmigaOS classic continua a evolversi, e AmigaOS 3.2 ha rappresentato il primo grande aggiornamento dopo molti anni, modernizzando il sistema per macchine come A1200, A3000 e A4000. Questa versione ha rinnovato Workbench, molti tool di sistema, i programmi di preferenze e gli strumenti da riga di comando, fungendo di fatto da “rinascita” del sistema classico.

Uno sguardo al nuovo sistema operativo

AmigaOS 3.3 non è solo una rivoluzione architeturale, ma lavora di fino introducendo nuovi strumenti, migliorando l'usabilità e affinando diversi componenti interni. Tra le novità principali c'è PartitionEdit, un nuovo tool di gestione dischi basato su ReAction che sostituisce HDToolBox, pensato per semplificare la configurazione di CF, SD e dischi moderni. Arriva anche il set di icone “Moonlight”, più pulito e coerente, pensato per risoluzioni alte e RTG,

ma resta possibile tornare alle icone classiche a 4 colori.

Il sistema di menu viene aggiornato con menu “floating” e “sticky”, in stile MagicMenu: non è più necessario tenere premuto il tasto del mouse, e i menu possono restare aperti finché non si sceglie un'opzione, il tutto configurabile nelle preferenze. La shell Amiga riceve miglioramenti per scripting e uso da riga di comando, e nei leak di sviluppo compare una nuova console chiamata WCON, con scrollbar verticali e tab per avere più sessioni in una sola finestra. Alcuni pannelli di preferenze, come Locale e Serial, vengono riscritti in ReAction per uniformare l'interfaccia e migliorarne l'uso.

Dietro le quinte, gran parte del lavoro riguarda Intuition, il motore GUI di AmigaOS. Vengono importate funzioni da sistemi più recenti (incluso AmigaOS 4), come WA_ToolBox, il disegno GN_EXTENT, il supporto nativo ai

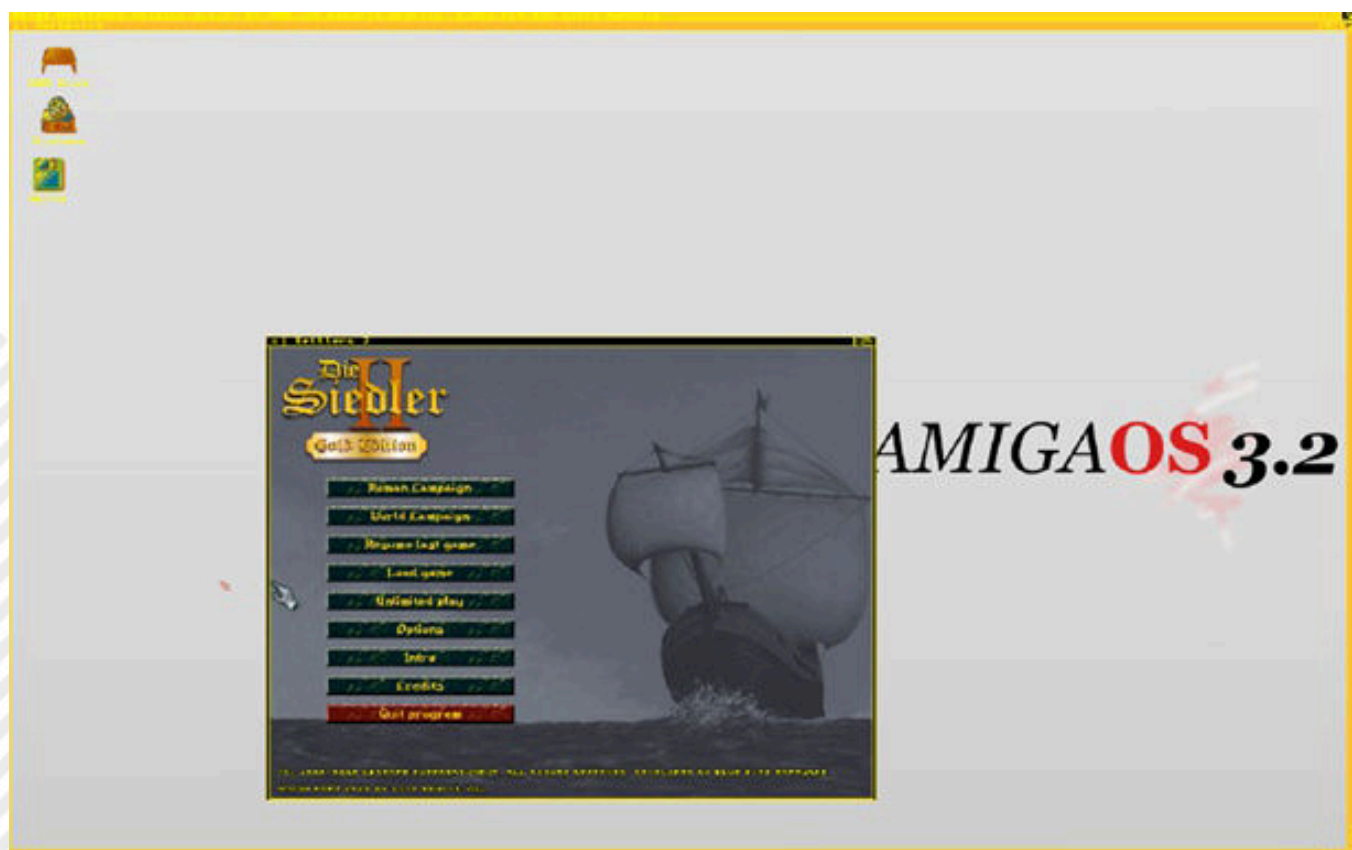


Fig. 1 - AmigaOS 3.2





Fig. 2 - Il nuovo AmigaOS 3.3

menu floating/sticky e un'infrastruttura per i menu contestuali. Si interviene anche sulla stabilità: ad esempio viene corretto il comportamento di `MakeClass()` nel calcolo delle dimensioni degli oggetti, evitando potenziali overflow, e si migliorano le allocazioni di memoria con flag come `MEMF_NO_EXPUNGE` per prevenire expunge indesiderati delle librerie.

Ci sono poi piccole ottimizzazioni di codice (eliminazione di duplicati, risparmio di memoria) e nuovi switch per vari comandi da CLI (`AVAIL`, `UNITS`, `DACONTROL GET`, `DISKDOCTOR VERIFY`, `MAKEDEV ICON`), che ampliano le possibilità di diagnostica e scripting.

In sintesi, AmigaOS 3.2 ha ricostruito e modernizzato la base del sistema classico, mentre AmigaOS 3.3 la affina con nuovi tool, aggiornamenti visivi, menu e shell migliorati e un Intuition più robusto e moderno, pur mantenendo la piena compatibilità con l'hardware e il software storici. Non ci resta che attendere l'uscita ufficiale prevista per il 2026 mentre da poche settimane Hyperion ha rilasciato la versione in digital download di AmigaOS 3.2 e dei suoi upgrade.

Link - https://secure.2checkout.com/order/product.php?PRODS=53302996&QTY=1&AFFILIATE=252771&__c=1





Intervista a Richard Farrell aka Vyper68

20 domande ad un tecnico elettronico: una figura importante nel retrocomputing, il tipo di esperto cui ci rivolgiamo quando i nostri amati sistemi vintage hanno bisogno di riparazioni
di David "Cercamon" La Monaca

Introduzione

Incontrare tecnici elettronici, specialisti delle riparazioni ed esperti di retro-hardware significa immergersi nel tessuto operativo della storia dell'informatica. La loro presenza sui social e nelle community specializzate è in costante crescita, riflesso di un interesse per il retrocomputing che non accenna a fermarsi. Questi professionisti e hobbisti mantengono nel tempo un bagaglio di competenze trasversali, custodi di un'eredità tecnica che sta diventando merce rara per chi studia, preserva o semplicemente utilizza sistemi vintage.

La sfida cruciale è: come possiamo tramandare e documentare questo immenso patrimonio di conoscenze pratiche? Gli home computer e le console prodotti tra gli anni '70 e '90 continuano a battere colpi grazie a un fragile ecosistema, sorretto dall'esperienza di chi ne domina architetture, componenti e limiti operativi. Eppure, il tempo non fa sconti nemmeno all'hardware più robusto: materiali che invecchiano, condensatori che perdono efficienza, batterie che corrodono le piste, chip di memoria e logiche TTL soggetti a guasti imprevedibili. Quando avviene il blackout, la domanda sorge spontanea: la riparazione è fattibile? Ne vale la pena? Le parti di ricambio sono sempre più difficili da trovare, i costi di riparazione lievitano e non è banale scovare veri esperti formati su tecnologie ormai fuori produzione da decenni.

In questo scenario, i riparatori specializzati giocano un ruolo fondamentale. Spesso lavorando lontano dai riflettori, intervengono con un mix di intuito e rigore metodologico: diagnostica strumentale, analisi dei circuiti, ripristino di piste danneggiate e sostituzione di componenti obsoleti con equivalenti moderni. Sono diventati le figure chiave per la salvaguardia del nostro patrimonio digitale, i custodi di un sapere tecnico che, altrimenti, andrebbe perduto. Per questo motivo RMW ha inaugurato una rubrica fissa dedicata alle interviste con questi "maghi" dell'hardware: professionisti e appassionati che operano dietro le quinte del retrocomputing. Abbiamo selezionato 20 domande per mettere in luce esperienze, procedure, sfide e soluzioni



Fig. 1 - Richard Farrell aka Vyper68

adottate da chi, con competenza e metodo, ridona vita alle macchine che hanno plasmato l'alba dell'era digitale. In questa intervista incontriamo Richard Farrell, noto anche con il nickname di Vyper68, dal Regno Unito. Da tanti anni Richard mette la sua passione per i primi home computer e i sistemi di intrattenimento, unita a una profonda conoscenza dell'elettronica di consumo, al servizio della sua sempre crescente collezione e anche degli utenti di forum e community online.

Q&A – 20 domande / 20 risposte

1. DLM – Un saluto e un ringraziamento a Richard/Vyper68, che ha gentilmente accettato questa intervista. Prima di tutto, potresti presentarti brevemente e raccontarci qualcosa del tuo background personale nel mondo dell'elettronica e dei computer/console?

Vyper68 – Ciao David! Dunque, sono nato in Scozia ma





ho passato la mia vita adulta nel nord-est dell'Inghilterra, in un tranquillo paesino chiamato Hurworth. Per quanto riguarda il lavoro, ora sono in pensione. Sebbene non fossi ufficialmente assunto come tecnico IT nell'azienda in cui lavoravo, finivo spesso per ricoprire quel ruolo: assemblavo PC per i colleghi e riparavo vecchie macchine, così mi chiesero anche di gestire il network e i server aziendali e di occuparmi del troubleshooting dei PC dell'ufficio come incarico extra. Mi divertivo così tanto a farlo che non ho mai chiesto un soldo in più! (ride, ndr). Ho avuto un'istruzione piuttosto standard e ho lasciato la scuola con diversi diplomi ("O levels" e CSE) tra cui Informatica e Fisica. Ho frequentato il college per approfondire gli studi informatici a un livello superiore, ma non sono mai andato all'università perché avevo la necessità di iniziare a guadagnare abbastanza presto. Così, mi sono tuffato direttamente nel mondo del lavoro.



Fig. 2 - Banco di lavoro

2. DLM – Com'è nato il tuo interesse per l'elettronica e l'informatica? Qual è la macchina o il brand vintage che più ti diverte utilizzare, espandere o riparare, e perché?

Vyper68 – Sono sempre stato un appassionato di tecnologia; mio padre adorava la scienza e guardavamo spesso insieme "Tomorrow's World" sulla BBC. Ero affascinato da come venivano costruite le cose e dai componenti interni che le facevano funzionare. Ricordo ancora la prima volta che smontai il mio Atari 2600 per vedere cosa ci fosse dentro: fissavo la scheda madre cercando di capirci qualcosa. Avevo solo dieci anni, quindi non avevo idea di cosa stessi guardando, ma volevo imparare.

Facendo un salto in avanti di qualche anno, la tecnologia aveva fatto passi da gigante. Vidi il mio primo microcomputer a scuola: il mio insegnante stava giocando a "The Hobbit"

su uno ZX Spectrum e stava mettendo in piedi un computer club. Più tardi riuscii a farmi regalare uno Spectrum per il mio compleanno, ma si guastò non appena lo collegai alla presa. Col senno di poi, era subito partita la RAM. Dovemmo tornare in negozio a cambiarlo, e fu lì che iniziai a pensare che avrei dovuto imparare a riparare i miei computer da solo.

Nel club della scuola iniziammo a costruire progetti elettronici e a imparare la programmazione. Il bello del club era che ogni studente aveva un computer diverso: Oric, C64, Atari e Dragon. Fortunatamente all'epoca c'era un'infinità di riviste e libri, così riuscii a studiare fino a raggiungere un livello che mi permettesse di riparare il mio Spectrum (cosa che feci un paio di volte) e poi quelli degli altri, man mano che si spargeva la voce.

Ho diversi computer a 8-bit preferiti, ma se dovessi scegliere, direi la gamma Atari, in particolare l'800XL e al secondo posto gli Oric-1/Atmos. L'800XL ha un design elegantissimo e il chipset custom lo rende una macchina fantastica (BASIC a parte). Esistono diverse espansioni per l'800XL/65XE, tra cui l'Ultimate 1MB che ne potenzia massicciamente memoria e funzioni, specialmente se usi il firmware di FlashJazzCat.

L'altro computer per cui ho un debole è l'Oric, per via della sua community e del chip sonoro AY. Ero un membro di Oric User Monthly e riuscii a rimediare un'interfaccia disco Cumana, così potei usare il sistema operativo SEDORIC; è stata la mia macchina principale per anni, anche se all'epoca avevo già un Atari STFM. Alcuni giochi moderni per Oric sono di qualità eccelsa, fanno sfigurare i titoli commerciali degli anni '80. Attualmente sto facendo il beta testing per una nuova ULA per l'Oric che permette di avere un'uscita DVI nitidissima; ci sono novità costanti anche per Oric, come per quasi tutti i computer di quell'epoca. Oggi abbiamo emulatori di drive e dispositivi di caricamento da SD che rendono tutto più semplice. Infine, mi piacciono molto i sistemi Dragon/Tandy, soprattutto per la CPU 6809; sto cercando di imparare l'assembler del 6809 sul mio CoCo2 PAL, giusto per tenere allenato il cervello!

3. DLM – Quali sono le sfide più comuni che affronti nel restauro e nella riparazione di hardware vintage? Nella tua esperienza, quali tipi di guasti tendono a verificarsi più spesso?





Vyper68 – La sfida principale è reperire i chip custom (IC) per questi computer o per le loro periferiche: penso all'ANTIC o al POKEY dell'Atari 800XL, o alla ULA dell'Oric. E non parliamo dei chip Commodore! Sono fuori produzione da una vita e l'unico modo per trovarli è "cannibalizzare" altre macchine, cosa che odio fare. Questo vale per quasi tutti i computer dell'epoca, che montavano componenti personalizzati; è senza dubbio il problema più grosso. Per quanto riguarda i guasti, oggi sui sistemi a 8 bit la causa numero uno è il RAM failure. Questo però è un male minore, dato che si riescono ancora a trovare chip 4164 o altre memorie come "New Old Stock" (NOS). Un altro problema frequente sono le saldature a freddo che causano malfunzionamenti difficili da scovare. Mi capitano molti Oric-1 con tastiere difettose e, regolarmente, il colpevole è una saldatura saltata sul PCB della tastiera.

4. DLM – Ti è mai capitato di portare a termine una cosiddetta "mission impossible"? C'è un successo di cui sei particolarmente orgoglioso?

Vyper68 – Difficile rispondere, perché all'inizio sembrano tutte missioni impossibili! Mi viene in mente un Oric Atmos che era andato in corto circuito. La porta di espansione è un connettore IDC con pin scoperti, posizionato proprio accanto all'ingresso 9V DC. Quando il computer è sulla scrivania non vedi i pin perché sono rivolti dalla parte opposta: se non stai attento, rischi di toccare i pin dell'espansione con lo spinotto dei 9V mentre cerchi l'ingresso. Mandare 9V su componenti che ne reggono solo 5 è un disastro. Ho dovuto dissaldare e testare ogni singolo chip logico, poi passare ai chip principali. In pratica l'ho ricostruito da zero! Il circuito PAL UHF aveva una PROM ormai introvabile, ma per fortuna oggi non serve più: quasi tutti usano l'uscita RGB con cavo SCART. Alla fine l'Oric è tornato in vita (UHF a parte) e ne sono rimasto davvero soddisfatto.

5. DLM – Come ti procuri i ricambi o i pezzi originali? Quanto è difficile trovarli a 30 o 40 anni dalla fine della produzione?

Vyper68 – Per la RAM mi affido a fornitori di fiducia scovati negli anni. Nel Regno Unito ci sono ancora piccoli negozi di elettronica che vendono componenti logici, RAM e CPU. Poi, se sei proprio disperato, c'è eBay, anche se lì ho solo un paio di venditori di cui mi fido. Tendo a evitare i siti



Fig. 3 – Riparazione di un Amstrad CPC

cinesi: c'è un grosso problema di chip contraffatti o rimarchiati. Mi è successo con controller per dischi, CPU, chip sonori AY e PIA. Come dicevo prima, il punto critico sono i chip custom: o trovi qualcuno che ha un fondo di magazzino (NOS), o qualcuno che ha smembrato una macchina, oppure devi comprarti tu stesso una macchina che faccia da "donatrice". In alcuni casi esistono sostituti moderni come il POKEYMAX o l'ORICULA, ma purtroppo la situazione è destinata a peggiorare col tempo. Fortunatamente il progresso dei rimpiazzamenti moderni basati su FPGA è una manna dal cielo, qualcosa che fino a pochi anni fa era impensabile.

6. DLM – Che ruolo gioca la community del retrocomputing nel tuo lavoro? Ti aiuta a mantenere viva la passione? Riesci a partecipare attivamente a forum o social media?

Vyper68 – Il retrocomputing e le riparazioni occupano gran parte della mia vita ora che sono in pensione; l'interazione con gli altri è fondamentale. Grazie a questo hobby ho stretto grandi amicizie. Spesso mi chiedono aiuto sui forum e cerco di fare il possibile; d'altra parte, io stesso chiedo consiglio agli altri perché non pretendo di avere tutte le risposte per ogni macchina. È uno scambio continuo. Evito i social media, preferisco i forum e i siti dedicati ai singoli sistemi.

7. DLM – Secondo te, quanto è importante preservare non solo l'hardware, ma anche il software e la documentazione (schemi tecnici, tabelle componenti, ecc.)?

Vyper68 – Credo sia vitale. Lo sviluppo dei microcomputer tra fine anni '70 e primi anni '80 è stato un momento cruciale per la tecnologia e la cultura. Per la prima volta





le persone avevano in casa qualcosa che prima era inaccessibile. Il boom del software, specialmente in Europa, è figlio di quell'epoca. La tecnologia corre, d'accordo, ma queste macchine sono pezzi di storia e vanno preservate, specialmente quelle ancora funzionanti. Lo stesso vale per manuali e schemi: abbiamo già perso troppa roba ed è un peccato. Per fortuna ci sono molte persone impegnate a tenere in vita queste macchine e a scrivere nuovo software.

8. DLM – Negli ultimi anni hai notato un ritorno di interesse per il retrocomputing tra le nuove generazioni? Chi è che di solito ti cerca per consigli o riparazioni?

Vyper68 – C'è moltissimo interesse per il retrogaming oggi. C'è sempre stato un "nocciolo duro" di appassionati, ma oggi la gente ha più disponibilità economica e ha iniziato a ricomprare i computer dell'infanzia, credo per pura nostalgia. L'effetto collaterale negativo è che i prezzi sono schizzati alle stelle. Le mini-console e l'emulazione hanno permesso a molti di capire "di cosa si trattasse". Basta guardare YouTube. Chi si rivolge a me? Spesso persone che dicono: "Ho comprato questo computer su eBay ma non va, aiuto!". Di solito sono neofiti che hanno preso una macchina venduta come "non testata", che tradotto dal linguaggio di eBay significa "l'ho attaccata, non si è accesa e quindi scrivo che non è testata". I veterani sanno a cosa vanno incontro, ma anche loro bussano alla mia porta per i restauri.

9. DLM – Cosa ne pensi dell'emulazione software e delle riproduzioni basate su FPGA? Aiutano o ostacolano la conservazione dell'hardware originale?

Vyper68 – Pirateria a parte, credo che l'emulazione sia positiva. Alcuni sistemi sono rarissimi e costosi, fuori dalla portata dei più. L'emulazione permette di scoprirli. Inoltre, è fondamentale per preservare il software: molti titoli sono già andati perduti. Usare un FPGA o un emulatore permette anche di non "usurare" l'hardware originale, il che è un bene, anche se far girare il codice sulla macchina vera ha un fascino che i sistemi moderni non avranno mai. Le riproduzioni come il C64 Mini o l'Atari Mini non sono come l'originale, ma d'altronde non pretendono di esserlo. In breve: approvo l'idea.

10. DLM – Se potessi restaurare un pezzo leggendario, magari per un museo o per la tua collezione, quale



Fig. 4 - Espansione RAM per 800XL

scegliresti e perché?

Vyper68 – Mi piacerebbe mettere le mani su un Computers Lynx, un SAM Coupé o un Atari Falcon030. Possedevo un Lynx e un SAM negli anni '90, ma purtroppo subii un furto in casa e gran parte del mio vecchio kit informatico sparì nel nulla; quei due erano tra la refurtiva. Mi piacerebbe lavorare sul Falcon030 perché è stato l'ultimo computer di Atari e sembrava una macchina fantastica. Ho usato Hatari per emularne uno e pare sia una vera bestia! Oggi i prezzi di tutti e tre sono folli, quindi dubito che ne possederò mai più uno (o che avrò mai un Falcon), ma come dicevo c'è sempre l'emulazione: Simcoupe è un ottimo emulatore per il SAM e Jynx è un emulatore in Java per il Lynx.

11. DLM – Quali strumenti hardware e software utilizzi solitamente per diagnosticare i guasti? Qual è il tuo metodo d'intervento principale?

Vyper68 – Ho il classico kit che ci si aspetta di trovare sul banco di un tecnico elettronico: saldatore, pompa dissaldante, treccia e una stazione dissaldante professionale. Spero di prendere presto una stazione ad aria calda per lavorare su sistemi e console più moderni. Ho anche un multimetro, una sonda logica e un piccolo oscilloscopio. Il mio punto di partenza è sempre il multimetro. Applico sempre il mio mantra: ASAP (Always Start At the Power - parti sempre dall'alimentazione). Inizialmente testo a voltaggio ridotto con un alimentatore moderno. Una volta superato il regolatore, potrei bypassarlo del tutto e iniettare 5V direttamente sulla scheda. Quando hai i 5V su tutti i chip, puoi dare un'occhiata al display per farti un'idea di cosa non vada e faccio anche il "test del dito" sulla RAM:





se un chip è guasto, di solito scotta in modo incredibile. Di solito comincio così. E poi...

12. DLM – Quando lavori su schede madri storicamente significative, preferisci preservarle allo stato originale o effettui modifiche per ripararle?

Vyper68 – Cerco sempre di preservare l'originalità il più possibile, ma esistono anche le modifiche di servizio ufficiali rilasciate all'epoca dai produttori. Quelle le installo sempre, a meno che non mi venga chiesto il contrario. Per esempio, l'Oric-1 ha un difetto per cui la ROM finisce col distruggere il chip sonoro AY (secondo Tangerine Computers): rilasciarono un bollettino tecnico per risolvere il problema. Se trovo un Oric che ne è privo, applico sempre il fix. Sono modifiche ufficiali che, ai miei occhi, rendono la macchina originale e al contempo migliore. Non perdo tempo invece con le modifiche amatoriali trovate online, almeno sulle macchine dei clienti.

Potrei modificare uno dei miei vecchi computer, come il VIC-20, per supportare l'S-Video e avere un'immagine migliore. Personalmente mi piace avere una macchina "stock" (originale) e una modificata per godermi entrambi i mondi. Non è sempre possibile, quindi alcuni miei computer hanno upgrade per migliorare l'uscita video e collegarli alle TV moderne, ma non lo farei mai per qualcun altro senza una richiesta specifica. In fondo, faccio quello che vuole il cliente (entro i limiti del ragionevole).

13. DLM – Come gestisci l'ossidazione dopo decenni di inattività? Quali sono le tecniche di manutenzione che consigli a chi riceve una macchina riparata da te?

Vyper68 – Se i piedini dei chip sono ossidati dal calore o dal tempo, uso un abrasivo molto leggero per pulirli; se lo zoccolo è in cattive condizioni, lo sostituisco. Uso detergenti per contatti su interruttori e connettori di tipo edge. Per questi ultimi (e per le cartucce) uso anche una gomma per cancellare: funziona sorprendentemente bene. Per quanto riguarda i case, non tratto lo scolorimento della plastica: non sono convinto che il retrobrighting faccia bene ai polimeri a lungo andare. Se un case è ingiallito, resta così. Il mio VIC-20 ne è l'esempio: è giallissimo, ma non lo tratterei mai!

Come dicevo, la RAM è il guasto più comune sugli 8 bit, insieme alle saldature a freddo. Sulle console può capitare di tutto, avendo molti chip custom e ASIC che possono

saltare. Se comprate una macchina su eBay, partite dall'alimentatore e controllate i voltaggi. Se potete, usate un alimentatore esterno per testare i 5V sulla scheda bypassando quello interno, pulite tutto bene e controllate la continuità dei circuiti.

14. DLM – Ti è mai capitato di dover ricreare componenti non più in produzione o di migliorarli con design moderni?

Vyper68 – È un ambito in cui non mi sono mai addentrato. La creazione di rimpiazzi moderni in FPGA per vecchi chip non mi attira particolarmente; ci sono persone molto più in gamba di me che lo fanno. Preferisco riparare piuttosto che progettare.

15. DLM – Utilizzi emulatori hardware FPGA o strumenti di reverse engineering per testare parti obsolete? Quali sono i tool diagnostici più veloci?

Vyper68 – Non uso hardware FPGA, ma utilizzo un tester per RAM e uno per chip logici: sono strumenti diagnostici molto rapidi ed efficaci.

16. DLM – Come gestisci la preservazione dei dati da supporti magnetici fragili (floppy, cassette, hard disk)?

Vyper68 – Dipende dal supporto. Per i floppy uso hardware specifico come Greaseweazle, SIO2PC o XUM1541. Per le cassette cerco di copiare il file usando software come Audacity, pulendo poi l'input audio. Non lo faccio più molto spesso, dato che ormai quasi tutti usano dispositivi moderni che emulano i drive, come Fujinet, Pi1541 o TZXDuo.

17. DLM – Cosa ne pensi dell'uso di condensatori moderni per sostituire gli originali? Riparare oggi è più facile o più difficile rispetto all'epoca d'oro di queste macchine?

Vyper68 – Sostituire i condensatori con componenti moderni di alta qualità è un'ottima idea. Queste macchine erano costruite al risparmio per massimizzare il profitto e i componenti non erano sempre eccelsi. Sostituirli significa farli girare per altri 40 anni.

All'epoca le riparazioni erano più semplici per certi versi: potevi chiamare il centro assistenza Atari o Commodore e comprare un POKEY o un SID nuovo di zecca. Oggi è impossibile. Di contro, oggi grazie a Internet abbiamo una quantità di informazioni incredibile a portata di click e gli strumenti moderni sono migliori e più accessibili.





Quindi, direi che vantaggi e svantaggi si compensano.

18. DLM – Quali linguaggi di programmazione o software utilizzi per interfacciarti con l'hardware legacy, specialmente per il burning di EPROM o aggiornamenti firmware?

Vyper68 – Non uso molto i linguaggi di programmazione, a parte il classico BASIC e un po' di FORTH, dato che quasi tutti i computer avevano un interprete BASIC integrato. È utile per scrivere piccoli programmi per controllare la memoria o le variabili di sistema. Mi diletto con l'assembler del 6809 per fare le stesse cose più velocemente, ma solo su Dragon e Tandy CoCo. Ho un programmatore di EPROM per fare il dump delle ROM o scriverne di nuove se l'originale è guasta o se il cliente desidera una versione aggiornata.

19. DLM – Come documenti il tuo lavoro? Prendi appunti, disegni schemi, scrivi guide o condividi codice e contenuti multimediali? Sei geloso delle tue tecniche o ti piace condividere conoscenze ed esperienze con gli altri?

Vyper68 – Scatto foto e segno sempre cosa sto facendo e i risultati ottenuti, così ho un riferimento pronto se in futuro mi ricapita un problema simile. Pubblico i risultati sui forum con foto e scrivo articoli di approfondimento. Non mi considero una persona speciale, sono solo uno con la passione per l'hardware; non sono geloso dei miei processi, perché sono sicuro che molti altri facciano lo stesso. Condivido molto volentieri quello che so: per me, più siamo e meglio è.

20. DLM – Quanto ti diverti davvero a riparare sistemi e diagnosticare guasti, e quanta soddisfazione provi quando risolvi un problema che all'inizio sembrava insormontabile? Ti è mai capitato di dover gettare la spugna perché i guasti erano troppi o il costo non valeva l'impresa?

Vyper68 – Mi diverto moltissimo, specialmente ora che sono in pensione e ho più tempo da dedicare a quest'attività. Non lo faccio per business, mi piace proprio il processo. Riparare un dispositivo mi dà una soddisfazione enorme: partire da un sistema totalmente morto e finire con uno funzionante è impagabile. Solo una volta ho avuto problemi seri: un Oric-1 che ero riuscito a far ripartire, ma che non voleva saperne di caricare da cassetta. Nonostante tutti i tentativi, non caricava nulla; alla fine non valeva più la

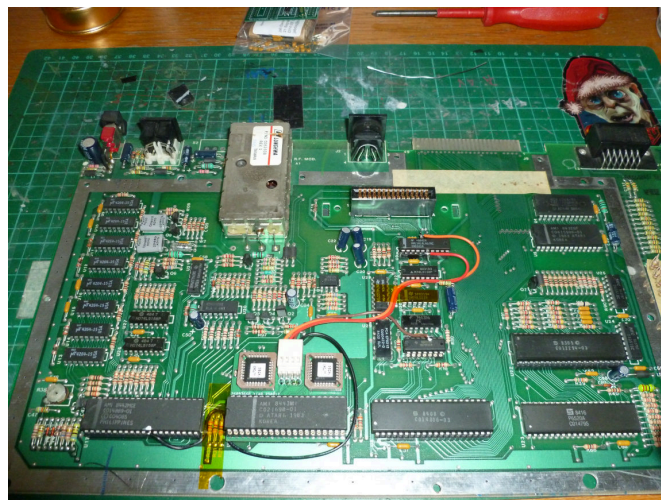


Fig. 5 - Motherboard Atari con espansione 512KB

pena spenderci altri soldi, così è diventato un sistema utilizzabile solo con espansione Erebus o disco. Considerando quanto fosse inaffidabile il caricamento via nastro dell'Oric, non è stata una gran perdita!

Conclusioni

Ringraziamo Richard per aver condiviso il suo punto di vista e per averci raccontato qualcosa dei suoi viaggi nella riparazione dei retrocomputer. Nel mondo dei tecnici elettronici non è sempre facile trovare qualcuno disposto a condividere storie, tempo e competenza, tutti elementi essenziali per mostrare il "dietro le quinte" delle riparazioni, le sfide quotidiane e, soprattutto, il valore di questo mestiere. È proprio la passione per i nostri retro-computer che spinge hobbisti e appassionati come Richard a continuare la missione di preservare e riportare in vita quei sistemi vintage che, negli anni, ci hanno regalato così tanto divertimento e conoscenza. Se anche voi avete a casa computer o periferiche del passato, fate un salto sui forum e sui canali social frequentati da Vyper68. Troverete consigli, documentazione, storie di riparazioni e un immenso bagaglio di esperienze condivise. Ma soprattutto, entrerete a far parte di una community accogliente dove scambiare messaggi, approfondire le vostre conoscenze e confrontarvi con Vyper68 stesso e molti altri entusiasti del retrocomputing e dell'elettronica.

Forums & Links

SpectrumComputing.co.uk, AtariAge.com, Atari-Owner.com/club, Forum.DefenceForce.org, Archive.WorldofTheDragon.com, StarDot.org.uk, ColorComputerArchive.com, Lemon64, CPCWiki.eu/Forum
FB Groups: OricOwners, SpectrumForEveryone





Collezionare o accumulare?

di Roberto Lanciotti

Ho scritto una riflessione personale sul collezionismo, su quello che per anni ho vissuto come passione e che oggi vedo anche in un'altra luce: quella sottile linea tra preservare e accumulare, tra memoria e possesso. È un pezzo molto personale, nato da un percorso lungo e da alcune consapevolezza arrivate solo di recente. Se qualcuno si riconosce in queste dinamiche o ha vissuto qualcosa di simile, magari può trovare uno spunto di riflessione.



Per molti anni ho creduto che il collezionismo fosse una delle forme più alte della passione: una cura verso la memoria, un modo per preservare le storie di un'epoca destinate altrimenti a scomparire.

Solo oggi, guardando indietro, mi rendo conto di quanto sia sottile il confine tra la volontà di custodire e il bisogno di possedere.

Quella che per lungo tempo ho chiamato preservazione, a tratti, è diventata accumulo. E l'accumulo, quando cresce silenziosamente negli anni, non occupa soltanto spazio fisico: finisce per prendere tempo, energie, relazioni e, in qualche modo, una parte di noi stessi. Chi colleziona non raccoglie semplicemente cose: raccoglie le memorie, le testimonianze materiali di passioni personali e collettive. In questo senso, il collezionismo ha sempre avuto una dimensione culturale molto profonda. È un gesto che parla di gusto, di ricerca e di conoscenza.

Eppure, proprio in questa apparente nobiltà, si nasconde un'ambiguità. L'esperienza personale mi ha insegnato



quanto sia fragile il confine tra la cura della memoria e la logica dell'accaparramento. All'inizio, ogni nuovo pezzo della collezione mi restituiva una certa soddisfazione: la sensazione di aver salvato qualcosa, di averlo riconosciuto, scelto e portato nel mio spazio.

Era un piacere che non riguardava solo l'oggetto in sé, ma ciò che esso rappresentava. Con il tempo, però, quella soddisfazione ha iniziato a rivelarsi effimera.

L'oggetto smetteva presto di essere scoperto e diventava abitudine. Così la tensione si spostava immediatamente verso la ricerca successiva, verso il prossimo acquisto, verso il desiderio di completare, superare, possedere ancora. È stato però soltanto in questi giorni che tutto questo mi è apparso con chiarezza.

Nel garage dove conservo gran parte della mia roba ho notato alcune macchie sul muro. In un primo momento ho pensato fosse muffa, ma osservando meglio si trattava di una sostanza biancastra, tipica dell'umidità. In realtà avevo un'infiltrazione d'acqua da qualche parte, che era scesa fino all'altezza dei miei scaffali. Quella parete mi ha costretto a fermarmi. Ho dovuto smontare gli scaffali e svuotarli dal materiale accumulato in anni e anni: cartoni di giochi accatastati, console, libri, tantissime cassette, accessori e doppioni di ogni tipo.

Ed è stato proprio in quel momento, mentre toglievo scatole su scatole, che finalmente ho fatto due più due. Ho capito di aver passato una vita a raccogliere. Troppo!! Mi sono rivisto in un gesto quotidiano diventato quasi automatico: ogni mattina, la prima cosa da fare era cercare l'occasione, l'affare, lo sconto irripetibile. Immagino che molti miei "colleghi" collezionisti (scusate, non voglio offendere nessuno) si riconosceranno in questa dinamica. La giornata iniziava spesso con la ricerca del pezzo giusto, del lotto conveniente, della rarità comparsa improvvisamente su qualche mercatino o annuncio. Era diventato un riflesso. Solo adesso mi rendo conto di quanto tempo, energia e attenzione questa abitudine abbia assorbito. Tempo sottratto alla famiglia, alle persone vicine, alle cose che





contano davvero. Non era sbagliato avere una passione. Era sbagliata la sproporzione. Avrei potuto farlo in modo più misurato, più parsimonioso, più consapevole.

A questo si accompagna anche un pensiero più doloroso: in alcuni momenti, lasciandomi trascinare troppo da un ego che finiva per occupare troppo spazio, ho finito per perdere anche amici, persone che a un certo punto si sono allontanate. Non è mai stato qualcosa di voluto, e forse proprio per questo oggi pesa ancora di più.

Senza accorgermene, mi ero concentrato troppo sulle mie ricerche e sulle mie priorità, finendo per dedicare meno attenzione a chi mi stava accanto. Molti incontri sono stati autentici, altri invece si sono sviluppati dentro dinamiche più ambigue: una sorta di compiacimento reciproco, alimentato dal desiderio di accumulare e possedere.

Talvolta mi sono avvicinato alle persone con un atteggiamento interessato, ma spesso si trattava di una dinamica reciproca perché anche dall'altra parte emergeva una forma di interesse. Il fine era, in qualche modo, la speranza che prima o poi quella persona potesse vendere, scambiare o addirittura regalare il pezzo mancante per la collezione.

Eppure, la spinta originaria, almeno per me, non è mai stata soltanto il possesso, quello che in fondo ho sempre cercato di trasmettere con il mio lavoro, era un senso di felicità semplice, quasi infantile: una forma di serenità quieta, fatta di calma interiore e di piccole cose che rendono tutto più leggero. Volevo ritrasmettere quel tipo di felicità che ricordavo avere da bambino, quando non serviva molto per stare bene, quando bastava condividere qualcosa per sentirsi in pace con tutto il resto.

Il problema è che, a un certo punto, preservazione e collezionismo iniziano a scontrarsi. Si parte con un'idea nobile: conservare, catalogare, fare dumping, trasferire contenuti, cioè evitare che la memoria tecnica e culturale vada perduta. Ma spesso, dietro questa giustificazione si insinua l'egoismo e, di conseguenza, l'accumulo.

È lì che il gesto culturale rischia di trasformarsi in abitudine compulsiva. Per fortuna, però, con il tempo si cresce e si

arriva a capire che esiste una differenza tra ciò che è davvero utile alla conservazione e ciò che invece andrebbe semplicemente lasciato andare. Va bene avere qualche pezzo di scorta, ciò che serve davvero per il dumping, per il recupero dei dati o per la preservazione dei supporti. Ma tutto il resto, spesso, non è memoria. È peso. Forse è proprio questo il nodo più interessante: il collezionismo, quando degenera, non produce solo eccesso materiale, ma anche impoverimento relazionale. L'oggetto, da simbolo di memoria, diventa schermo. Oggi ho capito che il vero discrimine non è tra collezionare e non collezionare, ma tra dare significato e cercare compensazione. Nessuna collezione, per quanto preziosa, può sostituire il valore del tempo vissuto con chi ci sta accanto. Ed è proprio questo che, a un certo punto, avevo dimenticato: da ragazzi si condivideva tutto e con poco si era felici. Si stava insieme, semplicemente. Se oggi sento il bisogno di mettere tutto questo nero su bianco, è anche per rivolgere un pensiero sincero a tutte quelle persone che, nel corso degli anni, forse non hanno ritrovato in me la persona che si aspettavano. A chi, per questo, ha scelto di allontanarsi. Oggi, con maggiore lucidità, porto con me il rammarico per ciò che avrei potuto fare e non ho fatto, per le attenzioni mancate, per il tempo non dato, per le persone che forse avrebbero meritato una versione migliore di me. Potevo fare di più, questo sì. Ma la vita è fatta anche di questo: di scelte prese in un certo momento, di priorità sbagliate, di entusiasmi che ti portano lontano senza che te ne accorgi. Eppure qualcosa di buono, alla fine, credo di averlo lasciato. Qualche pezzo salvato, qualche frammento di memoria preservato, qualche passione vissuta fino in fondo, anche se a volte in modo sbilanciato. Ma poi servirà tutto questo? Come diceva quella canzone, "Lo scopriremo solo vivendo".

L'animo umano è strano: si perde facilmente nelle proprie cose, ma col tempo per fortuna si riesce anche a guardarsi indietro e dare un senso, almeno parziale, a quello che è stato. E forse, se questo articolo avrà un senso, sarà anche quello di far riflettere qualcun altro sul proprio percorso, su ciò che si accumula e su ciò che invece conta davvero.

DumpClub64:

- <https://www.dumpclub64.it/>

Link al 67° Dumping:

- <https://www.dumpclub64.it/dumping/67-dumping/>





Apple compie 50 anni: mezzo secolo di “Think Different”

di Carlo Nithaiah Del Mar Pirazzini

Il 1° aprile 1976, in un garage di Los Altos, nasceva una delle storie più influenti dell'informatica moderna. Cinquant'anni dopo, Apple festeggia il proprio anniversario non solo con cifre da capogiro – quasi 4.000 miliardi di dollari di capitalizzazione, miliardi di dispositivi in circolazione – ma con una domanda scomoda: cosa significa davvero “Think Different” nel 2026?

Dal circuito “nudo” al futuro

Nel 1976 l'Apple I è, letteralmente, una scheda nuda. Niente case, niente tastiera, niente monitor, solo un circuito da infilare in un contenitore e collegare a una TV. È Wozniak a disegnarlo, con il suo approccio minimalista e geniale; ma è Jobs a capire che quella scheda può essere venduta, confezionata, trasformata da curiosità per hacker a prodotto.

Con l'Apple II arriva la svolta! Un case in plastica, tastiera integrata, colori, unità floppy. È uno dei primi computer che puoi immaginare su una scrivania non aziendale, e con VisiCalc diventa improvvisamente uno strumento da ufficio desiderato quanto una calcolatrice di fascia alta. Già qui si intravedono i tratti che Apple non abbandonerà più, cura per l'oggetto, attenzione maniacale all'esperienza d'uso, convinzione che dietro ai numeri debba esserci una storia.

Il Macintosh, nel 1984, alza la posta: interfaccia grafica,



Il logo per celebrare i “primi” 50 anni

mouse, tipografia degna di una tipografia “vera”. L'iconico spot “1984” non vende solo un prodotto, ma un'idea di ribellione contro l'omologazione informatica dominata da IBM. Anche quando gli utili iniziano a scricchiolare e i progetti non decollano, quell'identità “altra” rispetto al resto del settore resta impressa.

Cadute, ritorni e il miracolo della seconda vita

Dopo l'uscita di scena di Jobs a metà anni '80, Apple entra in quella che molti chiamano “la decade smarrita”. Troppe linee di Macintosh, strategie confuse, Windows che avanza destabilizzano l'azienda e i suoi clienti affezionati. I conti peggiorano, la quota di mercato crolla, e a metà anni '90 l'azienda flirta pericolosamente con il fallimento (e ci arriva davvero vicino).



Apple I: le schede assemblate in diversi case artigianali





Il successo di Apple II

Paradossalmente è proprio quella crisi a preparare il terreno per la rinascita. L'acquisizione di NeXT nel 1996 riporta Jobs in casa, insieme a una base tecnologica (NeXTSTEP) che diventerà il cuore di Mac OS X. La cura Jobs è brutale come lo è il suo approccio col personale. Taglio di prodotti, chiusura del programma cloni, concentrazione su pochi modelli chiari. E, soprattutto, un ritorno all'ossessione per il design e l'immagine con la campagna "Think Different".

L'iMac colorato del 1998 non è solo un computer ma il Manifesto visivo che dice "Apple è tornata". L'iPod, tre anni dopo, dimostra che l'azienda può uscire dal recinto del personal computer e ridisegnare mercati apparentemente maturi come quello musicale. Con l'iPhone nel 2007, il concetto di "telefono" viene fuso con quello di computer tascabile e piattaforma applicativa. Da lì in avanti, l'azienda entra in una traiettoria di crescita che la porta, secondo i dati riportati dall'articolo, a fatturati annui nell'ordine delle centinaia di miliardi e utili a tre cifre in miliardi di dollari.

"Think Different" come filo conduttore

La celebre campagna del 1997 non è solo uno spot nostalgico coi volti di Einstein e Gandhi, ma una dichiarazione d'intenti... Apple vuole posizionarsi dove tecnologia e

arti liberali si incontrano, laddove il valore non sta nei benchmark ma in come un prodotto cambia il lavoro creativo e la vita quotidiana.



Lo splendido iMAC

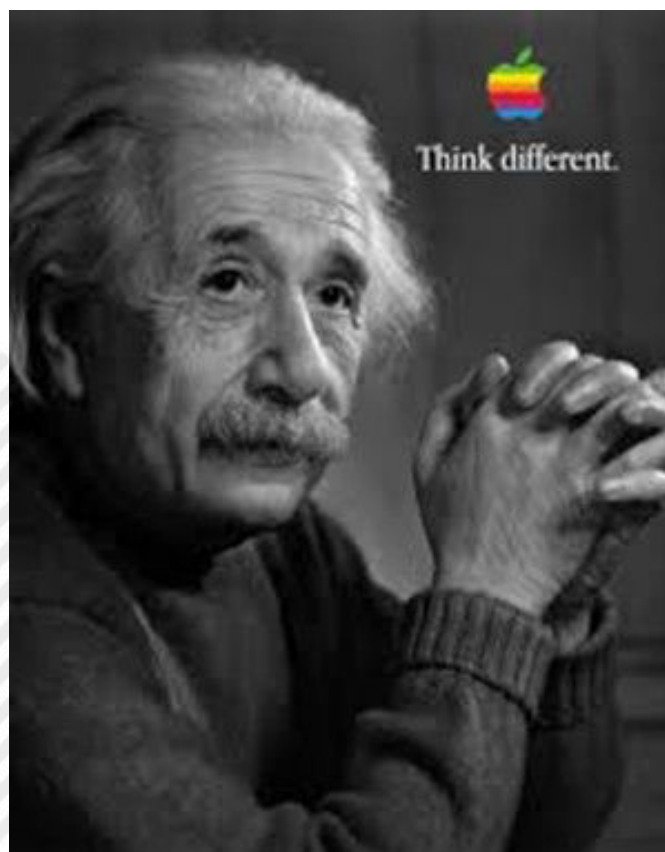




Jobs presenta il NeXT e il sistema operativo NeXTSTEP

Questo approccio si vede nei dettagli dei suoi prodotti: dalle interfacce coerenti tra Mac, iPhone e iPad, alle API pensate per dare agli sviluppatori accesso profondo all'hardware senza costringerli a reinventare la ruota, fino alle funzioni di accessibilità che permettono a non

vedenti, ipovedenti o persone con disabilità motorie di usare gli stessi dispositivi di tutti. È anche il motivo per cui l'App Store diventa, nel bene e nel male, il cancello d'accesso a un'economia di milioni di app: controllo rigido, ma anche una piattaforma stabile su cui costruire aziende intere.



La campagna "Think different"

L'era Cook ... oltre l'hardware

Alla scomparsa di Steve Jobs nel 2011, l'ombra del "dopo" è lunghissima. Tim Cook, ex responsabile Operations, non ha il carisma del fondatore, ma porta in dote una capacità organizzativa che permette ad Apple di fare ciò che prima era impensabile, mettere in mano ad oltre due miliardi di persone dispositivi complessi e mantenerli aggiornati per anni.

Sotto la sua guida, il perimetro si allarga. Vediamo l'arrivo di Apple Watch e AirPods che trasformano il corpo e le orecchie in estensioni naturali dell'ecosistema; i servizi (App Store, iCloud, Apple Music, TV+, Pay, Fitness+, Arcade) diventano una seconda gamba del business, molto redditizia. L'introduzione di Apple Silicon porta CPU e GPU proprietarie su Mac con salti di prestazioni e autonomia che ridefiniscono le aspettative del portatile "serio".

In parallelo, l'azienda si posiziona con forza su temi come





Tim Cook mostra il futuro

privacy, accessibilità e sostenibilità. Datacenter alimentati da rinnovabili, obiettivi di neutralità carbonica lungo la supply chain, sistemi di tracciamento limitati rispetto alla concorrenza, funzioni di sicurezza e salute che vanno dal rilevamento delle cadute alla fibrillazione atriale.

Cinquant'anni: tra celebrazione e banco di prova

Arrivati al 50° anniversario, Apple sceglie una celebrazione che mescola spettacolo e narrazione. Eventi dal vivo in store iconici come Grand Central a New York, performance di artisti come Alicia Keys, una probabile chiusura in grande stile ad Apple Park, iniziative "Today at Apple" dedicate alla creatività e un archivio social dedicato alle storie di utenti e sviluppatori. In una lettera pubblica, Tim Cook ringrazia apertamente chiunque abbia contribuito, dal primo dipendente che saldava schede in garage agli sviluppatori indie che oggi pubblicano su App Store, ribadendo che ogni prodotto è solo l'inizio di una storia che viene scritta da chi lo usa.

"Think Different"... nel 2026

Il bello (e il difficile) a 50 anni è che Apple non può più limitarsi a "pensare diverso" rispetto agli altri, deve dimostrare di saper pensare diverso anche rispetto a sé stessa. Questo significa, prima di tutto, evitare di trasformarsi in un ciclista che pedala sempre sulla stessa marcia (il classico iPhone annuale con aggiornamenti incrementali) aspettando che sia la strada a cambiare pendenza. Vuol dire continuare a usare l'integrazione verticale come leva per semplificare davvero la vita all'utente, e non solo per blindare l'ecosistema. E, soprattutto, significa mettere l'AI al servizio della persona (accessibilità, creatività, privacy) invece che usarla soltanto come motore per ottimizzare il fatturato. Il meglio deve ancora venire? Chi vivrà vedrà!





Le mascotte dimenticate (o quasi) del mondo Nintendo

di Carlo Nithaiah Del Mar Pirazzini e Giampaolo Moraschi

Konkichi, Stanley the Bugman, Excite-kun, Mach Rider... Nomi non di certo familiari come Mario e Link, ma che hanno lasciato un'impronta nel mondo dei videogames Nintendo. Scopriamoli assieme.

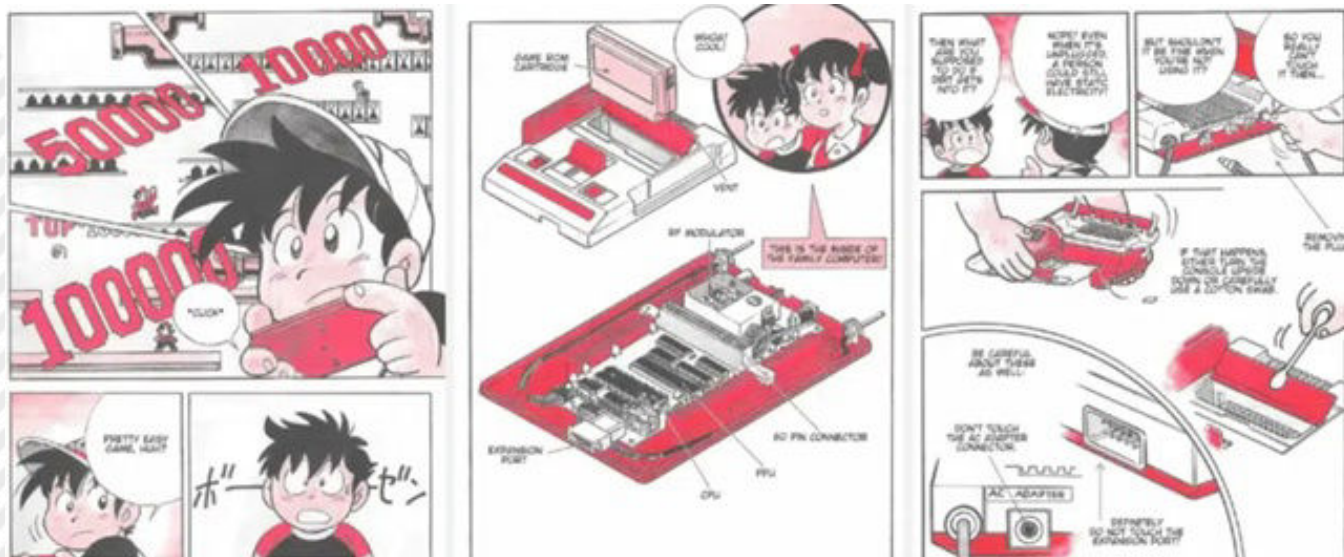
Prima di Mario c'era ... Konkichi!

Akira Toriyama. Nei primi anni '80 Mario si affacciava nel mondo dei videogiochi con il semplice nome di Jumpman. Era amato, ma serviva qualcosa di iconico alla casa nipponica. Konkichi apparve nei manuali del Famicom nel 1983. In due libretti ad esser precisi. Il giovane Konkichi spiegava, attraverso illustrazioni e dialoghi, come utilizzare al meglio



Ebbene sì! Prima di Super Mario, la mascotte ufficiale si chiamava Konkichi ed era molto simile ai personaggi di

la console. Un ragazzino vivace, capelli corti e un buffo cappello. Una mascotte pensata per i più piccoli e che



Famicom Manga Instruction manual... il simpatico libretto dove appare Konkichi





Stanley the Bugman

strizzava l'occhio ai personaggi di due grandi mangaka... Akira Toriyama e Osamu Tezuka. Nonostante l'aspetto buffo e il bel character design, Konkichi sparì quasi subito... il motivo? Il boom di Super Mario.

Stanley the Bugman

Protagonista del gioco "Donkey Kong 3", il suo nome è spesso confuso con altri personaggi, e il suo impatto sulla cultura videoludica è stato limitato, nonostante il gioco sia stato un successo commerciale. Doveva sostituire Mario nella serie con DK, ma fu presto scordato nonostante l'aspetto buffo. Riapparirà decenni dopo come trofeo in Smash Bros.

Excite-Kun

La mascotte della serie Excitebike, è un personaggio stilizzato che rappresenta il motociclista. Sebbene sia apparso in molte pubblicità e copertine di giochi, non ha mai avuto uno sviluppo di personaggio paragonabile a quello di altre icone Nintendo.



Mach Rider

Il personaggio di Mach Rider, protagonista dell'omonimo gioco di corse futuristico, è noto per la sua ambientazione post-apocalittica e per il suo stile unico. Nonostante il gioco abbia riscosso un certo successo, il personaggio non è diventato un'icona come altri protagonisti Nintendo.



Questi personaggi, sebbene meno famosi, contribuiscono a delineare la ricchezza e la varietà del mondo Nintendo, dimostrando che anche personaggi meno noti possono essere parte di una storia videoludica più ampia.





AmiGame Jam 2025 – Ritorno all'Arcade

di Carlo Nithaiah Del Mar Pirazzini e Giampaolo Moraschi

Questa nuova edizione si è dimostrata davvero interessante, con 20 titoli di buona qualità. Ma prima di addentrarci sui primi cinque titoli classificati parliamo un po' della Jam.

Le origini

L'AmiGameJam è una competizione internazionale di sviluppo dedicata ai nuovi giochi per Amiga, ospitata su itch.io e organizzata da Amiga Cammy (Moya Jackie McGeough), che nel giro di pochi anni è diventata una sorta di "stato di salute annuale" della scena homebrew del Commodore. L'edizione 2025, ad esempio, ha raccolto 20 titoli inviati tra il 31 maggio 2025 e il 28 febbraio 2026, tutti rigorosamente giocabili su hardware reale o emulatore, e ha generato ben 809 voti da parte del pubblico, con una media di oltre quaranta valutazioni a gioco: numeri che mostrano una community non solo nostalgica, ma ancora attiva e curiosa.



**La simpatica Moya Jackie McGeough
aka Amiga Cammy**

Il funzionamento è semplice ma molto efficace. Gli sviluppatori registrano il proprio progetto nella pagina della jam entro la scadenza, caricando il gioco completo o una versione sostanziosa che rappresenti bene l'idea. Una volta chiuse le iscrizioni, la palla passa alla community: per alcuni giorni i partecipanti e i giocatori possono scaricare, provare e votare tutti i titoli in gara direttamente da itch.io, lasciando punteggi e, spesso, commenti e suggerimenti. Alla fine del periodo di voto la piattaforma calcola una media dei rating e stila la classifica finale: nel 2025 il primo posto è andato a Echodrome di PixelPlop, seguito a ruota da Double Baboon Ninja di DanLabG, mentre Wings of Fear II e Creepy Crawlers Goliath Reloaded hanno chiuso il podio ex aequo al terzo posto, seguiti da una sfilza di progetti che spaziano dai platform

agli shooter, dai puzzle ai racer, fino a esperimenti cross-platform come la demo di Cogs N Logs per Amiga e Megadrive.

Lo scopo dell'AmiGameJam va però oltre la semplice graduatoria. Nasce per tenere viva la creatività intorno all'Amiga, offrendo un palcoscenico condiviso dove coder, grafici e musicisti possono mettere in mostra il proprio lavoro sapendo che verrà provato, discusso e confrontato con quello degli altri. Ogni edizione diventa così una fotografia della scena: da un lato ci sono i nomi noti che tornano con nuovi progetti o work in progress più rifiniti, dall'altro emergono piccoli team o sviluppatori solitari che trovano nella jam il modo più diretto per farsi notare. Il risultato è un catalogo variegato, in cui convivono giochi completamente originali, reinterpretazioni di generi classici e demo che fungono da trampolino per future produzioni commerciali o fisiche.



Il logo con il tema della competition di quest'anno

In questo senso l'AmiGameJam non è solo una gara di punteggi, ma una sorta di laboratorio collettivo. Il vincolo di una deadline spinge gli autori a chiudere un prototipo funzionante invece di perdersi in sviluppi infiniti, mentre il confronto pubblico con gli altri titoli aiuta a misurare il proprio lavoro sul piano di grafica, sonoro, giocabilità e presentazione generale. Per chi gioca, invece, la jam rappresenta una vetrina concentrata di novità: nel giro di pochi giorni è possibile scoprire venti giochi diversi, seguire il passaparola, confrontare impressioni e vedere come l'Amiga continui a ispirare idee nuove nonostante l'età. Un appuntamento fisso per chi ama l'Amiga: un evento che celebra il passato della macchina, ma con lo sguardo rivolto al presente e al futuro, trasformando ogni edizione in una piccola festa della creatività a 16 bit dove, al di là dei podi, il vero premio è vedere quanti giochi nuovi continuano a nascere per una piattaforma che ufficialmente





non esiste più da decenni, ma che su itch.io mostra ancora numeri e vitalità sorprendenti.

I primi cinque classificati ... e cosa ne pensa la redazione di RMW!!!

Echodrome - Pixelpop - <https://pixelplop.itch.io/echodrome-amigajam2025>



Primo classificato alla Jam, Echodrome è uno shooter arena fragoroso e divertente dall'aspetto minimalista ma "acido". Gira su Amiga OCS/ECS con 1 solo MB di memoria richiesta. Ondate e ondate di nemici da blastare e boss finale da riempire di colpi. Il level design è interessante e la difficoltà ben calibrata. Il podio è meritato.

Giocabilità: 90% - Longevità: 90%

Double Baboon Ninja - DanLabG - <https://danlabg.itch.io/double-baboon>



Babbuini, piattaforme, luci al neon anni '80, alieni pazzoidi... Un grandissimo gioco e il nostro "vincitore". Per ora è solo una preview con 4 livelli di gioco, ma è già un titolo coi fiocchi che mostra un grandissimo level design e una giocabilità coi fiocchi. Esteticamente è un capolavoro!

Giocabilità: 92% - Longevità: 80%

Wings of Fear II - Roartj - <https://roartj.itch.io/wings-of-fear-ii>



In questo numero vi abbiamo recensito il primo capitolo, qui vediamo una demo del secondo capitolo che si mostra con un nuovo scrolling (orizzontale al posto di verticale) e un notevole miglioramento nell'uso dei colori e della velocità di gioco.

Giocabilità: 85% - Longevità: 80%

Creepy Crawlers Goliath:Reloaded - Digilicious Nibbles - <https://digiliciousnibbles.itch.io/creepy-crawlers-goliath-reloaded>



Altro titolo da podio e shoot'em up solido e ben realizzato. Divertentissimo e temibilissimo! E' un gioco davvero da "sala giochi" che punisce anche il minimo errore d'attenzione. Gradevolissima la grafica.

Giocabilità: 90% - Longevità: 85%





Cake Rush - Hambo3 - <https://hambo3.itch.io/cake-rush>



Party game competitivo e giocabilissimo. Perfetto da giocare in multiplayer e dotato di quel sadismo tipico di questo genere. Fare il pasticcere in questo titolo è letale, sopravvivere è quasi impossibile.

Bello!

Giocabilità: 90% - Longevità: 90%

Gli altri classificati alla Jam sono:

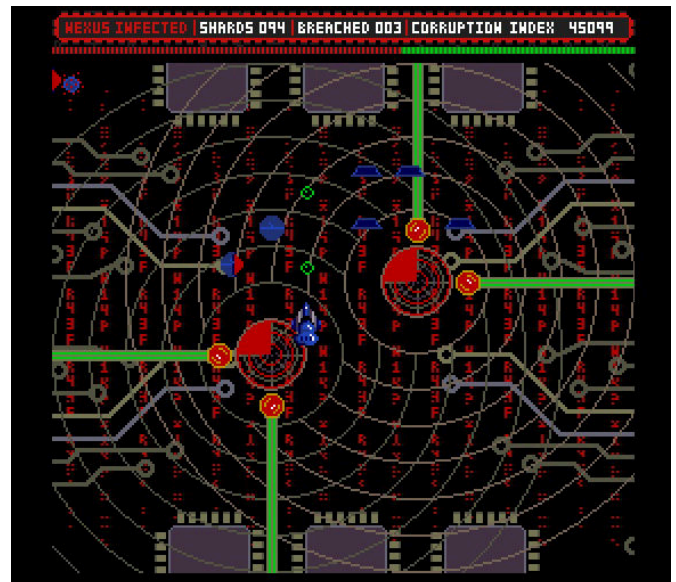
- 6° Blue And Red di Kaiju Wasp
- 7° The Osiris Infection di Hoyster Games
- 8° Microlyte Tankzone di SteveB
- 9° Pauldron di Zener
- 10° Glow Point di Retro Nanny
- 11° Strike Force Omega di Reborn80
- 12° Kaboomania di Axel
- 13° Santos Present Drop 5 di RobSmithDev
- 14° Arpeggio di Tedzogh
- 15° Submarine Attack di Camden Town Games
- 16° Backflip Endless di GameSoulStudio
- 17° Micro Karting di Alimede
- 18° Burn di Ripper253
- 19° Movimiento Constante di DJcaye
- 20° Cogs N Logs di RetroGamma

Li potete reperire da questo link:

- <https://itch.io/jam/amigamejam/results>



Blue And Red di Kaiju Wasp



The Osiris Infection di Hoyster Games



Microlyte Tankzone di SteveB





Il ritorno del gioco di ruolo su Apple II con Andoria

di Roberto Del Mar Pirazzini

Oramai lo avete capito. Amo in modo viscerale qualsiasi forma di gioco di ruolo. Classici, jrpg, action, dungeon crawler (questi li preferisce Nith), dall'alto... insomma li amo e la colpa di tutto ciò è da imputare alla saga di Ultima. Ho sempre adorato questa leggendaria saga realizzata da Richard Garriot e sono felice di poter affermare che questo nuovo progetto di Emmanuel Castello profuma di Britannia/Sosaria. Andoria è un RPG per Apple II che punta dritto al cuore di chi è cresciuto con la saga di Origin, ma con l'obiettivo dichiarato di evitare le lungaggini tipiche dei grandi classici anni '80.

Concept e ambientazione

L'autore presenta Andoria come un gioco "alla Ultima" scritto in 6502 assembly, pensato per girare su un Apple II Plus con language card, quindi perfettamente plausibile per il 1979. Il protagonista è Shamino, leader del villaggio di Thaligar, convocato da Lord Ricard (anche qui abbiamo un'affinità palese con Lord British) nel castello di Andoria per fronteggiare un antico male; la magia tradizionale è quasi scomparsa, sostituita da druidi e pozioni, in una trama volutamente semplice e fortemente nostalgica.

Ritmo di gioco e filosofia di design

Il designer dichiara di voler creare un RPG completabile

in settimane, non in mesi, mantenendo atmosfera e profondità senza scivolare nella ripetitività. Ogni passo sulla mappa rappresenta ore di viaggio, ciò rende fondamentali la pianificazione delle rotte e la gestione di cibo, acqua e riposo, pena la morte per fame, sete o stanchezza.

Esplorazione, sopravvivenza e mezzi di trasporto

Fuori dalle città non esistono alleati: ogni creatura è ostile e ogni escursione nel mondo esterno è un rischio calcolato. Deviazioni inutili e giri a vuoto si pagano con il rapido consumo delle scorte e il rischio di agguati notturni durante l'accampamento. La progressione include cavalli, zattere, imbarcazioni e, nella versione completa, un grande vascello oceanico usato come base mobile con capacità di carico maggiorata, destinato a cambiare radicalmente la scala dell'esplorazione.

Sistema di gioco e gestione dell'equipaggiamento

Andoria adotta un sistema di comandi a tasto singolo in puro stile Apple II: frecce per muoversi e una lettera per ogni azione (attaccare, entrare, parlare, cercare, ecc.). Una scelta caratterizzante è l'uso dei libri d'insegnamento: armi, strumenti e oggetti di combattimento diventano utilizzabili solo dopo aver trovato e letto il manuale



L'intro del gioco e l'invito ad inserire il secondo disco con il mondo di gioco



**Momento d'esplorazione**

corrispondente, spesso nascosto in librerie e scaffali da esaminare con il comando "Search". Un'opzione che ho trovato davvero innovativa!

L'arsenale è sorprendentemente articolato:

- Armi da mischia e da distanza, con balestre e archi per il combattimento tattico.
- Armature progressive, elmi, scudi e guanti per chi desidera un approccio più frontale.
- Gioielli e gemme come bottino ad alto valore e "valuta di lusso".
- Pozioni numerose e codificate per colore, dai classici curativi a effetti avanzati (invisibilità, visione notturna, bonus d'attacco/difesa, evocazioni), molte ancora non abilitate nella demo.

Combattimento e tattica

Il combattimento privilegia velocità e posizionamento. Il protagonista è più rapido di qualunque nemico, favorendo un uso intensivo delle armi a distanza. Il giocatore può alternare al volo armi e consumabili, ricorrendo a pozioni e oggetti offensivi come fireball, trappole, bombe e future abilità avanzate legate, per esempio, al controllo

**Altri incontri non graditi... ma stavolta siamo pronti!****Scontro con un villico andato decisamente male!**

dell'ambiente. È sempre possibile fuggire da uno scontro a costo di oro o oggetti, una scelta che riduce la frustrazione da game over e incentiva la gestione prudente delle risorse.

NPC, dialoghi e mondo di gioco

Città, villaggi e castelli sono popolati da NPC con cui interagire tramite un sistema di dialoghi a keyword, in pieno stile Ultima. Gli abitanti introducono gli argomenti principali e il giocatore può digitare parole chiave per approfondire, rendendo la lettura attenta e la presa di appunti parte integrante dell'esperienza.

Demo, limiti attuali e ambizioni

La demo disponibile rappresenta solo un assaggio: mappa ridotta, assenza di dungeon giocabili, vascello e molte funzionalità avanzate ancora bloccate. Nonostante ciò, emergono chiaramente la struttura prevista del gioco completo, la centralità del viaggio marittimo e il ruolo di dungeon e aree speciali nella progressione.

Il nostro giudizio provvisorio

Anche in forma incompleta, Andoria appare come un progetto coerente e ambizioso, che rilegge l'eredità di Ultima in chiave più moderna nei tempi e nelle richieste al giocatore. La ricchezza di equipaggiamento, mezzi di trasporto, pozioni e strumenti fa intravedere un mondo denso di possibilità emergenti, mentre il vincolo tecnico "authentic Apple II" contribuisce al fascino complessivo. Se lo sviluppo manterrà le promesse su dungeon, viaggio oceanico e missioni, Andoria ha tutto il potenziale per imporsi come nuovo titolo di culto nella scena RPG homebrew su Apple II.

Link per testare la demo:

- <https://andoria.neocities.org/>





CRYSTIAN AND THE LOST CRYSTALS

Anno: 2026

Editore/Sviluppatore: Ntpr

Genere: Platform

Piattaforma: Commodore 64

Sito web: <https://ntpr.itch.io/crystian-and-lost-crystals>

“Crystian and the Lost Crystals” è uno di quei titoli C64 che arrivano senza troppo rumore e, nel giro di pochi giorni, diventano immediatamente “caso” nella scena homebrew. Tecnicamente ambizioso, esteticamente curatissimo, e con un impianto narrativo più emotivo del solito platform labirintico.

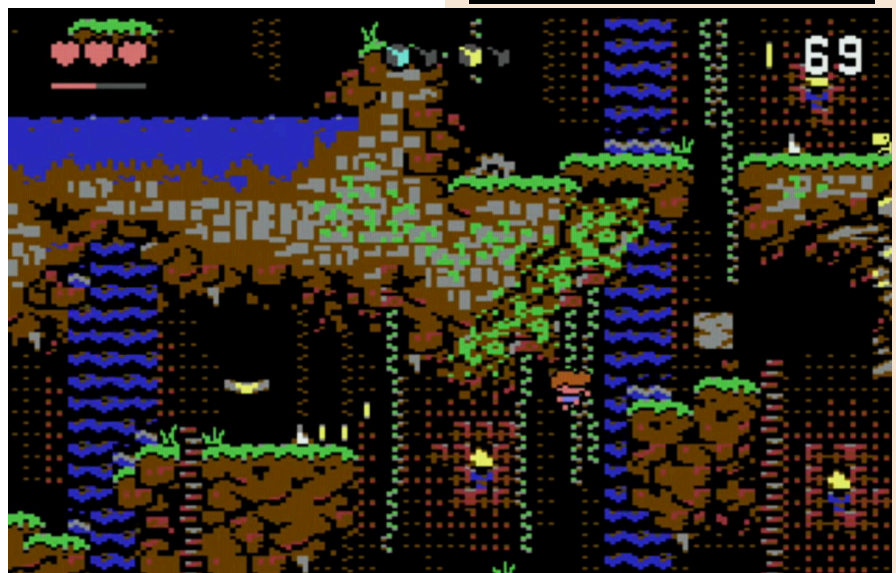
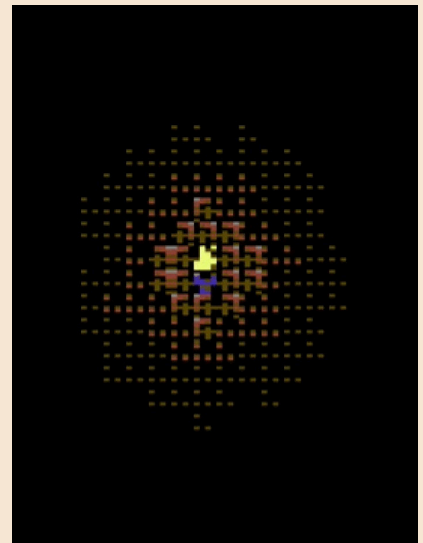
Il gioco, sviluppato da Marcel Sásik (ntpr) che firma codice, grafica, effetti sonori e concept, racconta la storia di Crystian, un ragazzo qualunque che scopre il valore affettivo delle “quasi-crystals” trovate da bambino in miniera insieme allo zio geologo, George. Quelle pietre non valgono nulla per il mondo, ma per lui sono il simbolo del legame con lo zio; quando un’apocalisse di frane, acqua e lava devasta le gallerie, disperdendo la collezione e lasciando lo zio ferito e immobilizzato, il gioco diventa una missione di recupero spinta più dal sentimento che dall’avidità. È una premessa semplice, ma insolita per un platform C64. Al centro non c’è “salva il mondo”, ma il peso della memoria personale e del valore emotivo degli oggetti.

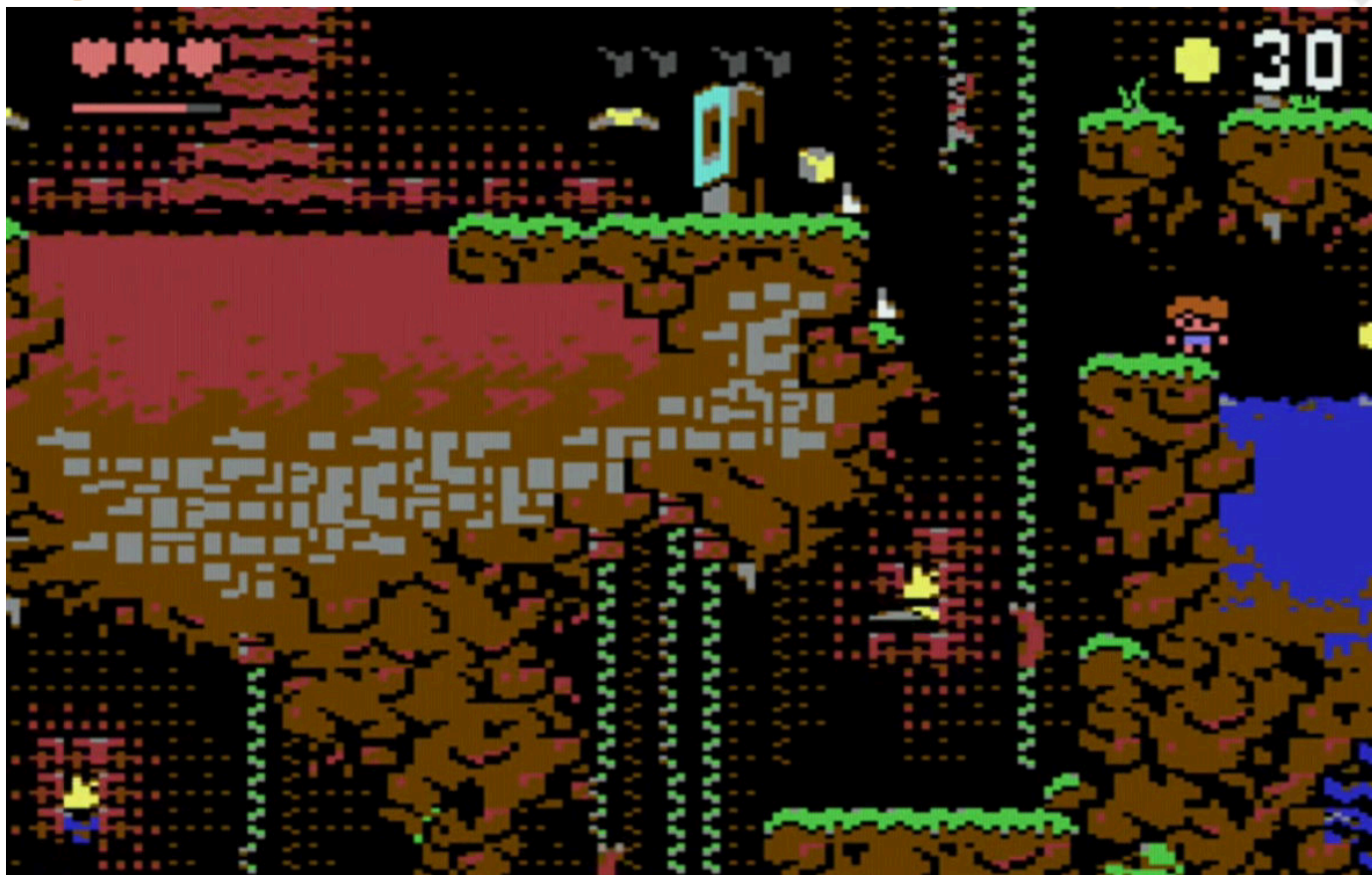
A livello ludico, Crystian è un platform-avventura a esplorazione di labirinti, con tre macro-ambienti graficamente molto ricchi, pieni di trappole, piattaforme mobili e nemici. Ogni location ospita oltre 60 nemici, con comportamenti diversi e danni differenziati, e la progressione è basata su una graduale esplorazione del



“maze” piuttosto che su semplici livelli lineari da sinistra a destra. Il level design punta molto sulla densità. Tanti sprite a schermo, animazioni fluide, fondali dettagliati e una chiara volontà di sfruttare il C64 fino al limite in termini di animazioni e numero di oggetti contemporanei.

Il sistema di gioco mescola piattaforme, raccolta di monete e un leggero strato puzzle (ma leggerissimo proprio). Oltre al classico coin collecting (100



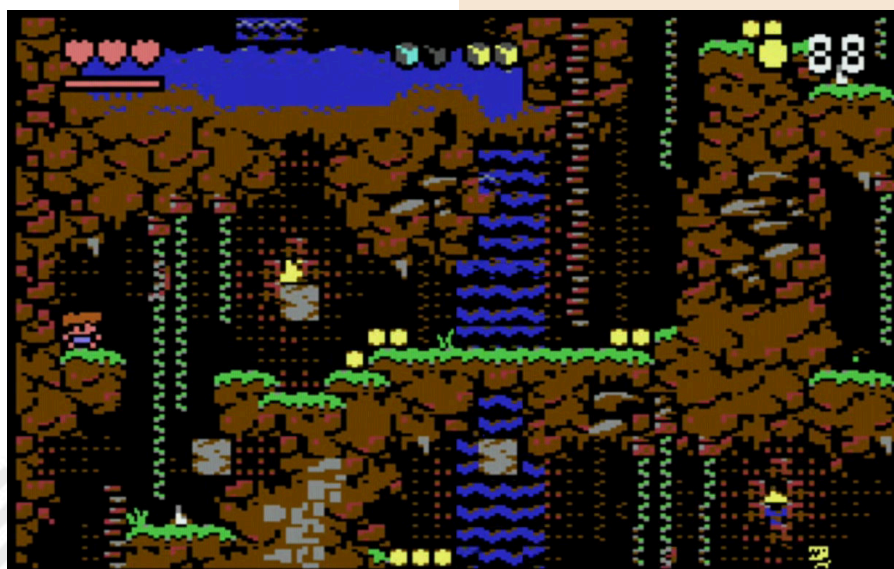
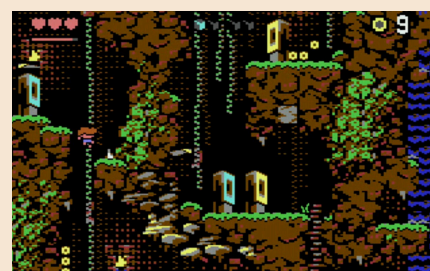


monete = una vita extra) e ai pickup di energia e vite, la progressione nelle aree è legata a un sistema di cubi magici che funzionano da chiavi per aprire porte e cancelli. Crystian può trasportare fino a quattro cubi in totale, due per tasca, e la gestione di quali cubi portarsi dietro e quando usarli aggiunge un minimo di pianificazione agli spostamenti. È un tocco di design che ricorda i puzzle “soft” integrati nel platforming, senza mai trasformare il gioco in un puzzle-game puro.

Il controllo è probabilmente l'aspetto più divisivo e caratterizzante. Il gioco si basa su un engine completamente “a griglia”. Ogni movimento, del giocatore e dei nemici, è allineato alle celle del livello. Questo si riflette direttamente nel salto: se provi a saltare “a metà passo” premendo fuoco nel mezzo di uno spostamento, il salto non parte. Per eseguire un salto corretto devi tenere premuto il fuoco e poi scegliere la direzione. Il balzo viene registrato ed eseguito al termine del movimento corrente, cosa che richiede di pianificare

un minimo il timing anziché improvvisare l'input. Inizialmente si rimane spiazzati e si prova una certa difficoltà, soprattutto su hardware reale, ma una volta compreso il trick (fire tenuto premuto, poi direzione) il gioco diventa molto più gestibile. Lo stesso autore spiega che questa scelta non è un bug, ma una componente intenzionale della difficoltà e dell'identità del motore.

Sul piano audiovisivo, Crystian è uno showreel personale. Animazioni, fluidità e ricchezza di dettagli e scenari sono





un esempio di quanto un singolo sviluppatore possa ancora tirare fuori dall'hardware nel 2026. Gli sprite sono numerosi e ben caratterizzati, con un uso pesante di movimenti e animazioni ambientali. Le palette sono un po' caotiche da distinguere, ma ci si ambienta presto. La colonna sonora, firmata da Hans Svala con contributi dello stesso Sásik, viene descritta come "atmosfera" e "cosy music" ma inserita in un gameplay piuttosto tosto. Un contrasto voluto tra l'estetica quasi contemplativa di alcune sequenze e la durezza delle sfide. La richiesta di un file SID standalone da parte di Exquisite Bytes e la risposta diretta del musicista confermano che le tracce hanno colpito anche chi segue la scena principalmente per l'aspetto audio. Dal punto di vista contenutistico, i tre mondi offrono una campagna compatta ma intensa, con salvataggio "per livello". Quando esaurisci le vite, riparti dall'inizio del livello raggiunto, non dall'inizio del gioco. Una difficoltà "ruvida" e adatta ai giocatori hardcore.

Gira su VICE e su hardware originale, è compatibile sia PAL che NTSC e viene distribuito come immagine .d64 acquistabile su itch.io per 3,99 dollari o più.

In sintesi, "Crystian and the Lost Crystals" è un platform-maze per Commodore 64 che punta forte su atmosfera, densità visiva e un controllo volutamente particolare, innestato su una storia personale inaspettatamente emotiva. È meno immediato di altri titoli arcade-oriented, soprattutto per via del sistema di salto a griglia, ma per chi è disposto a "entrare nel ritmo" offre esattamente quel tipo di esperienza che nel 2026 distingue un semplice homebrew carino da un gioco di riferimento per la scena.

di **Carlo Nithaiah Del Mar Pirazzini**



GIUDIZIO FINALE

» Giocabilità 80%

Ruvido e con alcune meccaniche da apprendere con cura e attenzione. Il level design è ben fatto ed esteticamente è bellissimo.

» Longevità 80%

Il sistema di controllo farà perdere un po' la pazienza, ma una volta appreso si va alla grande!





GHOST'N'GOBLINS AGA

Anno: 2026

Editore/Sviluppatore: JOTD

Genere: Action/Platform

Piattaforma: Amiga AGA

Sito web: <https://jotd666.itch.io/ghostsngoblins>

C'è una generazione che non ha bisogno di trailer in 4K per sentire i brividi: le basta vedere un cavaliere in mutande che corre sotto un cimitero pixelato. Per noi over 40, Ghosts'n Goblins non è solo un arcade: è l'odore di fumo e plastica dei bar, le monetine sudate strette nel pugno, il rumore metallico del gettone che scivola dentro il cabinato, la luce del monitor che buca il buio della sala giochi. È il gioco che non ti perdonava niente, quello che ti umiliava al primo zombie, ma che ti costringeva a riprovare ancora e ancora, perché ogni metro conquistato aveva il sapore di un'impresa epica. Era tornare a casa con il fantasma dei suoi jingle in testa, a disegnare Arthur sui quaderni di scuola, a raccontare agli amici "sono arrivato FINO AL CASTELLO" come se avessi scalato l'Everest. E oggi, quando rivediamo quel cielo viola, quelle tombe, quei demoni con le ali, non stiamo solo guardando un gioco: stiamo riaprendo una porta su un tempo in cui bastava un joystick, due tasti e tanta ostinazione per sentirsi eroi, anche solo per tre vite.

"Ghosts'N'Goblins" di jotd è il porting Amiga AGA che i fan Capcom aspettavano da trent'anni. Tutti e sette i livelli dell'arcade originale, 64 colori su schermo e una fedeltà al gameplay che sfiora la filologia, compresa la storica cattiveria del level design. Arthur corre, salta e lancia le sue armi esattamente come ci si aspetta, con collisioni, pattern nemici e ritmo d'azione che ricalcano da vicino il coin-op, senza ombra di



dubbio una "conversione brillante". A livello di opzioni il port non si limita a riprodurre il gioco, ma lo incornicia con una serie di extra graditissimi. Troviamo la regolazione di difficoltà e numero di vite, il "service mode" in pieno stile cabinato e una batteria di cheat integrati (invincibilità, salto di





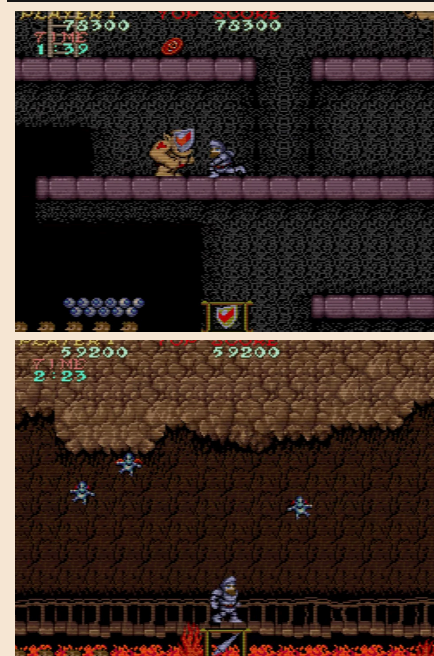
GIUDIZIO FINALE

» Giocabilità 90%

Reattivo, cattivo e rigoroso, nel bene e nel male. Arthur risponde bene ai comandi e il feeling del salto, lancio e collisioni ricalca da vicino il coin-op, ma proprio per questo la curva di difficoltà resta brutale e poco accomodante.

» Longevità 99%

Sette livelli completo, la difficoltà feroce e la presenza di opzioni apposite per questa versione e cheat rendono il gioco qualcosa a cui si torna spesso, sia per provare a migliorarsi, sia per esplorare le varie impostazioni.



livello, cambio arma, warp verticale, incremento del punteggio) che rendono l'esperienza modulabile dal maso-hardcore al turista nostalgico. La distribuzione in formato LHA con installazione WHDLoad lo rende inoltre perfetto per A1200 reali, setup potenziati e anche macchine come THEA500 Mini, come conferma lo stesso autore nei commenti sulla pagina itch.io. Non è tutto oro senza qualche pixel fuori posto. Le prime release non sono esenti da problemi di scrolling (soprattutto in PAL) e compatibilità peculiare con certe configurazioni di pad/CD32, con l'autore che suggerisce esplicitamente di forzare modalità NTSC o 50 Hz e ammette candidamente di essere ancora al lavoro sui compromessi imposti dai 64 pixel di scrolling orizzontale AGA. Parliamo comunque di una versione in evoluzione, aggiornata di frequente (le release 0.92 e 0.93 hanno già migliorato NTSC e corretto bug "cattivi" emersi in WHDLoad), segno di un progetto vivo e seguito.

Nel complesso, questo "Ghosts'n'Goblins" è una delle reinterpretazioni Amiga più convincenti dell'arcade Capcom. Rispettosa, tecnicamente ambiziosa e sostenuta

da una community entusiasta che sta contribuendo a limarne gli spigoli. Se avete sempre sognato di soffrire come in sala giochi ma con l'AGA che pompa grafica e audio, questo port è già adesso un passaggio obbligato, destinato a diventare un punto di riferimento definitivo man mano che lo sviluppo prosegue.

di **Carlo Nithaiah Del Mar Pirazzini**





MAPPY

Anno: 2026

Editore/Sviluppatore: JOTD

Genere: Platform/Puzzle

Piattaforma: Amiga AGA

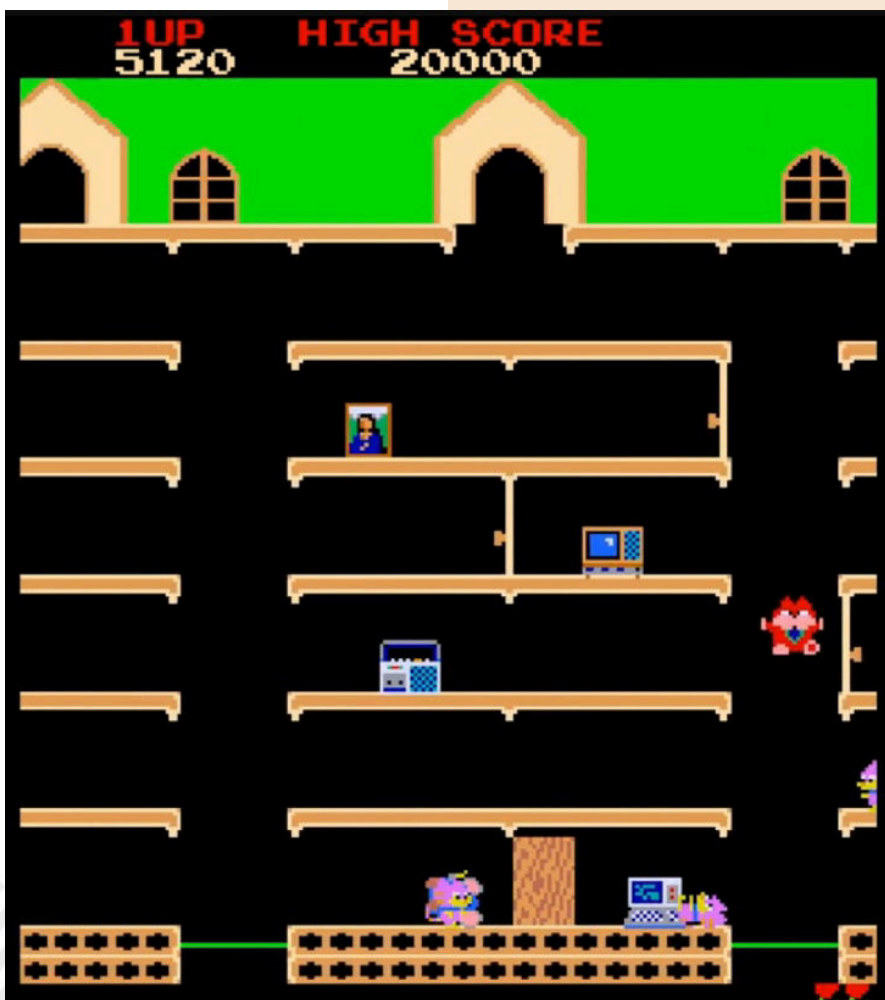
Sito web: [https://](https://jotd666.itch.io/mappy)

jotd666.itch.io/mappy

JOTD ha una missione dichiarata: quella di riportare i grandi classici della sala giochi su Amiga restituendo quel feeling dell'arcade.

Nel coin-op originale, Mappy è un topolino poliziotto incaricato di recuperare una serie di oggetti rubati all'interno di una villa-labirinto a più piani, braccato da una banda di gatti ladri guidati da Goro. Il gioco si svolge su un set di case a sei livelli collegate da tappeti elastici che sostituiscono il salto. Il personaggio non può saltare liberamente, ma rimbalza sui trampolini per spostarsi in verticale, con una meccanica che rende il timing fondamentale e trasforma ogni livello in una partita a scacchi a base di traiettorie e inseguimenti. Le porte rappresentano l'arma tattica principale, aprirle in faccia ai Meowky li stordisce, mentre le porte blu rilasciano onde che ripuliscono temporaneamente corridoi interi, aggiungendo uno strato strategico a un platform che vive di pattern, memoria e gestione degli spazi. Con l'avanzare degli stage compaiono campane da far cadere sui nemici e pavimenti arcobaleno che crollano dopo il passaggio, mentre il gioco accelera con il famigerato "Hurry!" e l'arrivo della palla punitiva Gosenzo Coin, che costringe il giocatore a chiudere rapidamente lo schema. Questo port Amiga AGA di JOTD nasce con una filosofia molto precisa ovvero quella di riportare il coin op in modo

perfetto su Amiga. Con gli stessi colori, lo stesso gameplay e la stessa "sensazione" da sala giochi.. L'autore utilizza reverse engineering per decodificare il codice originale, tradurlo in 68000 e adattarlo alla macchina Commodore, curando anche la conversione grafica e sonora. L'uso della palette AGA permette di replicare molto da vicino l'aspetto vivace del cabinato, che ai tempi girava su hardware Namco derivato da Super Pac-Man. Sprite, fondali e interfaccia riprendono la resa cromatica e la pulizia dell'originale, salvaguardando





quel mix di estetica cartoonesca e tensione che caratterizza il gioco. Siamo sicuri che si troverà il modo di vederlo anche in versione ECS, ma per ora su questa possibilità tutto tace.

Sul fronte della giocabilità, la versione Amiga riproduce integralmente la struttura dei livelli, i comportamenti dei nemici e la progressione degli stage. I controlli restano essenziali con il movimento tramite joystick o tastiera, un solo tasto per aprire le porte, e la consueta simulazione dei crediti arcade (5 per inserire "monete", 1 o 2 per avviare la partita). Il feeling rimane quello di un puzzle-platform travestito da action: senza salto libero, il giocatore è chiamato a gestire i rimbalzi sui trampolini e a prevedere le traiettorie dei gatti, sfruttando porte e onde d'urto per aprirsi corridoi sicuri. La fisica dei trampolini e la loro "usura" se sfruttati troppo spesso vengono mantenute, preservando la tensione tipica delle schermate avanzate. Su itch.io chiunque può scaricare gratuitamente la versione floppy (zip) o l'installazione HD (LHA) e, se lo desidera, lasciare una donazione a sostegno dello sviluppo.

Il gioco risente dello stile e dell'azione del titolo originale, che ricordiamo, era del 1983. Un'era diversa, dove i cabinati erano tarati per farci "perdere monetine"



a causa di una difficoltà non sempre calibrata. Mappy appartiene a quel mondo ed è perfetto per i nostalgici del tempo... meno per chi cerca una sfida più equilibrata.

Nel complesso si configura come un omaggio rigoroso a un classico Namco, più interessato a preservare la struttura e il carattere del gioco del 1983 che a reinterpretarlo. La combinazione di gameplay ricreato con cura, palette fedele e piccole comodità moderne rende questo port un tassello significativo per chi ama vedere l'Amiga misurarsi con i grandi arcade dell'epoca, e una porta d'ingresso ideale per chi vuole scoprire perché un topolino poliziotto assediato da gatti ladri continua a essere ricordato come uno dei puzzle-platform più intelligenti dell'era a 8 bit.

di **Giampaolo Moraschi**

GIUDIZIO FINALE

» Giocabilità 85%

Il coin op c'è tutto ed in tutto il suo splendore!

» Longevità 80%

...e risente dello stile di gioco del tempo e della curva di difficoltà non proprio bilanciata!





AKALABETH 1000

Anno: 2026

Editore/Sviluppatore: Dark

Genere: Dungeon Crawler

Piattaforma: Sega SG1000

Sito web: [https://](https://www.smspower.org/Homebrew/Akalabeth1000-SG)

[www.smspower.org/Homebrew/](https://www.smspower.org/Homebrew/Akalabeth1000-SG)

Akalabeth1000-SG

Nel nostro amato retro-mondo tutto gira attorno al ricordo e alla sperimentazione e questo Akalabeth 1000 sembra quasi uno scherzo.

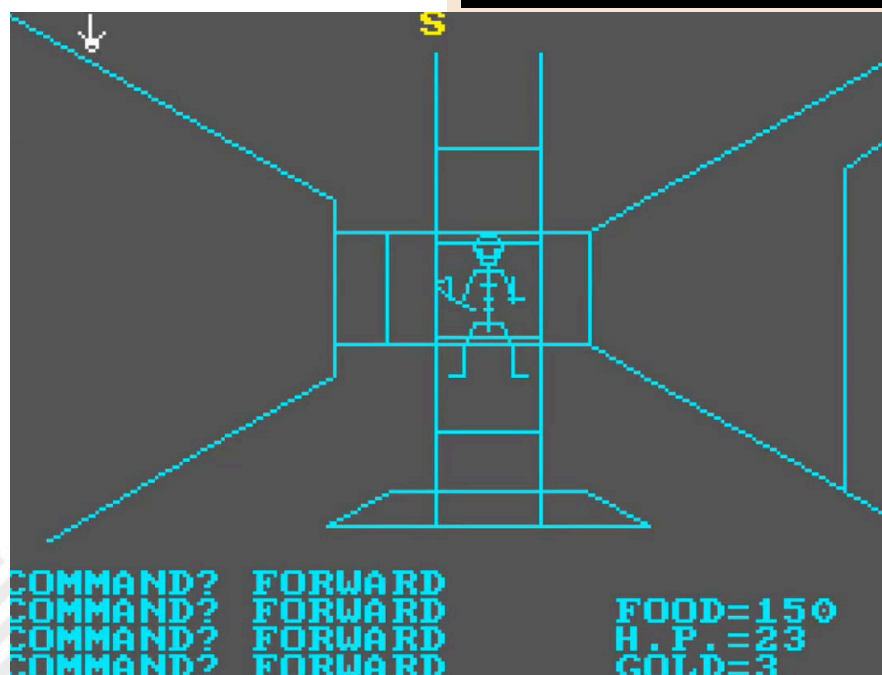
Prendere il prototipo di Ultima, nato come esperimento semi-amatoriale da Richard Garriott nel 1979 e infilarlo nella prima console domestica di Sega, una macchina nata per shooter e platform e non certo per dungeon crawler in prima persona. Il risultato, invece, non fa ridere ma funziona e sorprendentemente bene. Il cuore del progetto è un port della versione GW-BASIC di Akalabeth, ricodificata interamente in C e ottimizzata al punto da girare su un SG-1000 con solo 1 KB di RAM disponibile, con il grosso del lavoro demandato alla ROM che arriva quasi a 48 KB, compressione aggressiva di testi, grafica e audio inclusi. L'autore ha approfittato di questa struttura per aggiungere diverse migliorie rispetto al materiale di partenza, introducendo nuove feature e contenuti che trasformano il gioco in qualcosa di più di una mera curiosità storica travestita da esercizio di stile.

Giocato su SG-1000 o SC-3000, Akalabeth 1000 è ancora chiaramente un dungeon crawler di fine anni '70 dove si alternano mappe in vista dall'alto e dungeon in pseudo-3D wireframe, combattimenti testuali e una progressione ruvida, che conserva l'anima dell'originale, con tutta la sua brutalità e il suo fascino pionieristico. Sul fronte visivo, la resa è pulita ma spartana, con colori più sbiaditi sull'hardware reale rispetto



all'emulazione, e qualche zona meno rifinita, in particolare nel layout dei combattimenti e nell'output testuale, come lo stesso autore ammette parlando di "brutta" gestione di alcuni testi e possibili bug residui. Dal punto di vista storico e concettuale, è un'operazione affascinante per almeno

```
Choose thy lucky number:
42 1980 6174 30000 65535
Level of Play:
01 02 03 04 05 06 07 08 09 10
HIT POINTS...09
STRENGTH...199
DEXTERITY...199
WISDOM...199
GOLD.....07
Shall it thou play with these
qualities? ACCEPT REROLL
```





GIUDIZIO FINALE

» Giocabilità 85%

Brutale ma bellissimo e pionieristico per interfaccia e struttura di gioco.

» Longevità 80%

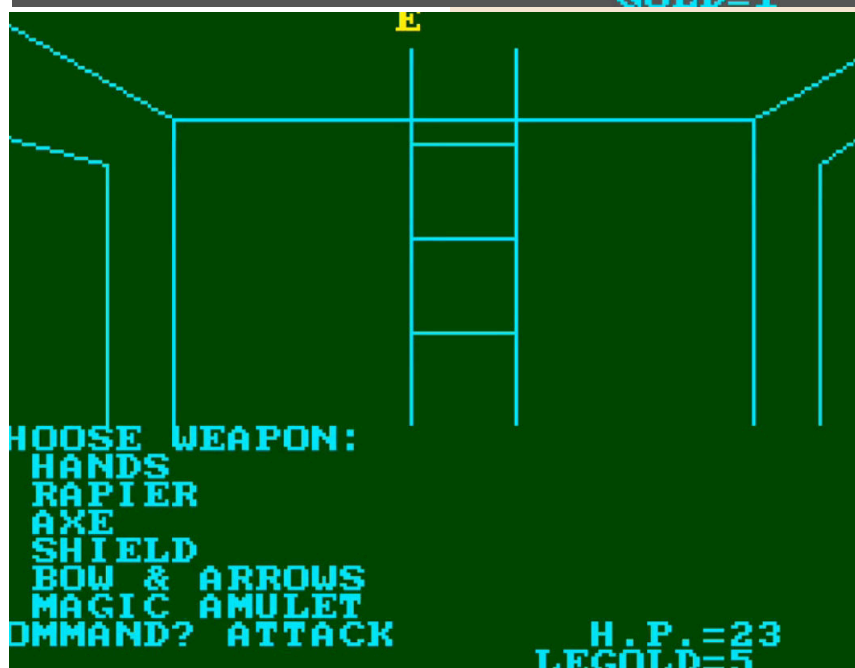
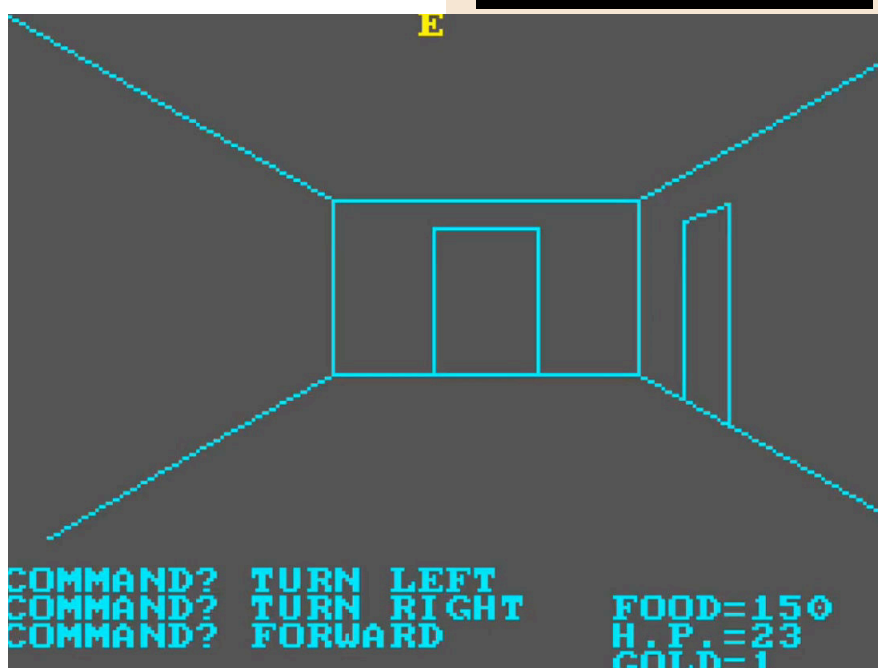
Non lunghissimo ma impegnativo.



due motivi. Da un lato porta uno dei padri fondatori del gioco di ruolo su una console che, all'epoca, non aveva praticamente nulla di paragonabile, anticipando idealmente gli esperimenti RPG che Sega avrebbe visto solo anni dopo; dall'altro dimostra quanto si possa ancora spremere dal modesto hardware dell'SG-1000 grazie a tool moderni e perfino a ottimizzazioni assistite da intelligenza artificiale, usate per compattare e velocizzare il codice C entro limiti quasi assurdi.

Akalabeth 1000 non è un RPG "accessibile" in senso moderno né un remake scintillante ma è un tributo profondamente rispettoso, tecnicamente ambizioso e dichiaratamente work-in-progress, che usa la scena homebrew per far dialogare tre epoche diverse; quella del BASIC dei tardi anni '70, una console dei primi '80 e gli strumenti di sviluppo del 2026. Per chi conosce la storia dei giochi di ruolo, o per chi colleziona stranezze SG-1000, è un pezzo che vale la pena esplorare, anche solo per vedere fino a che punto si può piegare una macchina nata per altro pur di farle raccontare un dungeon vecchio di quasi mezzo secolo.

di **Roberto Del Mar Pirazzini**





WEE NINJA

Anno: 2026**Editore/Sviluppatore:** Playsoft**Genere:** Platform game**Piattaforma:** Atari 5200**Sito web:** [https://](https://forums.atariage.com/topic/382115-wee-ninja-release/)forums.atariage.com/topic/382115-wee-ninja-release/

Wee Ninja è, senza mezzi termini, il platform “alla Mario” che l’Atari 5200 non ha mai avuto all’epoca (in realtà non ha mai avuto tanti giochi a dire il vero), ma che oggi arriva in forma sorprendentemente rifinita e consapevole dei limiti (e punti di forza) dell’hardware. Il gioco offre 32 livelli suddivisi in 8 mondi da 4 stage ciascuno, con progressione lineare e boss di fine mondo che coronano l’avanzata, inclusa una seconda variazione del boss per spezzare la monotonia degli scontri ripetuti. Il level design abbraccia in pieno la grammatica di Super Mario. Abbiamo tutto quello che un platform game richiede: piattaforme, buchi, tubi verdi accessibili che nascondono aree segrete e nemici che si prestano sia al salto “in testa” sia alla gestione strategica delle armi presenti. La curva di difficoltà è stata limata in corso d’opera fino a raggiungere un equilibrio più che gradevole, con un ritmo che evita sia la frustrazione e che risulta subito immediato già dalla prima partita.

Il controllo è quello classico in stile “nintendo 8bit” con lo stick per il movimento, tasto inferiore per il salto (con altezza modulata dalla durata di pressione) e tasto superiore dedicato all’attacco nelle forme potenziate, con una buona fisica nel salto che restituiscono una risposta moderna nonostante il pad 5200.

Il cuore del sistema è la tripla trasformazione del ninja, attivata colpendo i blocchi. Il ninja rosso garantisce invincibilità temporanea,



il nero consente di lanciare “shuriken quadrate” a distanza e il blu permette attacchi ravvicinati con la katana/bastone; in tutti i casi il colpo ricevuto fa retrocedere allo stato base anziché togliere subito una vita. La versione finale aggiunge una versione “easy”





con hit point extra rappresentati da un cuore accanto al counter vite, distribuisce cuori bonus nascosti ottenibili solo soddisfacendo condizioni particolari e introduce continue limitati (cinque) più world code dopo i boss, così da permettere un apprendimento meno “violento” rispetto ad una run classica.

Sotto il cofano, Wee Ninja sfrutta una ROM da 64K con schema di bank-switching M.U.L.E. di Bryan, soluzione storica per portare software di grandi dimensioni su 5200, e gira senza problemi su hardware reale tramite Atarimax Ultimate SD o BackBit Pro oltre che via Altirra (preferibilmente con il file .car per l'autodetect). L'uso del bank-switching permette di ospitare una quantità di grafica, logica di gioco e musica che va ben oltre lo standard dei titoli commerciali originali del sistema (anche qui non ci stupiamo, visto la scarsità di titoli e la loro qualità verso il basso), tanto che diversi utenti sottolineano come un prodotto del genere avrebbe fatto la differenza se fosse arrivato nei primi anni '80. Dal punto di vista audiovisivo, i fondali e i tiles ricordano immediatamente l'estetica Mario-like, anche troppo, ma sono adattati con gusto alle palette e alle capacità del 5200, mentre la colonna

sonora è una rielaborazione dei brani di Super Boy 3 su MSX, altro clone mariesco, con qualche glitch sonoro che l'autore cercherà di sistemare con una futura release. Pur dovendo tutto al modello Super Mario, Wee Ninja non è un semplice hack grafico. La gestione delle tre forme del ninja, il tuning della difficoltà, la presenza di codici mondo, extra life nascosti e varianti di boss restituiscono la sensazione di un platform completo, pensato per il 5200. Allo stesso tempo, la somiglianza estetica con l'originale Nintendo è così marcata che l'autore preferisce una distribuzione solo in forma di ROM per evitare possibili attriti legali su un'eventuale edizione fisica, lasciando volutamente spazio ai singoli per farsi la propria cartuccia artigianale. Nel panorama homebrew Atari, Wee Ninja si ritaglia quindi il ruolo di showcase tecnico e di manifesto, dimostrando quanto si possa ancora spremere dalla bistrattata console Atari in termini di platforming “in stile console nipponica” e, allo stesso tempo, quanto sottile sia il confine fra omaggio e clone quando si gioca con icone del passato. Ottimo prodotto!

di **Carlo Nithaiah Del Mar Pirazzini**

GIUDIZIO FINALE

» Giocabilità 90%

Ottima reattività nei comandi e una bella varietà nello stile di gioco. Un clone di Mario, ma con una bella varietà in più.

» Longevità 90%

I 32 livelli di gioco sono tanti e le sfide sono godibili. Vi terrà incollati al pad.





SPACE BASH: CONTACT!!

Anno: 2026

Editore/Sviluppatore: Dave's Retro Forge, The Geps, Hassink

Genere: Puzzle/Platform

Piattaforma: MSX, 16 KB di RAM

Sito web: <https://daves-retro-forge.itch.io/space-bash-contact-msx>

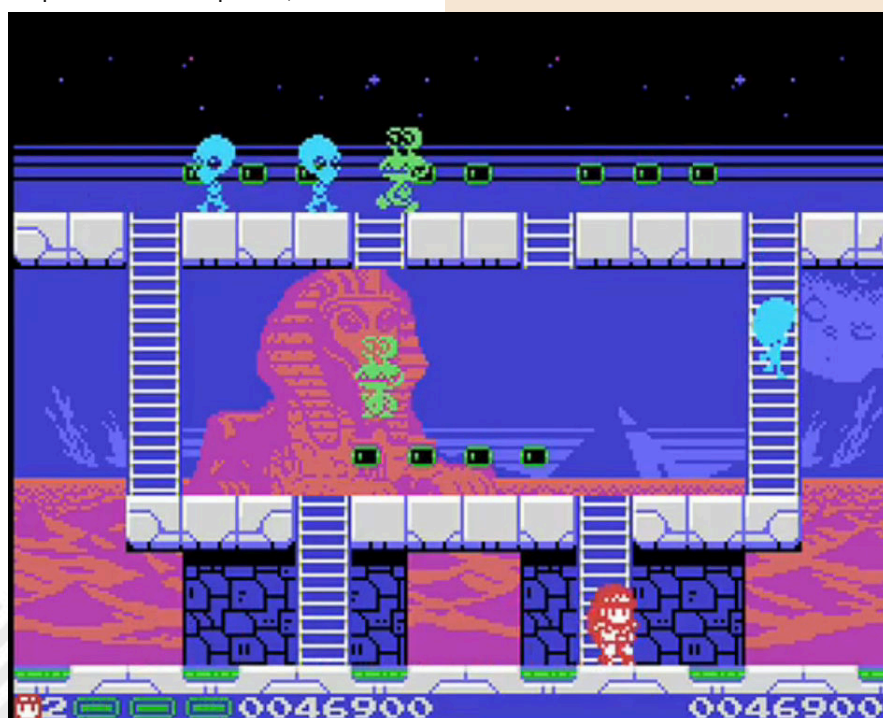
Mi chiamo Ingrid Poggiali e lo confesso: sono salita su Space Bash: Contact!! pensando a un giro turistico nello spazio, sono finita in un trasloco intergalattico armata di “raggio laser”, Kamehameha e scala portatile.

Sulla carta sembra un platform schermo fisso in stile Rodland, ma in pratica è una piccola lezione di game design e di amore per il titolo storico di Jaleco.

La missione è subito chiarissima! Ripulire venti livelli, ambienti interni ed esterni, da tutti i peggiori tipi di alieni spaziali dalle forme più bislacche e mai concepite. Il tutto in un mix puzzle-azione di piattaforme, pareti, spazi stretti e quella gestione chirurgica dei salti che ci fa capire davvero quanto amore maniacale ci sia dietro a questo gioco. Il fatto che tutto questa “BELLEZZA” giri su qualsiasi MSX con almeno 16 KB di RAM rende ancora più gustoso vedere quanta varietà si riesca a spremere da risorse così limitate. Pad alla mano – o tastiera, che tanto noi siamo abituate – Space Bash: Contact!! è una piccola palestra di controllo. Muovi l'eroina, costruisci scale pieghevoli per arrivare dove la semplice fisica non basta, ti lanci in planata per coprire distanze esagerate e nel frattempo cerchi di non farti abbracciare con troppo entusiasmo dagli alieni. Il colpo di grazia è il sistema d'attacco! Stordire un nemico e “sbatterlo” addosso agli altri,



ottenendo una catena di stordimenti che trasforma metà dello schermo in un bowling cosmico perfettamente soddisfacente, deliziosamente stupefacente! A questo, il team di





sviluppo ha aggiunto il colpo d'energia secondario, una palla energetica caricabile che ci libera in caso di situazioni complesse anche per un ninja spaziale. Usarlo ripulisce e fa stare bene fin da subito, ma è una risorsa preziosa, non un tasto "No Panic" infinito, e costringe a pianificare.

Sul piano tecnico si gioca pesante! Una Roma formato Konami SCC da circa 544 KB, musica ed effetti sonori affidati al chip SCC e una grafica che strizza l'occhio a una palette e a scelte cromatiche più "avanzate" pur restando perfettamente credibile sull'hardware base. Il lavoro sulle animazioni è superlativo. Amo gli alieni con gli occhi sopra la testa che sputano per farci fuori oppure i piccoli cubi "petomani" che svolazzano sullo schermo. Le musiche firmate da John Hassink pompano l'azione in modo degno di una sala giochi in miniatura. C'è tanto arcade storico che viene voglia di alzare il volume a manetta!

Dietro alle quinte c'è un team affiatato: concept, game design e grafica di Davide Bottino (il papà di Toki Remastered, Bubble Bobble Remastered e Bubble Bobble The Lost cave su C64, ndN), il

già citato Hassink per la musica e TheGeps (Giuseppe Ettore Pintus) al codice e alla programmazione. Questi tre bravissimi artigiani hanno confezionato il tutto per l'MSXdev 2025 con una distribuzione gratuita in linea con il regolamento del contest. Da brava "ingegnera" mancata (ovvero mi mancano ancora una marea di esami), non posso non apprezzare come Space Bash: Contact!! riesca a stratificare l'esperienza. Puoi giocarlo "alla leggera", limitandoti a sopravvivere e raccogliere batterie oppure studiare layout, sfruttare al millimetro scale e planata, orchestrare combo di stordimenti e spremere il sistema di punteggio per extra vite e prestigio personale. Posso solo ammettere che, se questo è il modo in cui nel 2026 si continuano a scrivere nuovi capitoli per l'MSX1, allora mi sa che io e il vecchio home computer di mio padre avremo ancora parecchie serate galattiche da passare insieme.

di **Ingrid Poggiali**

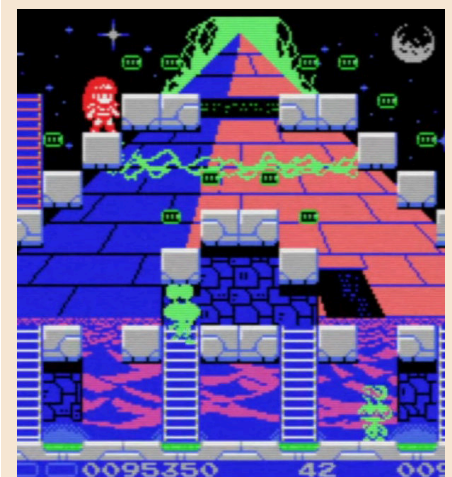
GIUDIZIO FINALE

» Giocabilità 95%

Hit box meraviglioso, aspetto tecnico incredibile, level design maniacale e cura estrema in tutti i settori. E' il titolo MSX di questo 2026.

» Longevità 95%

Una partita tira l'altra.





CELESTE

Anno: 2026

Editore/Sviluppatore: Pawel Juen Nowak

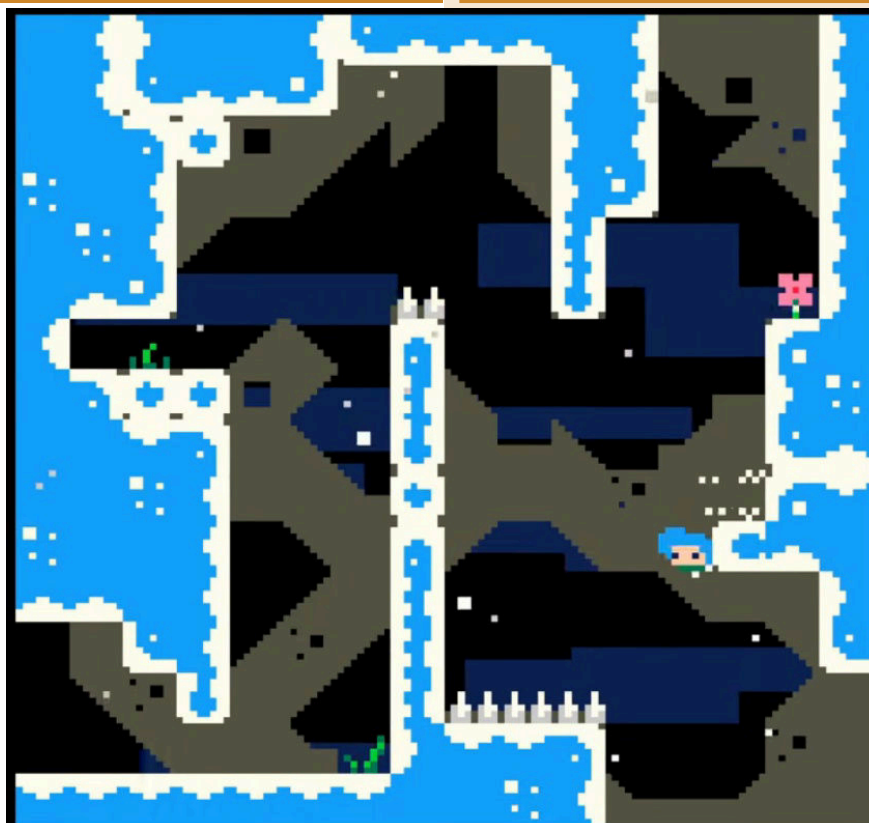
Genere: Action/Platform

Piattaforma: Amiga

Sito web: <https://twoj.szajb.us:442/index.php/s/250oeKmECbyw82g?opendetails=>

Il platform cult Celeste continua a invadere il retrogaming senza pietà! Celeste è un platform indie sviluppato da Maddy Thorson e Noel Berry (Maddy Makes Games / Extremely OK Games), nato come prototipo Pico-8 creato in quattro giorni durante una game jam. La versione completa è uscita il 25 gennaio 2018 su PC, console e Linux/macOS, e ci mette nei panni di Madeline, una ragazza che decide di scalare la montagna Celeste per affrontare ansia e depressione, con un gameplay basato su salti pixel-perfect, arrampicate a tempo e uno scatto a mezz'aria ricaricabile. Dietro ai riflessi da speedrunner c'è una narrativa molto personale, una colonna sonora di Lena Raine e un level design hard ma mai punitivo, con respawn istantaneo e morte usata come metrica del miglioramento. Il port Amiga di Paweł Juen Nowak parte proprio dalla versione Pico-8 open-source, cioè dal "Celeste Classic" con una trentina di livelli pensati per movimenti rapidi e precisi. Da lì, tutto viene adattato al mondo planar dell'Amiga usando C2P (Chunky to Planar), tecnica che converte il framebuffer lineare tipico delle macchine moderne/virtuali in piani grafici compatibili con OCS/ECS/AGA, con un costo in CPU tutt'altro che banale.

Dal punto di vista delle meccaniche, il port Amiga riprende il cuore del Celeste originale: movimento basato





su salto, arrampicata a tempo limitato e uno scatto a mezz'aria in qualsiasi direzione; level design compatto, pieno di ostacoli letali ma con respawn immediato; collezionabili tipo fragole in posizioni bastarde che spingono a padroneggiare il movimento.

Ovviamente non avremo l'intero pacchetto della release 2018 (tutti i capitoli, varianti B-side/C-side, DLC gratuito Farewell del 2019), ma la fetta portata su Amiga è quella più "pura": la scalata tecnica e di abilità, senza troppi fronzoli. La differenza grossa sta nel rendering. L'Amiga ha audio e grafica più ricchi rispetto a NES e Pico-8, ma la sua architettura planare rende più complessa la gestione di sprite e layer pensati per un framebuffer lineare, quindi ogni frame fluido è un mezzo miracolo di ottimizzazione. Il gioco è pensato per Amiga "muscolosi" (A1200/A4000 con acceleratori vari) o emulatori ben configurati, con la community – in particolare il forum PPA.pl – che fornisce config e feedback per spremere ogni ciclo di CPU. È un port che non punta all'accessibilità universale della versione

moderna (che offre perfino modalità Assistita per modulare la difficoltà), ma è chiaramente indirizzato a chi conosce l'hardware, accetta i compromessi e vuole vedere fin dove può spingersi la cara vecchia Amiga con un design del 2018.

di **Giampaolo Moraschi**

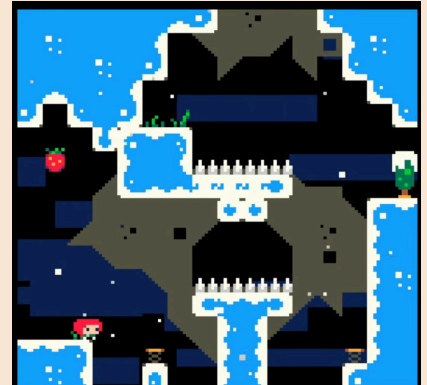
GIUDIZIO FINALE

» Giocabilità 85%

Celeste è divertente e godibile. Il level design è ben definito e si gioca che è un piacere.

» Longevità 80%

Il titolo è sicuramente duraturo ma soffre molto ed è adatto solo a configurazioni "alte".





RANDOOM

Anno: 2026

Editore/Sviluppatore: Epy,
Csabo, Unreal, SukkoPera

Genere: Puzzle/Platform

Piattaforma: Plus/4

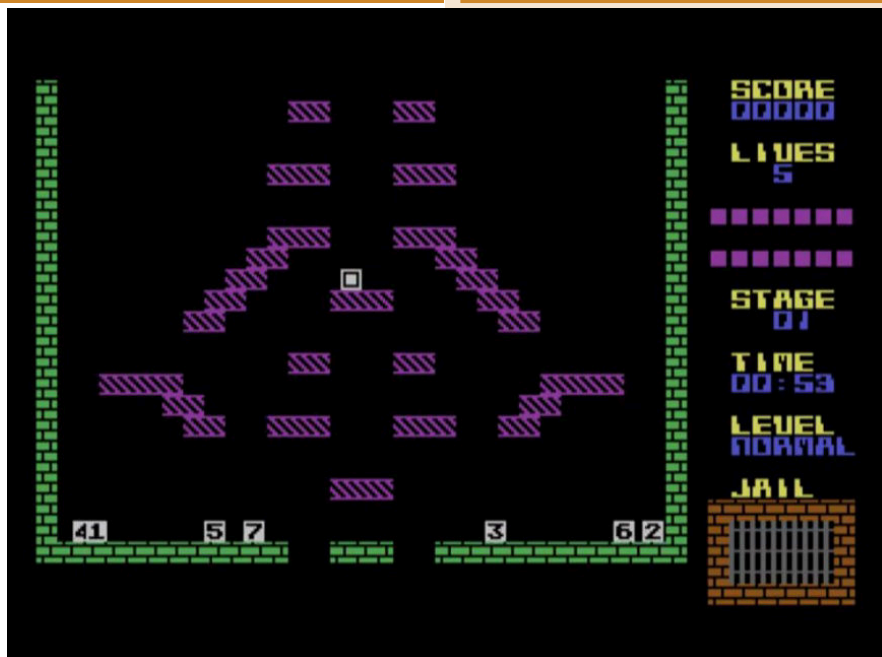
Sito web: <https://plus4world.powweb.com/software/Randoom>

Randoom ti frega! Un gioco sulla carta “minuscolo” ma che in solo due minuti netti di gioco ti fa entrare in un loop fatto di “ancora una partita e giuro che smetto”.

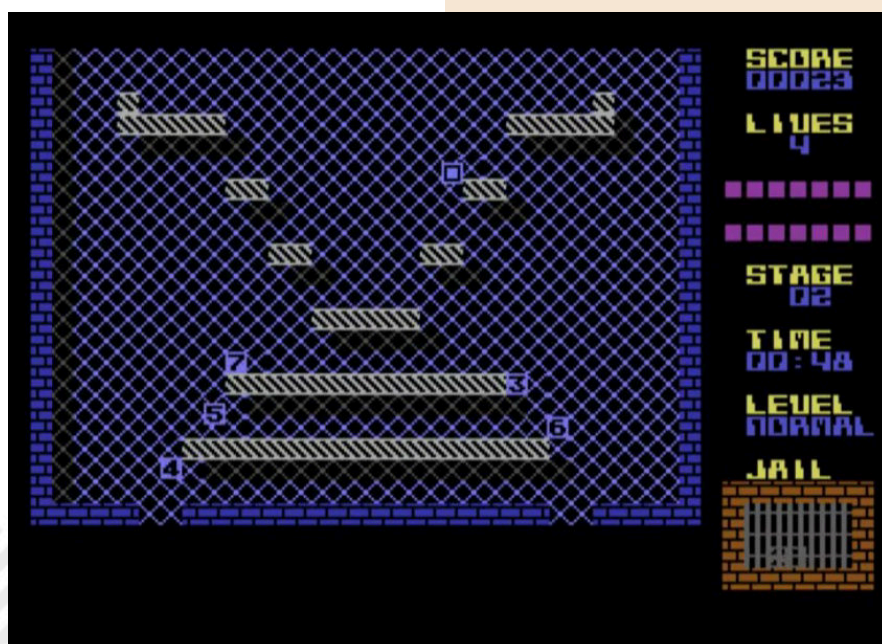
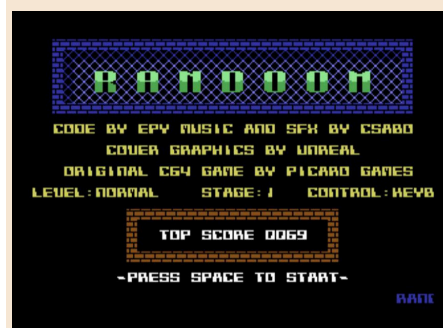
Platform nato con un’idea minimalista, che ci mette nei panni di un “omino” che ha il compito di raccogliere quadrati nel giusto ordine per completare il livello e passare a quello successivo. Semplice da capire, ma tutt’altro che banale da padroneggiare quando il timer corre e la disposizione cambia di livello in livello.

La struttura è basata su 31 stage selezionabili a cui si affianca fin da subito un sistema di difficoltà insolitamente generoso che permette di calibrare il livello partendo da cinque modi selezionabili: too easy, easy, normal, hard e insane. Ognuno di questi modifica il ritmo, la severità del timer e la precisione, permettendo al giocatore di adattarsi. Perfetto quindi per il neofita quanto per il master gamer alla ricerca del “Punteggio Perfetto”.

Sul fronte delle meccaniche, oltre al movimento standard spicca il salto al volo a mezz’aria, un piccolo balzo che non stravolge la fisica ma ci salva la pelle, aggiungendo quel margine di controllo in più necessario per correggere traiettorie all’ultimo istante. Centrale è la gestione del tempo: ogni secondo risparmiato si



traduce in Tempo Bonus, incentivando il gioco aggressivo, quasi al limite della speedrun, piuttosto che un approccio prudente ma inefficiente. Tecnica e qualità del prodotto sono decisamente sopra la media del nostro amato Plus/4. Sfrutta al meglio i 64K





GIUDIZIO FINALE

» Giocabilità 92%

Bellissimo da giocare, con una marea di opzioni per realizzare la sessione “ideale” e il supporto inaspettato ma piacevole per il pad dello SNES.

» Longevità 90%

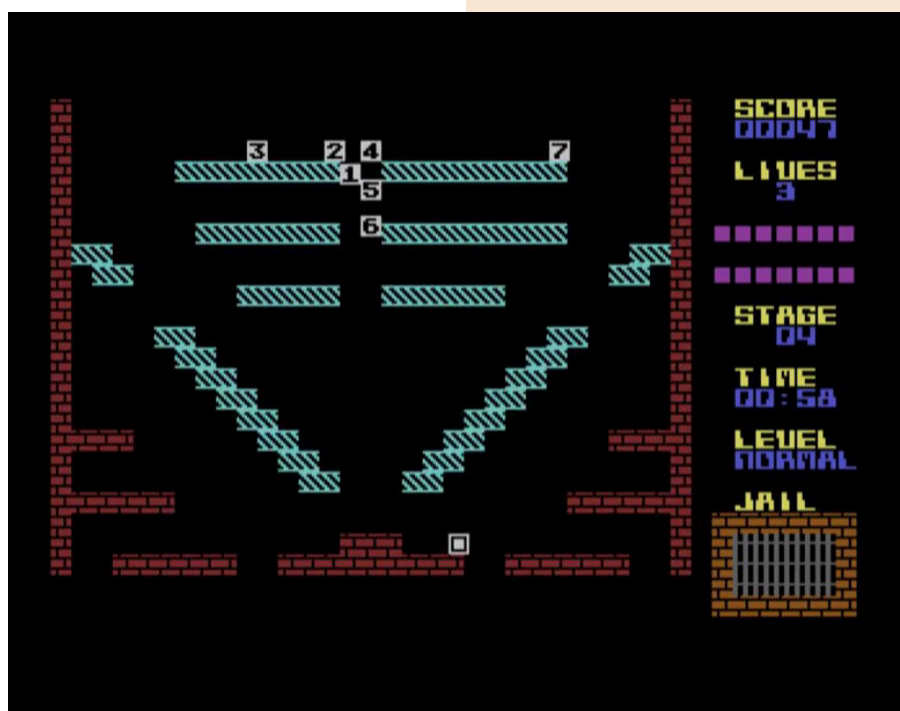
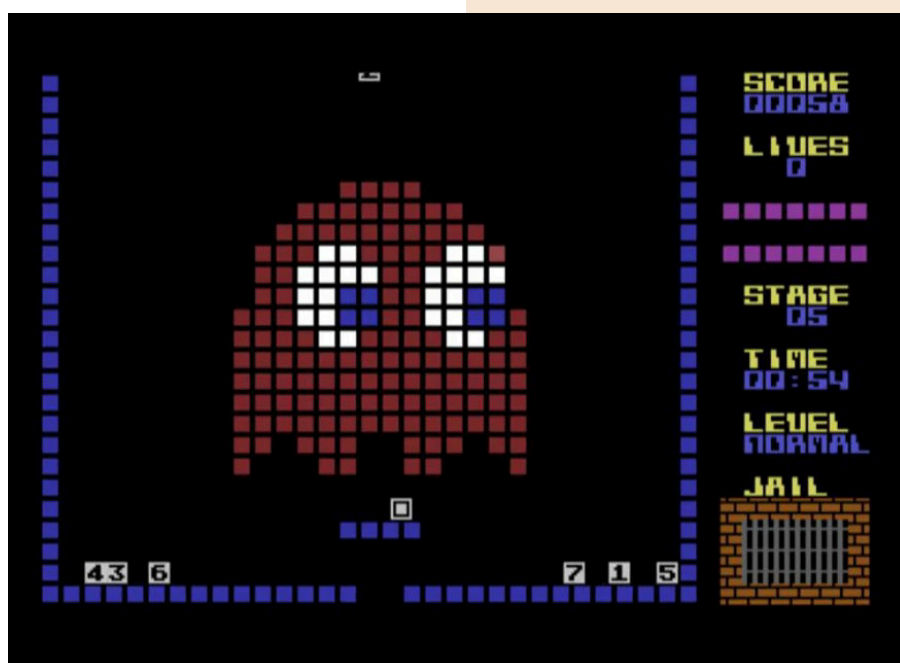
Il sistema di difficoltà aiuta ad apprendere le meccaniche a qualsiasi tipo di giocatore. E' senza ombra di dubbio il titolo da “una partita tira l'altra”.

della macchina, con una grande programmazione e con una serie di opzioni degne dei moderni indie. Interessante il supporto per una marea di controlli, dalla tastiera al pad per il Super Nes. Quest'ultima “chicca” rende il gioco super-comodo anche in sessioni molto lunghe, dettaglio non scontato sull'hardware originale.

Il team dietro a tutto questo è rodato e ben noto alla scena Plus/4. Al codice troviamo Epy, gradica di Unreal, colonna sonora di Csabo e ulteriore codice a supporto da parte di SukkoPera, il tutto confezionato come freeware in modalità solo-PAL.

Randoom è un platform d'azione ultrapulito nelle sue regole, che fa della precisione dei controlli, della scalabilità della difficoltà e dell'ossessione per il punteggio il suo vero “gancio”, dimostrando ancora una volta quanto il Plus/4, nelle mani giuste, sappia ospitare design dal taglio sorprendentemente contemporaneo. Dopo averlo apprezzato su altre piattaforme (MSX e C64) è un piacere vederlo in gran spolvero anche qui.

di **Carlo Nithaiah Del Mar Pirazzini**





CHRONO RUNNER

Anno: 2026

Editore/Sviluppatore:

Associazione MSX Italia

Genere: Platform/Puzzle

Piattaforma: MSX

Sito web: [https://](https://www.msxdev.org/2026/03/14/msxdev25-38-chrono-runner/)

[www.msxdev.org/2026/03/14/](https://www.msxdev.org/2026/03/14/msxdev25-38-chrono-runner/)

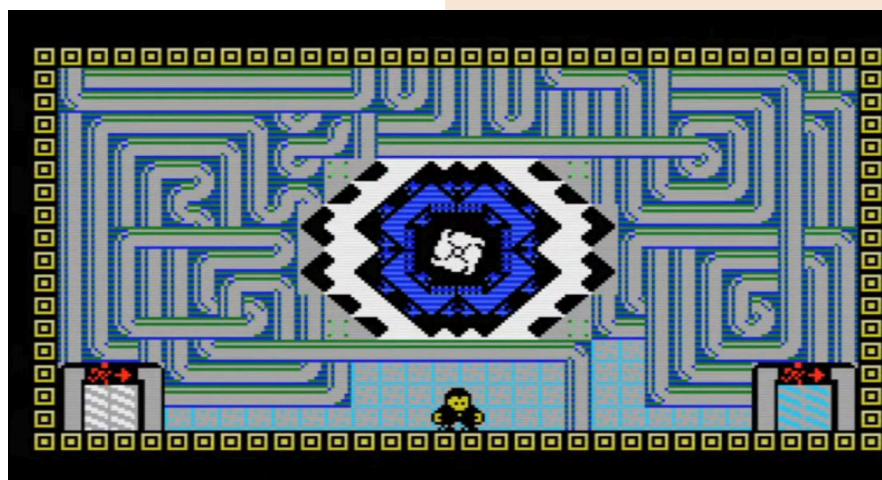
[msxdev25-38-chrono-runner/](https://www.msxdev.org/2026/03/14/msxdev25-38-chrono-runner/)

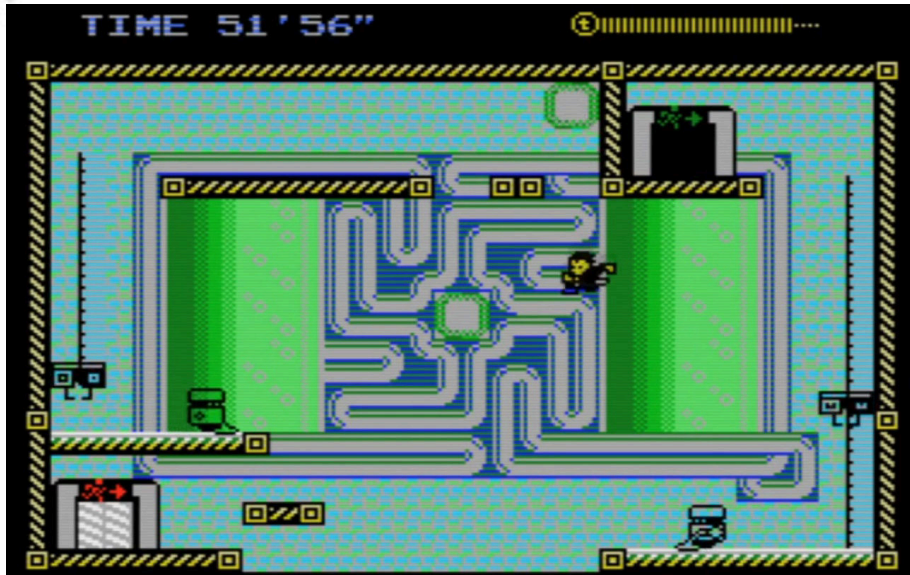
Solo 60 minuti (reali) per trovare quattro Singolarità di Tachioni in un mega-labirinto pieno di stanze situato all'interno di un centro di ricerca ad altissima sicurezza a Berlino. Una corsa contro il tempo e contro droni di sorveglianza, trappole e labirinti claustrofobici da affrontare a suon di velocità, prontezza di riflessi e pazienza... tanta pazienza!

Un gioco dove ogni secondo in più è un'arma nel proprio arsenale... solo a quei matti dell'Associazione MSX Italia poteva venire! Ed è venuto proprio bene!!!

Un platform/puzzle con una struttura labirintica e tante stanze da esplorare ma con lo spirito e l'adrenalina classica dei giochi adatti agli speedrunners. Sotto il cofano di Chrono Runner troviamo l'MSXgl di Aoineko, un engine moderno C/ASM ben collaudato, troviamo una grafica intrigante e veloce e un solidissimo level design. Tutto questo pensato per girare su un MSX senza espansioni folli, SCC... solo quello che avevamo in camera da letto nei gloriosi anni '80.

Il gioco presenta controlli reattivi che richiedono riflessi fulminei. I salti calibrati, i tempi di reazioni contro gli avversari e in genere lo studio del livello richiedono attenzione fulminea... il ragionamento "molle"





GIUDIZIO FINALE

» Giocabilità 95%

Controlli precisi e fulminei che richiedo pratica ma danno tantissima soddisfazione. Il level design è pazzesco!

» Longevità 95%

E' un gioco difficile che però ti incolla al joystick. Attualmente è il mio retro-titolo più giocato da inizio anno.



Runner. L'MSXDev quest'anno è senza dubbio l'edizione migliore e con un alto tasso di competitività! Bravi!

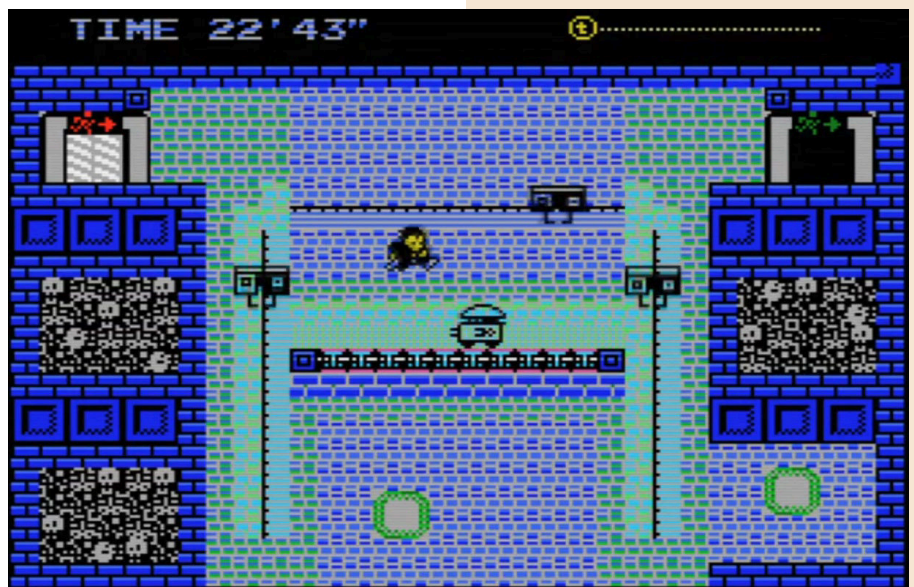
di Giampaolo Moraschi

non è consentito. Ogni stanza ci parla e ci dice chiaramente in che ordine recuperare l'accesso, evitare i droni, raggiunge l'uscita senza buttare via secondi preziosi... basta solo leggerla con attenzione in pochi secondi.

Attenzione, questo gioco non è per tutti ma solo per gli amanti delle sfide e per chi è dotato di quello spirito da hardcore gamer... tutti gli altri vadano su altri lidi e altri stili. Questo è un gioco duro e richiede un incredibile sacrificio per essere portato a termine.



Non ci resta che rimandarvi ai giudizi finali e ringraziare i ragazzacci dell'Associazione MSX Italia per Chrono





BLACK CASTLE II

Anno: 2026

Editore/Sviluppatore: User0x7f

Genere: Action/Platform

Piattaforma: Game Boy

Sito web: <https://user0x7f.itch.io/black-castle-2>

Questo gioco mi fa venir voglia di tirar fuori il mio Game Boy, prepararmi il tè, e annullare qualsiasi tipo di impegno con questa scusa “perdonatemi, il mago malvagio Malakar ha la precedenza... sapete come sono questi cattivoni dei videogiochi”.

Seguito diretto di Black Castle, sempre firmato User0x7f, sempre action-platform per Game Boy, ma con quella bella aria da “stesso vestito, taglio migliore”. Insomma, più livelli, più armi, più segreti, più di tutto.

La trama fa subito soap opera medievale come piace a me. Anni fa il cavaliere Aric aveva già preso a schiaffi il sinistro stregone Malakar nel Black Castle. Tutto bene? Macché. Il fetido sopravvive, si lecca le ferite nell'ombra e torna più carico che mai, deciso a buttare il mondo in un'eclissi eterna usando un rituale in cui, dettaglio di poco conto, ha scelto Lyra – sorella di Aric – come sacrificio. E quindi lei, da brava sorella ansiosa davanti allo schermo, si ritrova a pensare “ok, Aric, rimetti l'armatura, prendi la spada e vedi di sistemare il pasticcio di famiglia un'altra volta”. Rispetto al primo episodio qui si va decisamente di upgrade strutturale. Via l'avventura “snack” da quattro stage ma ben cinque livelli belli pieni e belli gonfi come piacciono a me, ognuno con il suo boss finale, i suoi passaggi nascosti e i suoi oggetti da scovare. Il gioco resta rigorosamente “old school” nei controlli – A per saltare, B per attaccare, croce per muoversi, Start per iniziare/pausa,





Select decorativo – ma il feeling è più rotondo. Il level design “pulito” e molto più intuitivo, con piattaforme e nemici piazzati dove il tuo istinto di giocatrice direbbe “qui ha senso metterli”, non dove un designer sadico ha deciso di farti soffrire.

E quindi ci si ritrova a combattere attraverso i corridoi del castello, stanza dopo stanza, con un ritmo che ti tiene sempre un filo tesa ma raramente ingiusta. Il bello è che lungo la strada non trovi solo cuori e punti, ma anche nuove armi, passaggi segreti e oggetti bonus che cambiano un po' il modo in cui affronti le schermate. Uno dei giocatori cita un'area segreta nel secondo livello che sembra un omaggio diretto a Super Mario Bros, e l'autore conferma che sì, le ispirazioni sono proprio Mario e Castlevania, con addirittura mattoni rompibili sparsi in giro. In pratica, se ami saltare contro ogni pezzo di muro sospetto, questo gioco premia di continuo. Sul fronte estetico, è la versione “ho messo il rossetto e il vestito buono” del primo capitolo. Grafica super pulita e uno scroll morbido con look che sembra

uscito direttamente da una line-up commerciale Game Boy di fine anni '90, nel senso buono del termine. Ogni schermata è leggibile, i nemici si distinguono bene dallo sfondo, e il castello nero è un posto in cui ci si sente a casa dopo pochi minuti. Orribili e tetri corridoi, cappelle poco sante, segrete piene di roba che vuole strapparle l'ultimo cuore rimasto. Musicalmente siamo sulla stessa lunghezza d'onda e non a caso la colonna sonora è firmata da Retro Asobi, che confeziona chiptune da scuotere la testolina mentre si schivano proiettili.

La cosa divertente è che, pur essendo un sequel più grosso, non diventa una di quelle maratone infinite che ti fanno invecchiare davanti allo schermo. La curva della difficoltà è ben bilanciata e punisce lo sbaglio lasciando sempre la sensazione che, con un'altra run, si può arrivare più avanti. Insomma, una bella perla per GB e una saga che speriamo vivamente continui... magari con un bel Black Castle 3!

di **Ingrid Poggiali**

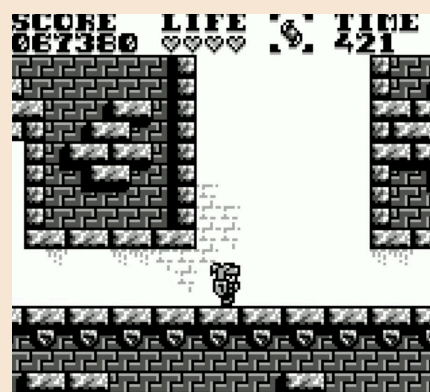
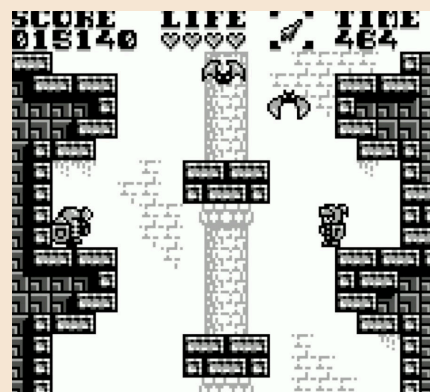
GIUDIZIO FINALE

» Giocabilità 90%

Solido, classico, pulito e ben strutturato. Bello, bello, bello!

» Longevità 82%

Difficile e punitivo, ma con quel buon sapore di ... ci riprovo!





ECHODROME

Anno: 2026**Editore/Sviluppatore:** PixelPlop**Genere:** Shoot'em up**Piattaforma:** Amiga**Sito web:** [https://](https://pixelplop.itch.io/echodrome-amigajam2025)pixelplop.itch.io/echodrome-amigajam2025

Echodrome è nato con un obiettivo chiarissimo: riportare l'Amiga nelle sale giochi, senza filtri nostalgici né scuse tecniche. Sviluppato interamente in C++ in circa sei mesi per l'AmiGameJam 2025 – il cui tema era “Return of the Arcade” e che ha vinto! – è un concentrato di design essenziale e ossessione per le prestazioni su hardware reale, non un semplice “gioco da emulatore”.

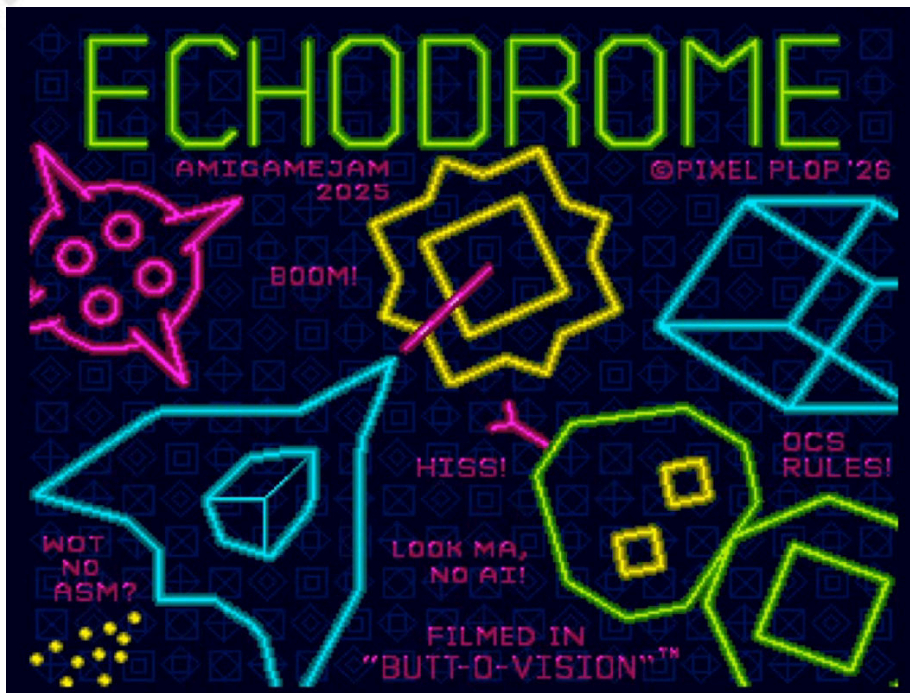
E sapete cosa fa? Gira su un hardware basico in modo pazzesco. Richiede infatti un Amiga OCS/ECS o superiore con 1 MB di memoria, nello specifico 512 KB di Chip e 512 KB di Slow RAM, più joystick o tastiera. Su questa base estremamente sobria, Echodrome costruisce un'arena shooter che, visivamente e meccanicamente, richiama Geometry Wars e Tron, ma con una propria identità ben definita. L'azione si svolge interamente all'interno di “The ECHODROME”, definita dallo stesso autore come “la suprema arena digitale di morte e distruzione”, un reticolato neon dove ogni stanza è un'arena chiusa da ripulire. L'obiettivo di ogni camera è semplice: eliminare tutte le ondate di nemici, evitare il contatto diretto, non calpestare le mine disseminate sulla griglia, individuare e abbattere il boss dell'arena per ottenere un bonus sostanzioso e accedere all'area successiva.

Dal punto di vista del controllo, il sistema è sorprendentemente ricco per un titolo OCS. Nella configurazione



standard, il gioco si lascia giocare con joystick singolo o tastiera: il movimento è gestito tramite leva o tasti cursore, mentre il fuoco è affidato al pulsante di sparo o al tasto Shift sinistro. Tenendo premuto il fuoco, la direzione di tiro viene bloccata e il personaggio continua a sparare automaticamente, consentendo di





GIUDIZIO FINALE

» Giocabilità 92%

Impeccabile dal lato estetico e incredibile per design dei livelli e del gioco. Perfette le combinazioni di gioco con pad, doppio stick e pad CD32.

» Longevità 92%

Un gioco che difficilmente verrà tolto dal "drive" dei nostri Amiga e che dimostra come si fanno i giochi per bene anche su configurazioni minime.

separare in parte il movimento dalla mira, una soluzione molto efficace per simulare un feeling à la twin-stick su hardware a un solo tasto. Per chi vuole l'esperienza completa, Echodrome supporta due setup avanzati: la modalità Twin Stick, con due joystick (uno per muoversi, l'altro per sparare in otto direzioni), e una modalità Four Button che sfrutta i joypad CD32 a quattro pulsanti per emulare un twin-stick "a pulsanti", assegnando direzioni di fuoco diverse ai vari tasti. Queste opzioni si attivano dal menu delle opzioni modificando lo stile di input e, nel caso della CD32, impostando opportunamente il tipo di joystick in emulatori come UAE, MiSTer o A500 Mini. Per un progetto jam, il livello di attenzione alle periferiche è insolito e mirato chiaramente a chi vuole una sensazione davvero arcade, fino all'uso di due Monster Joysticks in parallelo come raccontato da chi lo prova su A500+.

La struttura delle arene è organizzata come una sorta di labirinto di stanze collegate, con percorsi multipli che il giocatore deve "scegliere con saggezza", perché alcune camere sono nettamente più letali di altre. Ogni stanza ospita pattern diversi di nemici, trappole e



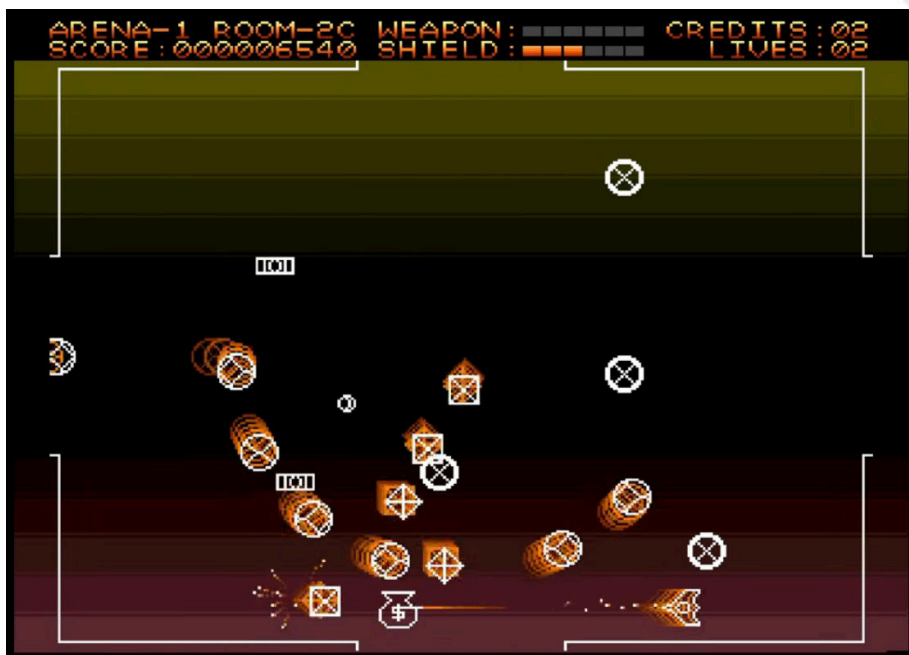


mine, e la scelta del percorso diventa a tutti gli effetti un elemento di strategia: non si tratta solo di sopravvivere all'ondata corrente, ma di pianificare in quale ordine affrontare gli scontri più duri per massimizzare le probabilità di arrivare al boss in condizioni accettabili. La presenza del boss di arena, da individuare e abbattere per ottenere un "big bonus" ed avanzare al blocco successivo, completa questa struttura quasi "roguelike" nella percezione, pur restando tutto rigorosamente arcade e senza elementi di progressione metanarrativa.

Tecnicamente, Echodrome sfrutta l'hardware con una pulizia che ha impressionato la scena. Lo fa in appena 880 KB di ADF (la versione 0.11 distribuita su itch.io), il gioco integra un motore in grado di gestire numerosi sprite, proiettili, esplosioni e deformazioni della griglia senza perdere in leggibilità né in fluidità

Sul fronte visivo, l'estetica TRON-like è il marchio di fabbrica di Echodrome. L'arena è una griglia luminosa su sfondo scuro, i nemici sono figure geometriche e vettoriali che richiamano l'immaginario digitale dei primi anni '80, mentre le esplosioni e le scie lasciano "echi" visivi che giustificano il titolo stesso. L'uso del colore è contenuto ma estremamente studiato con un contrasto netto tra elementi attivi (nemici, mine, proiettili) e pavimentazione dell'arena permette di mantenere alta la leggibilità anche nei momenti di massimo caos, problema non banale in qualunque arena shooter, a maggior ragione su un chipset grafico nato ben prima di questo tipo di design.

Dal lato sonoro, la colonna sonora e il sound design scelgono una direzione coerente. Brani elettronici tesi, costruiti per sostenere un ritmo di gioco costantemente alto, e una selezione di effetti (colpi, esplosioni, feedback



di hit) che puntano più sull'impatto e sulla chiarezza che sulla varietà timbrica fine a sé stessa. Con la versione 0.11 l'autore ha anche migliorato ulteriormente la musica, aggiungendo una schermata di game over e una di inserimento high-score "abbellite" con fuochi d'artificio e tracce dedicate, segno che il lavoro di rifinitura prosegue oltre la scadenza della jam.

L'insieme di scelte tecniche e di design ha prodotto un "arcade-style shooter" e dal feeling da sala giochi, ricco di extras e dettagli molto curati. Echodrome riesce così nell'intento che si era dato fin dall'inizio: non limitarsi

a essere un tributo all'era arcade, ma agire come se l'Amiga fosse ancora una piattaforma viva, capace di ospitare un twin-stick shooter moderno su una macchina con 1 MB di RAM e un joystick a un solo tasto. Per chi guarda l'OCS/ECS con occhi tecnici, è una piccola lezione di efficienza; per chi lo guarda con occhi da giocatore, è semplicemente uno di quei titoli che ti incollano allo schermo.

di **Roberto Del Mar Pirazzini**





DRAGON VIEW

Anno: 1994

Editore/Sviluppatore: Kemco/
Kotobuki System

Genere: RPG

Piattaforma: Super Nintendo

Bentrovati appassionati di giochi di ruolo e rarità: oggi si va a scoprire un sequel uscito solo su SNES davvero particolare.

Dragon View è quel seguito di Drakkhen che, sulla carta, sembra la risposta a tutti i problemi del primo episodio e che invece finisce per sistamarne molti, perdendo però una buona dose di quella stranezza ipnotica che rendeva unico l'originale. Tecnicamente rappresenta un salto avanti netto, sul piano ludico è molto più giocabile, ma come action-RPG per Super Nintendo del 1994 oggi va osservato con un minimo di distacco. Il suo predecessore era un gioco di ruolo europeo pubblicato da Infogrames su Amiga e Atari, poi portato prima su MS-DOS e, nel 1994, su SNES, terreno fertile per questo genere ma con una forte concorrenza da parte di ARPG e JRPG più classici. Drakkhen presentava meccaniche particolari, diversi problemi legati a un sistema di gioco rigido e una visione occidentale poco adatta al pubblico console, più vicina alla sensibilità del giocatore PC. Fortunatamente questo seguito, uscito solo per la macchina a 16 bit di Nintendo, cambia le carte in tavola, modificando in profondità molto dello stile originario.

La prima grande differenza è il sistema di gioco. Niente più party di quattro personaggi, comandi semi-automatici e menù rigidi. In questo episodio il giocatore controlla solo Alex, guerriero biondo dell'isola di Drakkhen, e il combat system è completamente action. Ci si muove in otto direzioni,



colpendo con la spada in base all'angolo di attacco – davanti, verso l'alto, dall'alto al basso in salto – e si ha a disposizione un attacco speciale che consuma una porzione degli HP. È presente anche un'arma secondaria a distanza, l'hauza, una sorta di boomerang che sulla carta amplia le possibilità, ma che in pratica resta spesso inutilizzata perché meno intuitiva della spada. A questo si aggiungono gli oggetti rapidi, in primis gli anelli elementali di fuoco, ghiaccio e fulmine, che consumano punti magia ma permettono sia di ripulire lo schermo con attacchi ad area sia di risolvere piccoli enigmi ambientali rompendo rocce, congelando superfici o raggiungendo zone altrimenti irraggiungibili. Sopra a tutto c'è un sistema di crescita classico: ogni nemico sconfitto e ogni boss portano esperienza, la barra di level-up sotto gli HP sale in modo molto leggibile,

le statistiche aumentano in maniera prevedibile e i nemici lasciano a terra cuori, stelle e Jade, la valuta del gioco. Tutto questo rende Dragon View infinitamente più immediato di Drakkhen, senza per questo scadere nell'azione pura e priva di profondità. La struttura del mondo riprende l'idea di base del predecessore con il mondo esterno in pseudo-3D con Mode 7, tutto il resto su mappe 2D. L'isola che il giocatore esplora è enorme, suddivisa in regioni ben riconoscibili – pianure, montagne, paludi, zone innevate, deserto – e punteggiata di città, templi e dungeon. Con l'avanzare dell'avventura si recuperano mappe che rivelano aree sempre più ampie e indicano chiaramente dove si trovano i punti chiave e a quale distanza. In teoria è una scelta che valorizza l'esplorazione, in pratica emergono subito due limiti: la quantità di ostacoli fisici che costringono a fare lunghi





giri e un ritmo di movimento non proprio scattante. L'isola è piena di catene montuose, rocce e barriere attraversabili solo da varchi specifici; spesso si vede una destinazione a pochi passi, ma per raggiungerla bisogna compiere una deviazione notevole. Alex, di suo, non è un fulmine e il gioco ha un debole per il backtracking: manda avanti e indietro tra città, templi e PNG, chiedendo di ripassare in luoghi già visitati per attivare nuovi eventi o recuperare oggetti chiave. I teletrasporti via costellazioni attenuano il problema, ma entrano in scena tardi, quando molta strada a piedi è già stata accumulata. Il risultato è un overworld concettualmente interessante ma zavorrato da andature lente e ritorni frequenti, che alla lunga pesano più del dovuto.

Quando Dragon View chiude il giocatore nei suoi spazi interni, però, la musica cambia. I dungeon sono costruiti con mestiere. Layout leggibili ma non banali, un minimo di verticalità, interruttori e chiavi usati in modo sensato, qualche deviazione per raccogliere extra e, soprattutto, una buona progressione nella densità degli scontri. I boss, in particolare, sono uno dei punti di forza:

grandi sprite che occupano buona parte dello schermo, pattern chiari, un mix riuscito tra osservazione e riflessi. Anche le città funzionano bene come hub per rifornimenti, dialoghi e avanzamento della trama, pur soffrendo di un problema di tono che attraversa un po' tutto il gioco: battutine meta, riferimenti alla cultura pop e gag volutamente fuori contesto convivono con momenti che vorrebbero essere serissimi o drammatici, creando uno stacco che non sempre funziona.

La gestione dei salvataggi è una delle scelte più discutibili. Se in Drakkhen era possibile salvare praticamente ovunque nell'overworld, qui i progressi sono vincolati a santuari di cura o guaritori erranti, sparsi e non sempre a portata di mano. Questa decisione, combinata con gli spostamenti lenti e il backtracking, fa sì che una morte sfortunata possa cancellare non solo l'ultimo spezzone giocato, ma intere sequenze di livellamento, esplorazioni e dialoghi. La tensione aumenta, certo, ma il rischio è di trasformare una sessione andata male in un'esperienza frustrante, soprattutto in un gioco che non è lunghissimo ma nemmeno brevissimo.

Sul fronte tecnico, Dragon View è

GIUDIZIO FINALE

» Giocabilità 80%

Sistema di gioco snello e "da console" ma pessimo sistema di salvataggio.

» Longevità 80%

Dura il giusto e quello che giocate è piacevole. Da riscoprire.

nettamente più rifinito di Drakkhen. Gli sprite sono più dettagliati, Alex gode di un'animazione fluida – con la chioma che svolazza quasi protagonista a sé – gli NPC non sono semplici manichini ma muovono la testa seguendo il personaggio, e il bestiario offre una buona varietà di nemici, dagli uomini-serpente agli imp, dai giganti agli scorpioni, fino a reaper e blob viscidici. I boss sono spettacolari per dimensioni e design, spesso composti da più parti animate. L'overworld sfrutta il Mode 7 con gradiente sul cielo, elementi del paesaggio che scalano in entrata e in uscita – rocce, alberi, ghiacci, mura – e una resa generale molto più pulita del vecchio Drakkhen. Gli interni dei templi, le foreste e le cascate mostrano un gusto per l'ambientazione che, per un titolo di fascia media su SNES, è più che dignitoso. Sul versante audio, la colonna sonora abbandona le sonorità new age spiazianti del primo capitolo per uno stile più classico per l'epoca, a metà strada tra orchestrale e rock. Alcuni brani di overworld, diversi temi di tempio e certe musiche della fase finale funzionano bene e restano in testa, mentre altre – come il tema delle città o la musica dell'ultimo dungeon – risultano piatte e poco memorabili. Almeno le battaglie hanno finalmente un accompagnamento musicale adeguato, e gli effetti sonori, dalle risate degli Imp ai versi dei boss, sono curati quanto basta.

Il gioco fatica di più sul piano narrativo e atmosferico. La trama muove da uno





sunto molto convenzionale: Alex, giovane guerriero, salva il maestro intrappolato in una caverna, ma al ritorno scopre che la sua amata Katarina è stata rapita dal mago Argos; dietro di lui si cela Giza, vero antagonista che mira a trascinare il mondo nell'oscurità usando un potente artefatto. Funziona come motore dell'azione, ma raramente sorprende. Giza, in particolare, è caratterizzato come cattivo sarcastico e sopra le righe, che dovrebbe risultare carismatico ma finisce per essere più irritante che minaccioso. Il legame con Drakkhen, poi, esiste ma è trattato quasi di sponda: un testo nel tempio di Keire cita chiaramente un eroe e tre compagni che in passato hanno riequilibrato il mondo, rimando diretto agli eventi del precedente episodio, ma oltre a qualche menzione ai draghi e pochi altri dettagli l'universo di Drakkhen viene ripreso in maniera molto superficiale. Spariscono quasi del tutto la mitologia bizzarra, il bestiario surreale, le costellazioni parlanti, quel senso di mondo alieno che, pur con tutti i difetti del primo gioco, contribuiva a un'atmosfera irripetibile. Qui ci si muove in un territorio action-RPG decisamente più tradizionale. Merita un accenno anche il finale, perché

rappresenta una scelta tecnica e di regia piuttosto estrema per un gioco su cartuccia SNES. Dopo lo scontro conclusivo, Dragon View si concede circa ventisei minuti di epilogo tra dialoghi non skippabili, sequenze di chiusura e titoli di coda. È quasi una dimostrazione di forza su quanta narrativa si possa infilare in ROM, ma pad alla mano può diventare estenuante, soprattutto se lo si confronta con finali più concisi ma emotivamente efficaci di altri RPG dell'epoca.

Messo tutto sul piatto, Dragon View è un action-RPG competente da riscoprire, con un sistema di combattimento solido, dungeon generalmente riusciti e una produzione tecnica superiore alla media delle seconde linee su Super Nintendo. Allo stesso tempo, è un gioco che tende a intralciare se stesso: l'overworld enorme ma lento, il backtracking frequente, i limiti del sistema di salvataggio, una trama poco incisiva e la perdita quasi totale dell'atmosfera stramba di Drakkhen lo tengono lontano dal livello dei veri imprescindibili del genere. Chi apprezza Drakkhen soprattutto per la sua follia probabilmente troverà Dragon View "troppo normale"; chi invece cercava un'evoluzione più giocabile del primo



capitolo qui la trova, ma deve accettare alcune scelte di design che, ancora oggi, fanno alzare più di un sopracciglio.

di **Roberto Del Mar Pirazzini**





ESCAPE FROM PETSCII PLANET

Anno: 2026

Editore/Sviluppatore: Jimbo

Genere: FPS

Piattaforma: Commodore 64

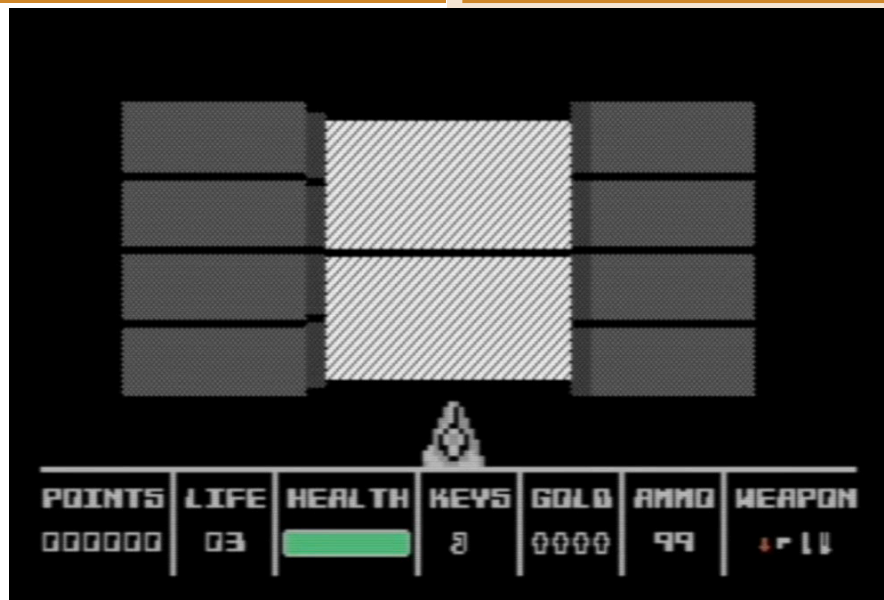
Sito web: <https://tinyurl.com/ch356rdm>

Un vero FPS ray-casting ispirato a Wolfenstein 3D, costruito sopra l'estetica PETSCII e pensato per essere giocato e "spremuta" nel 2026. Ecco cosa succede quando carichiamo Escape From PETSCII Planet sul nostro biscottone.

Lo sviluppatore, Jimbo, parte da un'idea già di per sé folle. Quella di portare su C64 un motore in prima persona nato originariamente per il Commodore PET (folle vero?), e poi potenziarlo fino a farlo diventare un dungeon shooter completo, con più piani, chiavi, segreti e un loop da "cerca l'uscita" molto classico. Il risultato è un gioco in cui ci ritroveremo bloccati su un pianeta remoto, imprigionati in un dungeon sotterraneo e pronti a tutto per sopravvivere.

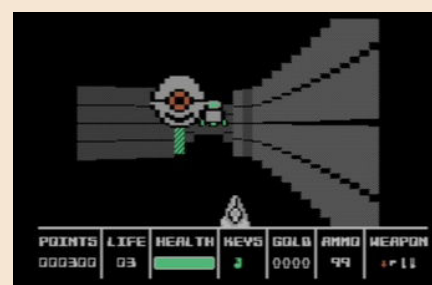
Dovremo esplorare corridoi, porte chiuse, passaggi segreti, raccogliere armi, munizioni, chiavi, oro e trovare l'ascensore che ti porta sempre più in alto, fino alla fuga. Gli antagonisti sono robot alieni ostinati che presidiano i corridoi e popolano i vari livelli; ogni piano è progettato come una piccola arena labirintica con la sua dose di tensione e "mayhem".

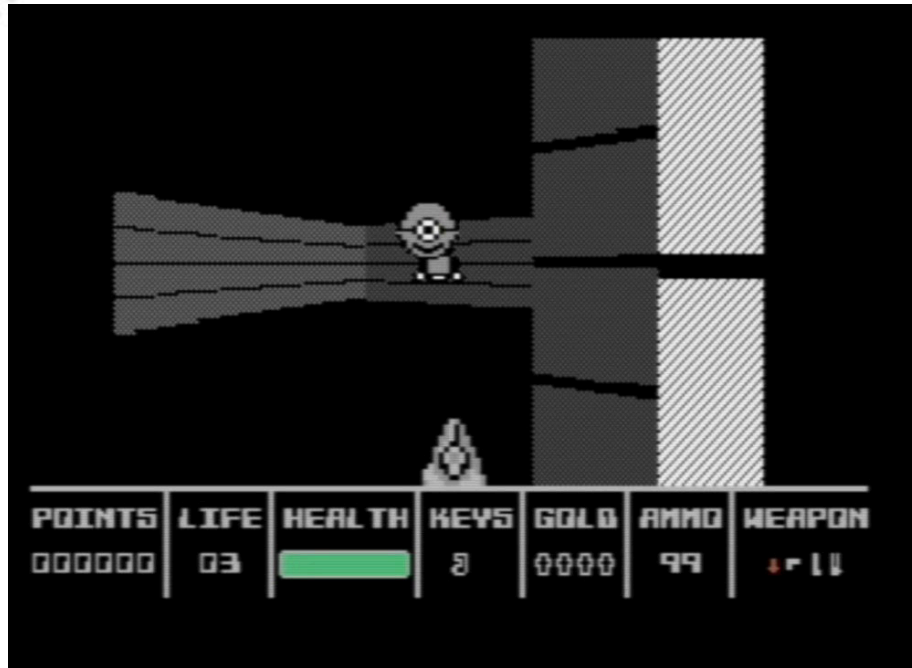
Dal punto di vista strutturale, il pacchetto include il gioco completo con la Campagna A: cinque piani con oltre 100 nemici complessivi, distribuiti tra più difficoltà selezionabili. Oltre alla modalità normale, ci sono un extra-hard dedicato a chi vuole soffrire davvero, una modalità Gold Rush a tempo, in cui il focus si sposta sulla



raccolta di oro in velocità, e un Zeus Mode in cui diventi sostanzialmente onnipotente. La progressione viene salvata automaticamente alla fine di ogni piano, scelta molto pratica che rende la struttura a livelli più digeribile su hardware reale, senza costringerti a run maratona ogni volta.

Il cuore del progetto è il motore di ray casting 8-bit, esplicitamente ispirato a Wolfenstein 3D e adattato ai vincoli del C64. Lo stile non punta al finto-realismo, ma abbraccia la grammatica PETSCII, e quindi ci ritroveremo pareti, elementi e UI costruiti con caratteri grafici, creando un look ultra-stilizzato ma leggibile, che sta a metà tra un vecchio roguelike e un raycaster primitivo, con una velocità che, per essere il nostro C64, è notevole. Va sottolineato come il motore riesca a dare una sensazione di movimento fluido pur restando entro i limiti della macchina. Sul fronte del design, l'autore non si limita a una





sequenza di corridoi anonimi. Ogni piano ha porte bloccate da chiavi colorate, muri segreti dietro cui si celano oggetti importanti e, naturalmente, i classici elementi da dungeon crawler. A corollare il tutto ci sono le munizioni sparse, armi da prelevare, zone di rischio e layout pensati per farti girare in tondo fino a quando non inizi a “leggere” davvero la mappa.

Un plus non da poco è la presenza di un Level Designer separato, fornito come tool dedicato per creare e giocare i propri livelli personalizzati. L'autore invita esplicitamente i giocatori a progettare mappe e condividerle con lui, trasformando il gioco in una piattaforma potenziale per contenuti generati dalla community. In un ecosistema come quello C64, dove gli editor spesso sono esterni e scomodi, avere un designer pensato ad hoc per questo engine apre scenari interessanti per remix, jam e release future.

Sul piano dell'accessibilità hardware, Escape From PETSCII Planet è sorprendentemente flessibile. Può essere giocato con tastiera, mouse o joystick, come confermato nei post promozionali sui gruppi C64/Ultimate.

Il file distribuito è un classico .d64, compatibile con hardware reale e soluzioni come C64 Ultimate, e il gioco è disponibile su itch.io in formula “pay what you want”... tecnicamente gratuito, ma con la possibilità di supportare lo sviluppatore se ne siamo convinti (cosa che consiglio di fare sempre).

Complessivamente, Escape From PETSCII Planet è un esperimento riuscito di portare il linguaggio degli FPS dei '90 su un 8-bit, senza cercare di imitare Doom, ma andando dritto al cuore wolfensteiniano del genere. L'estetica PETSCII gli dà una personalità visiva forte, la struttura a campagne e modalità extra ne aumenta la longevità, e il level editor lo spinge oltre il semplice “gioco-evento” da provare una volta e archiviare. Per chi mastica già raycaster su hardware ridotto, è un must-try; per chi viene dal C64 più tradizionale, è una dimostrazione pratica di quanto spazio di manovra ci sia ancora quando decidi davvero di spingere l'8-bit fuori dalla comfort zone.

di **Carlo Nithaiah Del Mar Pirazzini**

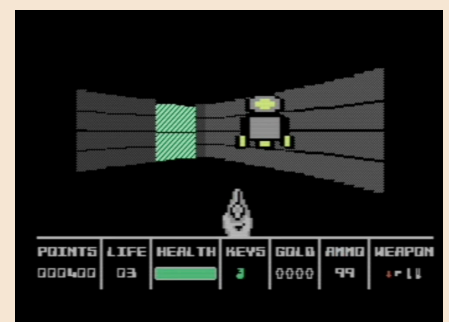
GIUDIZIO FINALE

» Giocabilità 90%

Veloce e con un buon design di gioco.

» Longevità 90%

Ben bilanciato e godibile. Il Level Editor è una bella sorpresa, aumentato di fatto la longevità del prodotto grazie allo sviluppo di nuove mappe e di possibili Jam.





JUNK RUNNER 64

Anno: 2026

Editore/Sviluppatore: Team Ultra Rare

Genere: Adventure

Piattaforma: Nintendo 64

Sito web: <https://drive.google.com/file/d/1MMoweVrkuQkwsFSR0XqIownXE-CicYTV/view>

Junk Runner 64 è un homebrew per Nintendo 64 sorprendentemente ambizioso. Un open world avventura/running game che sfrutta molto bene e in modo sensato l'hardware, sia come scala delle mappe sia come design di gioco.

Nel gioco interpreteremo Jax, deciso a salvare il fratello Samson malato riparando le macchine del villaggio con pezzi di rottami sparsi nel mondo. Ogni macchina richiede almeno un componente. Il primo è vicino, gli altri sono sparsi in aree lontane e il robot nel centro del villaggio fornisce mappe e direzioni, rendendo l'esplorazione guidata ma non banale.

Dopo aver riparato l'hoverbike, il gioco comincia sul serio. Potremo esplorare un grande ambiente aperto diviso in tre macro zone – fiume a sud est, dune di sabbia a nord, città deserta a sud est – ognuna contenente i pezzi per una macchina specifica, riducendo il backtracking ripetitivo. Solo l'ultimo oggetto ha pezzi distribuiti in tutto il mondo, spingendoci a sfruttare la mappa in modo più sistematico.

Nella mappa sono disseminati circuiti con checkpoint. Alcune piste richiedono da uno a tre giri completi e una sola tra queste nasconde un pezzo fondamentale per la riparazione finale, mentre le altre ricompensano con 20 "nuts". Le nuts sono la valuta del gioco e servono esclusivamente per potenziare la moto:

- Boost di velocità attivabile a comando
- Scanner per individuare le nuts vicine

• Teletrasporto rapido verso punti chiave della mappa

Ne bastano 50 per comprare tutto, quindi il "grinding" è limitato! Il focus resta sull'esplorazione e sulla scoperta piuttosto che sulla raccolta compulsiva. La mappa si rivela man mano che esploreremo, ma potremo anche trovare mappe nascoste che scoprono intere porzioni in un colpo solo, velocizzando la navigazione. Un primo giro "alla cieca" richiede circa un'ora, ma una volta imparata la disposizione di oggetti e percorsi il gioco diventa molto più rapido da completare.

Tecnicamente Junk Runner 64 è notevole! Un open world di dimensioni tali che si potrebbe considerare qualcosa che nemmeno molti titoli retail dell'epoca avevano osato. La ROM pesa circa 16,2 MB, segno di un uso importante di risorse grafiche e logiche su un hardware notoriamente





GIUDIZIO FINALE

» Giocabilità 88%

Mondo molto vasto da esplorare anche se con poche aree. Il sistema di controllo è interessante e mai noioso ed invoglia a giocare ed esplorare, ma anche a girovagare per rilassarsi!

» Longevità 80%

Ci sono diversi bug che minano la "durabilità" del prodotto. Nulla di estremo, ma sono fastidiosi.

limitato in termini di memoria e banda. Il modello di guida dell'hoverbike è uno degli elementi meglio riusciti. Rimbalza alla grande sul terreno, sfrutta la conformazione del suolo in mondo naturale e il tutto è molto piacevole (anche girare a vuoto). Dal punto di vista del level design, le tre macro zone sono abbastanza diversificate da evitare la sensazione di deserto piatto: tra fiume, dune e città, lo scenario cambia e mantiene l'attenzione del giocatore. Graficamente il gioco è di alto livello per un homebrew. I modelli dei personaggi ricordano uno stile action 3D dell'epoca ma "ripulito", con personaggi che ricordano una versione migliorata di certi platform 3D più noti. Anche se la mappa è tecnicamente vasta e in parte vuota, la disposizione degli elementi fa sì che l'ambiente non risulti ripetitivo: c'è sempre qualcosa da osservare e la transizione tra aree è abbastanza netta da mantenere viva la curiosità.

Non manca però un po' di "problemi" tecnici. L'area del villaggio soffre di rallentamenti visibili che influenzano anche la musica, rendendo l'esperienza a tratti pesante in quel punto. In alcune occasioni si rischia di cadere attraverso

la geometria; esiste però un failsafe che riporta subito il personaggio in superficie, quindi il bug resta fastidioso ma non rovina la partita. E' presente anche un crash, con potenziale perdita di tutti i progressi di gioco.

Questi problemi sono tipici di un progetto molto ambizioso per l'hardware, ma mitigati dalla presenza di fallback intelligenti (come il recupero automatico quando si cade fuori dal mondo). Junk Runner 64 è stato sviluppato da Team Ultra Rare (James Lambert, Pyroxene, SapphireOnze, Beta_dynast, Terzdesign, Kaelin) per la N64brew Game Jam 2025, uscito il 2 febbraio 2026 e vincitore del primo posto nella competizione. La ROM e il codice sorgente sono disponibili pubblicamente (ROM scaricabile e codice su GitHub), cosa che rende il progetto anche un'ottima base di studio per sviluppatori homebrew interessati a open world e veicoli su N64.

Per chi è interessato alla scena homebrew o allo sviluppo su N64, è un titolo da studiare oltre che da giocare.

di **Roberto Del Mar Pirazzini**





SOUL BLAZE

Anno: 2026

Editore/Sviluppatore: Rogério
Perdiz, Subin Karkani

Genere: Adventure/Action

Piattaforma: Sega Dreamcast

Sito web: <https://darkpath.itch.io/soul-blaze>

Soul Blaze è un nuovo progetto/minigioco per Dreamcast che mette in mostra, con sorprendente ambizione tecnica, ciò che il motore Spiral Game Engine è in grado di fare oggi su hardware reale SEGA.

E' concepito come mini game ad alto ritmo, non come gioco completo, e nasce esplicitamente per mostrare "potenza e flessibilità" di Spiral su Dreamcast. È stato realizzato per la jam Dream Disc '25 e distribuito come immagine CDI e come pacchetto ZIP, in modo da essere usato sia su console reale sia in emulazione.

Nel gioco interpreteremo Rhyo, giovane artista marziale alla ricerca di un raro "flownder" per una missione personale, ma questa cornice narrativa serve soprattutto a dare un contesto a un villaggio esplorabile e a una serie di combattimenti contro nemici meccanici. Tutto è costruito per far vedere fluidità di movimento, gestione delle collisioni e resa grafica del motore. Tecnicamente, l'elemento più importante di Soul Blaze è il fatto che gira sul Spiral Game Engine, un motore in sviluppo specifico per Dreamcast. Gli sviluppatori confermano che Spiral è già dotato di "quite a lot of features", ma non è ancora pronto per il rilascio pubblico; l'obiettivo è una release gratuita, ma solo quando sarà considerato sufficientemente solido. Questo perché ci sono

diversi problemi e alcuni glitch che lo rendono ancora "traballante" (soprattutto in emulazione).

Questi indizi mostrano un engine che lavora abbastanza vicino all'hardware, con timing e pipeline video sensibili, cosa tipica dei progetti che vogliono massimizzare la resa su una macchina specifica come Dreamcast.

Sul piano del gameplay, Soul Blaze offre un mix di esplorazione di un villaggio, con NPC con cui parlare per ottenere indizi sul flownder, combattimenti a base di pugni, calci e combo contro ondate di granchi meccanici che infestano l'area e battaglie coi boss finali piuttosto articolata ma piacevole.





Il ritmo è dichiaratamente divertente ma adatto ai giocatori più scaltri. Il focus è sulla prontezza dei controlli e sulla risposta del motore a molte entità sullo schermo più che su una progressione lunga. L'obiettivo è permettere al giocatore di percepire rapidamente animazioni fluide, effetti grafici, transizioni e gestione dei combattimenti. L'arte e il design visivo sono firmati da Rogério Perdiz, ed è davvero un lavoro di pregio e di cura tecnica con un forte identità visiva su ambienti 3d, colori e design dei villaggi. I personaggi presentano tutti uno stile coerente, in particolare i granchi meccanici e il boss, che combinano estetica arcade con dettagli moderni.

Il fatto che alcuni emulatori abbiano problemi con shadow volumes e FMV indica che vengono usate feature grafiche non banali, pensate in primis per l'hardware reale, e che Soul Blaze è progettato per "spingere" un po' il Dreamcast, non solo una tech demo visiva.

La colonna sonora è composta da Subin Karkani, che si sposa perfettamente all'ambientazione gioco. Insieme a grafica e animazioni, contribuisce a dare al prodotto un profilo audiovisivo

sopra la media per un progetto jam homebrew.

Per chi è interessato all'homebrew Dreamcast e alla tecnologia dietro i nuovi motori per console retro, Soul Blaze è meno "solo un gioco" e più una vetrina tecnica ben confezionata, che lascia intravedere cosa potrà fare la versione matura di Spiral nei futuri progetti indie per Dreamcast. Ci auguriamo di vedere in aggiunta una maggiore compatibilità con il mondo emulativo e magari di espandere ancora di più il mondo di gioco. Per ora possiamo solo dire bravi al team e dare un nostro primo voto al titolo.

di **Roberto Del Mar Pirazzini**



GIUDIZIO FINALE

» Giocabilità 80%

Controlli precisi e un buon game design, con una grande cura tecnica.

» Longevità 70%

C'è ancora molto poco e sono presenti diversi bug, soprattutto in emulazione.





ZEZITO AND THE LOST STARS

Anno: 2025

Editore/Sviluppatore: Iapetus

Genere: Platform

Piattaforma: Acorn Electron
32K

Sito web: <https://iapetus2019.itch.io/zezito-and-the-lost-stars>

“Zezito and the Lost Stars” è un platform per Acorn Electron 32K che punta a valorizzare in modo esplicito l'hardware, combinando un'estetica molto colorata con un gameplay accessibile che ci permette di rivalutare l'Acorn Electron, storicamente trattato come il cugino povero della BBC Micro.

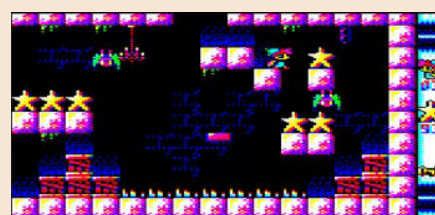
La premessa è semplice ma azzeccata: la strega Malandra ruba 99 stelle e le chiude nel suo castello; se nessuno le recupera, il cielo notturno diventa un CRT spento. Zezito approfitta del fatto che la strega è al “meeting annuale delle streghe” (folle vero?) e si infila nel castello come un piccolo Indiana Jones 8-bit, deciso a riportare luce nel firmamento una stellina alla volta. Il risultato è un platform labirintico a schermate singole collegate, dove ogni stanza è una micro-puzzle room alla Manic Miner, ma ripulita dalle cattiverie gratuite di Miner Willy e più vicina allo “sfida ma non ti odio” di un buon gioco da computer anni '80.

Una delle cose più sorprendenti è quanto il titolo “smascheri” il potenziale dell'Electron. Una delle migliori grafiche mai viste sulla macchina con sprite completamente mascherati (addio orrendi artefatti EOR quando due sprite si sovrappongono), movimento fluido, palette sfruttata con coraggio. Per capirci, sembra quasi di vedere un Electron che

prova a fare il Sonic 8-bit del suo catalogo – non per velocità, ovviamente, ma per impatto visivo rispetto allo standard storico del sistema.

Dal lato tecnico, Iapetus gioca sporco in senso buono. Per riuscire a tenere insieme Mode 2, sprite mascherati e un gioco completo in 32K, ricorre a un trucco molto “da demoscener”. Usa circa 8 KB nello spazio schermo nascosto, 100 righe di display sopra l'area giocabile, per ospitare codice e dati. Niente backbuffer, niente luxury, soltanto un engine di sprite affinato in anni di sperimentazioni, che lui stesso definisce “il migliore che abbia scritto finora”. È un po' come guardare quei giochi C64 che sfruttano il bordo overscan o i raster split per infilare più roba dove non dovrebbe starci: qui l'Electron riceve lo stesso trattamento di rispetto tecnico.

Il controllo, come da tradizione 8-bit british, ha le sue piccole idiosincrasie. Si nota che, se senti di saltare con un blocco subito





GIUDIZIO FINALE

» Giocabilità 80%

Richiede qualche accorgimento per prenderci la mano, ma è piuttosto ben fatto e con una buona varietà dei livelli. Graficamente è gradevolissimo!

» Longevità 80%

Una buona via di mezzo. Non è estremamente difficile, ma nemmeno una passeggiata.



sopra la testa, il movimento orizzontale si comporta in modo “strano”, e che i salti tra stanze, soprattutto passando dal top di una schermata al bottom di quella successiva, richiedono un timing un po’ più ragionato. L'autore risponde in modo molto onesto. Il primo comportamento è un compromesso volontario, un piccolo prezzo da pagare per far quadrare il resto dell'engine; il secondo è nella sua lista di potenziali miglioramenti futuri. È lo stesso tipo di rapporto amore/odio che molti di noi hanno avuto con i salti di Jet Set Willy o Monty on the Run: un po’ “quirk”, ma parte del carattere del gioco.

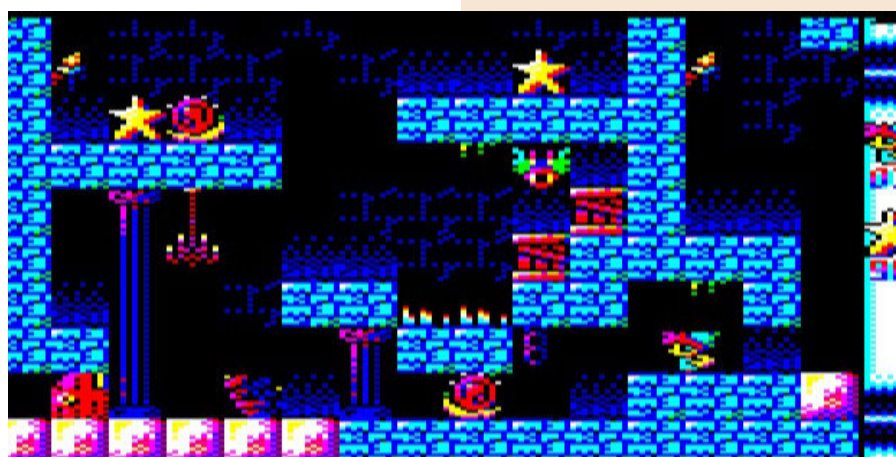
L'atmosfera generale è quella di un platform “tenerello ma cattivello”. Non è da “joystick assassino e pixel perfect” ma non è nemmeno una passeggiata. Le lumache spinose e i pipistrelli fanno il loro lavoro sporco, ma il tono resta leggero, quasi da cartone animato pomeridiano.

Sul fronte distribuzione, ZeZito lo ha rilasciato nel 2025 su itch.io con una versione disk .ssd e due varianti tape .uef (inclusa una fast-load per non impazzire ai tempi di caricamento), il tutto perfettamente compatibile in real hardware o in emulazione.

In definitiva, “ZeZito and the Lost Stars” è un piccolo atto d'accusa contro l'idea che l'Electron fosse destinato solo a clone scolastici

della BBC. Qui sembra di vedere cosa sarebbe successo se, ai tempi, qualcuno avesse deciso di trattarlo come un mini-C64 anziché come il fratello povero. È un gioco che sta a metà tra il platform da rivista e la tech demo con anima, e che funziona proprio perché riesce a farti dimenticare l'etichetta “macchina minore” e ti lascia solo con una domanda molto semplice: “ok, cosa altro potete tirare fuori da questo coso?”.

di **Giampaolo Moraschi**





BURGERTOM

Anno: 2026

Editore/Sviluppatore: Swap d0

Genere: Platform

Piattaforma: Atari Jaguar

Sito web: [https://](https://swapd0.itch.io/burgertom)

swapd0.itch.io/burgertom

BurgerTom è una nuova conversione homebrew di BurgerTime per Atari Jaguar, realizzata dal coder spagnolo swap d0 (già noto sulla scena per altri porting e remake come Jumping Jack), pensata per far riscoprire un grande classico Data East sul sistema a 64 bit di casa Atari con una veste tecnica adeguata all'hardware e un feeling quanto più possibile fedele all'originale arcade.

La storia resta sempre la stessa del classico arcade: nei panni dello chef (qui ribattezzato Tom Tomatoes) si corrono piattaforme e scale su livelli multipli tentando di completare enormi hamburger, camminando sugli ingredienti per farli precipitare verso il piatto in basso, mentre si evita un piccolo esercito di "cibi ostili" che pattugliano lo schermo.

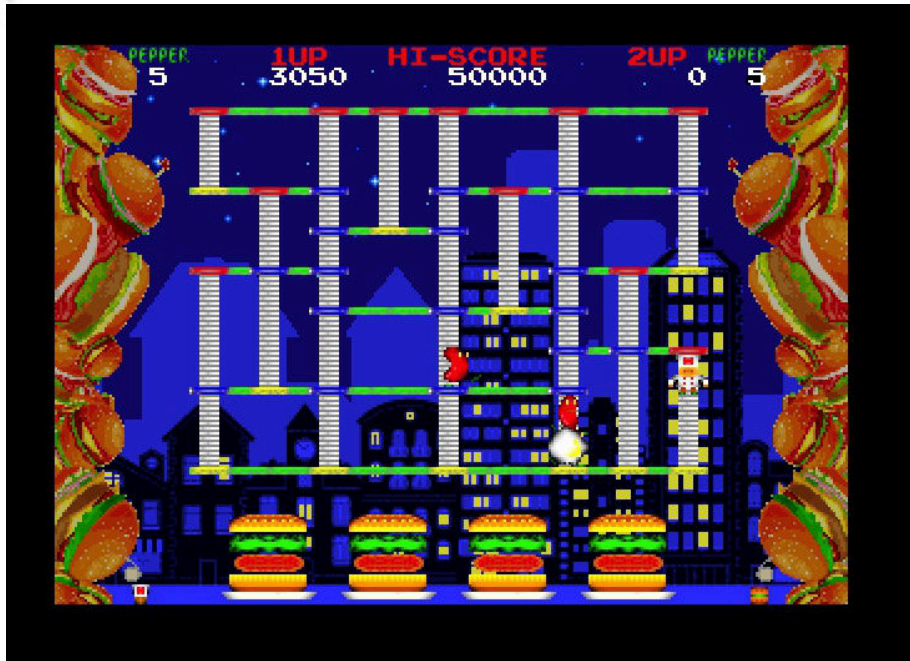
Ogni piano completato contribuisce alla costruzione del burger, con la pressione degli ingredienti che può schiacciare o far cadere gli avversari sottostanti, riproducendo la dinamica classica di BurgerTime. Il ritmo è quello tipico degli arcade dell'era dorata degli anni '80, fatto di pattern da memorizzare, gestione accurata delle traiettorie e una difficoltà che cresce rapidamente dopo i primi livelli.



Su Atari Jaguar, BurgerTom si presenta come una ROM stand alone (circa 265 kB) da caricare su Skunkboard o JagGD, quindi chiaramente pensata per l'hardware reale e per chi è abituato agli strumenti della scena Jag.

La descrizione dell'autore promette "enhanced performance" e controlli





GIUDIZIO FINALE

» Giocabilità 85%

Perfetto nel suo essere spietatamente arcade.

» Longevità 85%

E' un piacere farci una partita di tanto in tanto.

fluidi, puntando sulla reattività dell'input e su una grafica vivace che richiama il cab originale, con palette sature e sprite ben leggibili anche sulle uscite RGB tipiche delle configurazioni Jaguar moderne.

Promessa mantenuta al 100%. Siamo di fronte ad un titolo solido e in linea con il gioco originale.

L'obiettivo dello sviluppatore è quello di colmare una lacuna nel catalogo ufficiale, offrendo una versione casalinga di un classico, con tutta la comodità di una ROM caricabile su flash-cart, con aggiunta una presentazione curata e un controllo perfetto in real hardware. Se amate BurgerTime e avete fame di giochi nuovi su Jaguar, questa è una delle conversioni che vale la pena avere. ma vivace libreria di titoli "nuovi" per Jaguar che fanno leva sulle icone degli anni '80.

di Alex "LEX" Fridman





TONY DO IT!

Anno: 2026

Editore/Sviluppatore:

Monochrome Production

Genere: Puzzle game

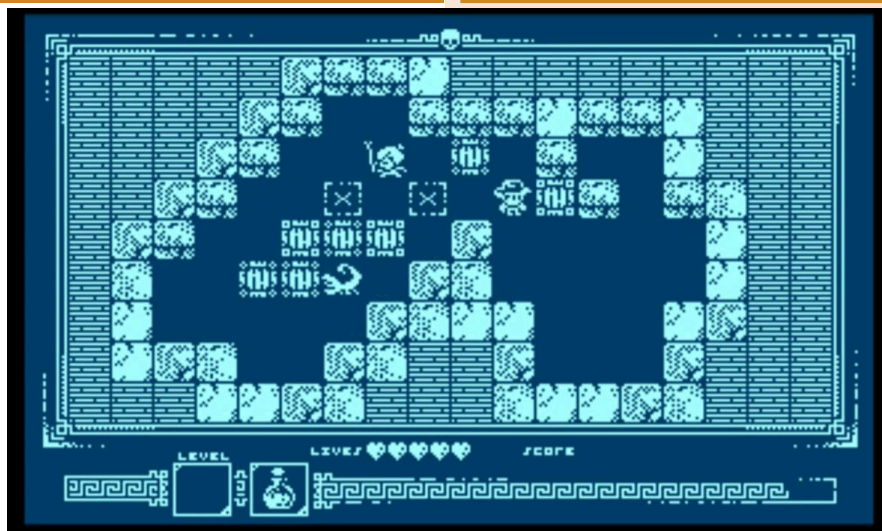
Piattaforma: Commodore 64,
Atari 8bt

Sito web: <https://monochrome-productions.itch.io/>

Tony Do It! è il terzo capitolo della saga sviluppata da Monochrome Production (dopo Tony Montezuma's Gold e Go! Tony Go!), ispirato alle meccaniche di Sokoman. La storia parte dal finale di Montezuma's Gold, e vede il buon vecchio esploratore che si ritrova intrappolato in una stanza con l'uscita bloccata da casse da sistemare strategicamente sulle croci designate.

Le meccaniche sono le stesse del Sokoban, ma arricchite da innovazioni che lo distinguono. Tony può non solo spingere le casse, ma anche tirarle usando il tasto Fire del joystick, aprendo possibilità di movimento più fluide e creative per risolvere i livelli. Questo dualismo "push-pull" permette di accedere a zone altrimenti inarrivabili e di creare configurazioni complesse, richiedendo una pianificazione attenta fin dal primo momento. A complicare le cose intervengono scorpioni e nativi erranti che pattugliano lo schermo. Se Tony li tocca, prosciugano rapidamente il tempo disponibile, introducendo un elemento di urgenza e rischio che trasforma ogni mossa in una decisione critica.

Una delle chicche più apprezzate è la gestione dei nemici, che aggiunge profondità tattica senza appesantire



l'esperienza. Invece di evitarli passivamente, i giocatori possono intrappolarli in spazi chiusi spingendo casse attorno a loro. Una volta bloccati, muoiono dopo qualche secondo, liberando lo spazio e permettendo di progredire.

Questo approccio strategico premia l'osservazione e la pazienza, rendendo i puzzle non solo logici ma anche dinamici. I livelli sono compatti e progressivi, con sfide che crescono





GIUDIZIO FINALE

» Giocabilità 85%

Meccaniche del Sokoban rinnovate e con una buona strategia da utilizzare.

» Longevità 75%

In partite brevi regala il meglio... in sessioni più "corpose" può risultare ripetitivo.

gradualmente di difficoltà, ma restano sempre risolvibili senza frustrazione eccessiva grazie ai cheat integrati come F3 per tempo infinito o F5 per saltare un livello, utili sia per i principianti che per chi vuole sperimentare soluzioni alternative.

La grafica in pixel art, curata da Rafał Dudek, sfrutta al massimo la palette colori della macchina con sprite vivaci e animazioni fluide, creando un'atmosfera avventurosa che richiama le glorie degli anni '80. Il codice di Popmilo garantisce un framerate stabile anche nei momenti più caotici, mentre la colonna sonora di Manganoid, con jingle catchy e effetti sonori netti, completa un pacchetto sonoro che non delude gli appassionati. Rilasciato a marzo 2026 e distribuito gratuitamente su itch.io in formato .prg, è compatibile sia con hardware originale che con emulatori come VICE su C64 o Altirra su Atari 8bit, rendendolo accessibile a un pubblico ampio senza barriere economiche.

Forse le meccaniche alla lunga potrebbero risultare noiose, ma resta

comunque un buon titolo adatto agli appassionati di puzzle game su entrambe le piattaforme.

Ad ogni modo, questo episodio si distingue per la sua immediatezza. È un titolo perfetto per sessioni brevi ma soddisfacenti, ideale per fan del puzzle solving retro che apprezzano l'innovazione senza stravolgimenti.

di **Giampaolo Moraschi**





WINGS OF FEAR

Anno: 2025

Editore/Sviluppatore: Roartj

Genere: Shoot'em up

Piattaforma: Amiga AGA

Sito web: <https://roartj.itch.io/wings-of-fear>

Wings of Fear è un sogno personale diventato realtà, plasmato con pazienza artigianale e rifinito insieme alla community fino a trasformarsi in uno shoot 'em up verticale capace di stare fianco a fianco con i classici a cui si ispira. Nato dalla passione di Roar "Roartj" Tjørhom e realizzato con Scorpion Engine per le macchine AGA, il gioco ci mette ai comandi di un Sopwith Camel bianco e oro in piena Prima guerra mondiale, lanciati in una missione segreta alla ricerca di una base tedesca nascosta tra cieli tempestati di piombo, città bombardate e paesaggi che alternano fedeltà storica e suggestioni quasi "weird war". Fin dai primi minuti si capisce che non siamo davanti a un semplice esercizio nostalgico. Wings of Fear ragiona come un coin op anni '80/'90, con una struttura scandita da livelli, boss, missioni speciali e bonus stage, ma lo fa con una sensibilità moderna nella costruzione del ritmo. Le sei missioni principali, affiancate da due impegnative bombing run e da una serie di stage bonus, dosano con attenzione dogfight serrati, fasi di bombardamento al suolo e momenti in cui il gioco ti lascia tirare il fiato, dandoti l'occasione di rimpinguare scorte di vite e potenziamenti prima della prossima ondata di furia antiaerea. I bonus stage non sono un semplice extra, ma parte integrante del bilanciamento: se non sfrutti bene queste "zone franche", i boss – soprattutto quelli a più fasi e con pattern di missili guidati – diventano rapidamente muri invalicabili.





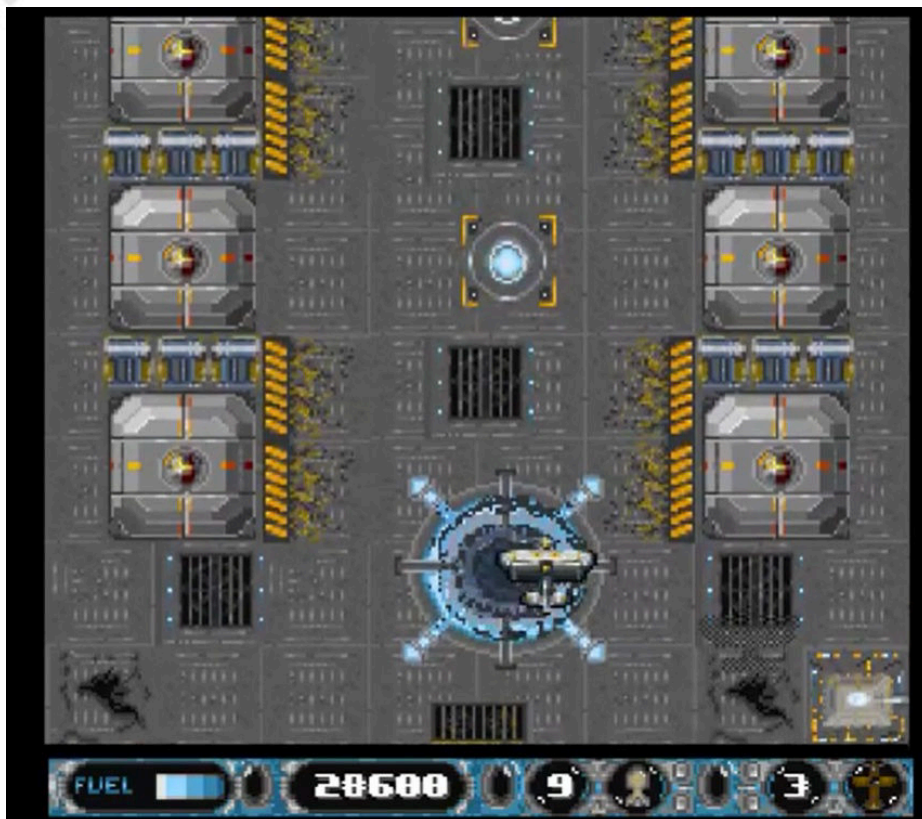
GIUDIZIO FINALE

» Giocabilità 80%

Feeling molto anni'80 e un solido gameplay accompagnano per tutta l'avventura.

» Longevità 75%

Un titolo che non regala nulla in termine di gioco. Richiede attenzione e una conoscenza dei pattern degli avversari.



La gestione dei pickup è una delle chiavi del gameplay. I barili rossi fanno da ancora di salvezza per il carburante, quelli blu curano i danni e permettono di potenziare l'arma principale, e ogni scelta di buttarsi o meno su un power up, magari in mezzo a una salva di colpi, è un continuo gioco di rischio e ricompensa. A differenza di tanti shooter "comfort" di oggi, Wings of Fear non ha paura di farti soffrire. Lo si percepisce già dal primo boss, capace di mettere in crisi i meno allenati. Allo stesso tempo, però, la difficoltà non è mai gratuita, esistono strategie precise, come sfruttare la parte bassa dello schermo per "buttare fuori" i missili a ricerca, e chi si prende il tempo di leggere le traiettorie e studiare il layout trova in cambio quella soddisfazione tipica dei titoli arcade ben pensati.

Sul piano visivo, Wings of Fear presenta 178 colori a schermo che vengono sfruttati non per fare sfoggio di saturazione, ma per disegnare ambienti leggibili e pieni di atmosfera come mari agitati, campagne martoriate, installazioni militari, fino a scenari via via più strani e inquieti man mano che si entra nel cuore della base nemica. Sprite e fondali,

realizzati con strumenti moderni come Aseprite, Resprite e Procreate, sono tutti disegnati a mano; l'autore ha scelto consapevolmente di non usare grafica generata da AI, e questa coerenza stilistica si sente. Il gioco ha quell'aria da produzione "vera", quasi commerciale, più vicina a un titolo da scaffale di metà anni '90 che a un hobby project improvvisato, grazie anche al lavoro di





animazione sui nemici e alle esplosioni differenziate per i vari bersagli a terra. La colonna sonora è l'altro grande pilastro dell'esperienza. Il tema principale firmato da Márton Imre Gáspár introduce le missioni con un tono epico che sa di briefing prima del decollo, mentre in gioco le tracce di Frank "Curt_Cool" Ewil Larsen alternano ritmi serrati e toni più cupi, seguendo il crescendo di tensione dei livelli di bombardamento e degli scontri con i boss. Non è un semplice accompagnamento, ma un tappeto sonoro che restituisce quella vibrazione tipica della scena Amiga: più di una recensione lo sottolinea come uno dei punti di forza del titolo, tanto da essere spesso citato nei video come elemento che contribuisce a quell'impressione di "modern revival" perfettamente in linea con l'anima della macchina Commodore. Lo Scorpion Engine viene sfruttato in fetchmode 3, con l'area di gioco spostata sulla destra dello schermo per spremere prestazioni extra dal chipset AGA: una scelta che a qualcuno potrà sembrare una piccola eccentricità estetica, ma che si traduce in uno scorrimento più fluido e in una maggiore stabilità del frame rate sulle macchine dotate di sufficiente Fast RAM. Per non lasciare indietro chi preferisce un layout più tradizionale, la versione 4.8 mette inoltre a disposizione un build in fetchmode 1, con schermo centrato e un occhio di riguardo alla compatibilità, seppur con qualche compromesso in termini di velocità nelle situazioni più affollate. Il gioco richiede un'Amiga AGA con almeno 2 MB di Chip e 2 MB di Fast, ma dà il meglio di sé su configurazioni con 4 MB di Fast e CPU 68040 o superiore, sia su hardware reale sia su FPGA ed emulazione; non a caso, i pacchetti curati da AmigaVault e soci includono versioni WHDLoad, installazioni per hard disk, CD32 e perfino configurazioni dedicate per A500 Mini, frutto di lunghi test per

evitare rallentamenti e problemi di memoria.

Naturalmente non è un titolo per tutti. La difficoltà elevata (specie se affrontato senza sfruttare al massimo i bonus stage e senza studiare i pattern dei boss) può scoraggiare i giocatori meno avvezzi al genere, abituati a shooter più permissivi o carichi di continue e salvataggi. Allo stesso modo, l'esigenza di un hardware AGA decente o di un'emulazione ben configurata lo rende meno "plug and play" rispetto ad altri homebrew più leggeri, e la scelta dello schermo decentrato delle prime versioni può far storcere il naso ai puristi del layout centrale. Ma sono compromessi che nascono da precise scelte di design e di target. Wings of Fear punta dichiaratamente a chi vuole un'esperienza arcade solida, in linea con i grandi nomi del passato, e accetta volentieri che la soglia d'ingresso sia un po' più alta pur di non snaturare la propria identità.

di **Giampaolo Moraschi**





NEW GAME

BUBBLE BUDDIES

Anno: 2026

Editore/Sviluppatore: geon3

Genere: Platform

Piattaforma: Commodore 64

Sito web: <https://geon3.itch.io/buddies>

Di parlare di un gioco come Bubble Bobble non ci si stanca mai, anche se si rischia di essere ripetitivi e forse non è il caso, visto che continua a regalarci emozioni con la versione classica che viene giocata tutt'oggi dopo quasi quarant'anni.

Prodotta in tutte le salse per quasi tutti i computer e console, come se non bastasse, ai giorni nostri stanno uscendo altri titoli che non sono soltanto nuovi capitoli, bensì espansioni (se vogliamo chiamarle così) o rivisitazioni per creare nuovi livelli, nuove sfide e aggiungere giocabilità!

Dopo la versione Remastered e The Lost Cave, ecco Bubble Buddies, un altro titolo pronto ad entusiasmarci con altri nuovi cento livelli e curiosità dell'universo Bubble Bobble.

Il titolo propone la sfida a ben cento nuovi livelli tutti da esplorare e con miriadi di nemici, oltre ai vari gadget e frutti da cogliere, tutti bonus che ci aspettano in ogni livello.

Per renderlo ancora più interessante e ricco di sfide, i nemici non compaiono nei livelli più avanzati o in ordine di progressione del loro grado di minaccia, bensì quasi subito. Quindi i simpatici folletti spara-bottiglie sarà possibile incontrarli già dai primi schermi.

I livelli sono strutturati quasi tutti su semplici piattaforme, per fortuna non

inaccessibili, anzi perfettamente raggiungibili in single player, ma questo personalmente lo ritengo un po' un peccato, perché mi divertiva vedere e giocare i livelli fatti con le scritte anche se alcuni erano di una difficoltà inaudita.

La qualità grafica è rimasta quella della versione Remastered senza se e senza ma, e anche la colonna sonora (che forse poteva essere cambiata).

Giocandolo per la prima volta mi son detto: "Ma sì, faccio solo una decina di livelli, poi vado a nanna", invece mi sono ritrovato ad affrontare il boss finale del centesimo! E per favore, non cominciate a farvi domande su chi sia, piuttosto gustatevi con calma questo nuovo capolavoro, tanto il boss è sempre lì che vi aspetta pur sapendo che sarà sconfitto per la milionesima volta.

A disposizione ci saranno i soliti nove crediti, più che sufficienti direi per arrivare fino in fondo, senza contare i vari ombrelli che in questa versione pare siano piuttosto generosi... E non sono solo quelle le scorciatoie utili.

Sulla giocabilità probabilmente è inutile spendere altre parole perché è semplicemente la migliore di tutte le versioni!

di **Daniele Brahimi**



GIUDIZIO FINALE

» Giocabilità 95%

La migliore, senza se e senza ma.

» Longevità 90%

Anche dopo il centesimo livello, una voglia infinita, che dura una vita.





MIRACLE BOY IN DRAGON LAND

Anno: 2026

Editore/Sviluppatore: SamSoft

Genere: Avventura/Rpg

Piattaforma: Atari ST

Sito web: <https://samsoft.itch.io/miracle-boy-in-dragon-land>

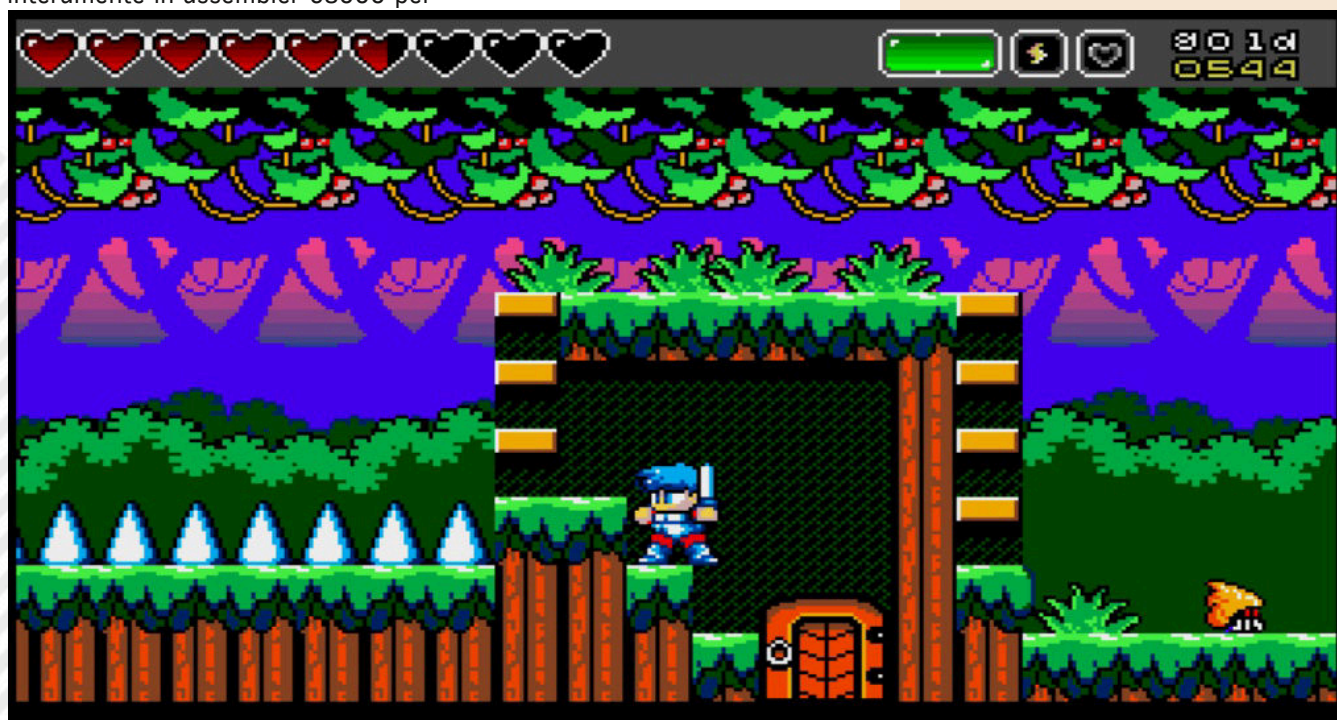
Miracle Boy In Dragon Land è pura magia ed emozione. E' un action-adventure/platform che nasce come grande lettera d'amore a Wonder Boy in Monster Land, ma si impone come uno dei progetti più ambiziosi e tecnicamente notevoli della scena moderna su questa macchina.

Il gioco è ambientato in Miracle Land, un regno fantasy sotto la minaccia del malvagio Conte Drago. Il protagonista è un giovane eroe impegnato a liberare il regno attraversando territori popolati da mostri, puzzle, negozi, NPC e mini-giochi. La struttura richiama quella degli action-RPG della serie originale con mappe collegate, boss ricorrenti, equipaggiamento migliorabile e un mondo che si comporta più da piccola avventura che da semplice sequenza di livelli lineari.

Tecnicamente, Miracle Boy è sviluppato interamente in assembler 68000 per



Atari 1040STf/STE, con uno scrolling fluido e prestazioni che lo rendono una vera e propria "vetrina tecnica" per la macchina. La grafica sfrutta bene la palette e la risoluzione dello ST, e l'audio DMA-SC sugli STE offre musiche e effetti considerati di altissimo livello per lo standard della piattaforma. Personalmente trovo eccezionale l'uso delle palette cromatiche, le animazioni, la varietà di personaggi inserita e l'incredibile





GIUDIZIO FINALE

» Giocabilità 95%

Bello, reattivo, dal grande gusto estetico e dall'ottimo design di livelli. E' il capitolo di Wonder Boy che mancava su ST.

» Longevità 95%

Il mix platform/avventura/Rpg è perfetto! C'è tanto da esplorare, da scoprire, da combattere e da vedere.



accompagnamento musicale.

Lo sviluppo è durato quasi quattro anni ed è stato completato il 5 aprile 2026; la versione finale è prevista per metà maggio 2026, con edizione fisica limitata già in pre-ordine.

Wonder Boy in Monster Land: la matrice arcade-avventurosa

Per capire l'anima di Miracle Boy In Dragon Land è fondamentale guardare al suo modello principale, Wonder Boy in Monster Land. Il gioco Sega del 1987 (arcade System 2, poi convertito su numerose piattaforme domestiche) rappresenta una transizione cruciale per la serie Wonder Boy. Dal platform puramente arcade del primo episodio a una formula ibrida che combina azione, platforming, elementi da RPG e struttura pseudo-open con città, dungeon e boss multipli. Un successo incredibile che si riconfermò nello splendido Wonder Boy Dragon's Trap per Master System.

In Wonder Boy in Monster Land il protagonista attraversa un regno suddiviso in aree collegate, con villaggi dove parlando con NPC si ottengono informazioni, negozi in cui acquistare armature, stivali, spade e magie, e dungeon in cui affrontare nemici sempre più impegnativi, raccogliendo monete e oggetti che sbloccano equipaggiamento migliore. Il tutto è filtrato attraverso una grammatica da sala giochi: tempo limitato, vite finite, boss di fine livello con pattern specifici, e un forte incentivo a imparare percorsi, segreti e punti di farming delle monete.

Miracle Boy In Dragon Land riprende proprio questa struttura ibrida. L'idea di muoversi in un regno condiviso, di incontrare NPC "colorati" che danno indizi o servizi, di entrare in edifici che fungono da negozi, locande o punti chiave della progressione, e di affrontare boss in punti specifici della mappa è direttamente debitrice di Wonder Boy in Monster Land. Il fatto che l'avventura sia strutturata in 19 livelli con 5 boss, che includa mini-giochi e puzzle, e che si fondi sull'upgrade dell'equipaggiamento, colloca Miracle Boy nella stessa famiglia di "action-adventure da sala" in cui Monster Land è un capostipite.

Anche la scelta di un'estetica colorata, fantasy e relativamente luminosa, più fiabesca che dark, rientra nell'eredità di Wonder Boy in Monster Land, che già all'epoca si differenziava da molti altri action-RPG per il suo tono visivo brillante e cartoonoso. Miracle Boy ne raccoglie l'eredità su Atari ST, machine che ai tempi non ha mai ricevuto una





conversione altrettanto curata di quel modello.

Wonder Boy: The Dragon's Trap e la rilettura moderna

L'altra grande influenza citata per Miracle Boy In Dragon Land è Wonder Boy: The Dragon's Trap, ovvero il remake moderno del classico Wonder Boy III: The Dragon's Trap (Master System / PC Engine), ripensato da Lizardcube e DotEmu nel 2017.

The Dragon's Trap prende un gioco già "metroidvania ante litteram" – con trasformazioni del protagonista, backtracking massiccio, abilità che sbloccano nuove aree e una struttura molto più libera del platform lineare – e lo reinterpreta per il pubblico contemporaneo con una cura estrema per presentazione, animazione e fluidità del controllo, senza però snaturare il design originale. Mantiene la mappa interconnessa, i boss iconici, le trasformazioni (drago, topo, piranha, leone, falco), ma modernizza l'interfaccia, rifinisce il feeling dei combattimenti e inserisce qualità-divita in linea con gli standard moderni. Miracle Boy In Dragon Land guarda a questo approccio più che al semplice modello arcade originale. Non si limita a "copiare" Wonder Boy in Monster Land, ma adotta la stessa filosofia di The Dragon's Trap: prendere una formula anni '80 e ripensarla per oggi, mantenendo il cuore meccanico e l'impianto di progressione, ma lavorando su fluidità, leggibilità, densità di contenuti e coerenza narrativa.

La scelta di sviluppare il gioco interamente in assembler 68000 per ottenere uno scrolling e una reattività al livello delle produzioni console, l'attenzione alla colonna sonora DMA, l'integrazione di puzzle, mini-giochi e momenti "lenti" tra una sezione d'azione e l'altra, e la cura nel worldbuilding di Miracle Land, vanno proprio in questa direzione: usare l'hardware Atari ST per costruire un'esperienza che, pur girando su un computer di metà anni '80, si avvicina come filosofia alle reinterpretazioni moderne dei classici,

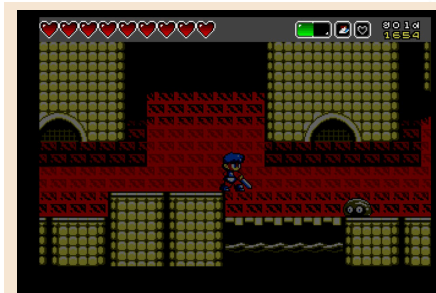
di cui The Dragon's Trap è un esempio di riferimento.

In questo senso, Wonder Boy in Monster Land fornisce a Miracle Boy la struttura di base (regno, villaggi, equip, boss, progression arcade-RPG), mentre Wonder Boy: The Dragon's Trap suggerisce il modo di farla sentire "viva" e coesa nel 2026: controlli stretti, ritmo ben dosato fra esplorazione e combattimento, attenzione ai dettagli estetici e alla musica, e una costruzione del mondo che invita il giocatore a considerare Miracle Land come un luogo da esplorare, non solo come una sequenza di stage da superare.

Conclusione

Inserito in questo contesto, Miracle Boy In Dragon Land si presenta non solo come un grande tributo a Wonder Boy in Monster Land, ma anche come un tentativo riuscito di applicare, su Atari ST, la stessa sensibilità che ha reso Wonder Boy: The Dragon's Trap un remake di riferimento. Ne eredita la miscela di platform, azione e progressione da RPG, l'attenzione all'atmosfera e al worldbuilding, e la volontà di far sembrare "nuova" una struttura nata negli anni '80, pur restando fedele al suo spirito arcade-avventuroso.

di Carlo Nithaiah Del Mar Pirazzini





NEW GAME

CURSE OF THE PHARAOH

Anno: 2026

Editore/Sviluppatore: Odin Strabo Jensen

Genere: Platform

Piattaforma: Commodore 64

Sito web: <https://furidk.itch.io/curse-of-the-pharao>

Dopo le uscite mensili che ormai sono una tappa fissa anche dopo quarant'anni dalla prima comparsa del Commodore 64, rieccoci di nuovo ad esplorare tombe egizie... E non solo.

Qui non si parlerà dell'ennesimo clone di Indiana Jones o Lara Croft, oppure Rick Dangerous, se vogliamo rimanere in tema Commodore, bensì di un nuovo simpatico platform ambientato in una tomba egizia piena di pericoli e maledizioni.

The Curse of the Pharaoh è un platform ambientato in una tomba egizia con dieci livelli ricchi di suspense, mummie e precisione millimetrica.

Lo scopo del gioco è quello di raccogliere una quantità stabilita di pietre dorate sparse per il livello e dirigersi verso l'uscita, compito non facile data la moltitudine dei nemici quali mummie, serpenti e chi più ne ha più ne metta, pronti a ostacolarci. Oltre a questi, si aggiungono altri pericoli come spuntoni e lance che spuntano dal pavimento. Tanto per non farsi mancare niente...

Dieci livelli sono pochi per i miei gusti e la difficoltà non è molto alta, tant'è che lo finirete alla quarta o quinta partita. Però l'idea e l'ambientazione sono davvero carine oltre a ricordare le imprese di archeologi su grande schermo e archeologhe su console.

Al decimo livello troveremo ad aspettarci il Faraone, o quel che ne

resta, a sorvegliare l'ultima stanza che ci porterà all'uscita e alla vittoria.

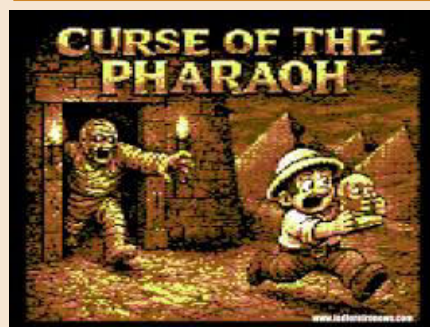
L'unica azione che possiamo compiere è il salto per raggiungere le piattaforme ed evitare i nemici. Stranamente non si muore se si cade da altezze elevate e questo è un punto a suo favore come anche la fluidità generale che mi ha piacevolmente sorpreso.

Penso che questo sia uno di quei titoli che, anche una volta terminato, ti lascia la voglia di rigiocarci, anche solo per fare una partitina veloce. Insomma, un sempreverde!

Spero che un giorno o l'altro facciano un nuovo capitolo (anche più di uno), magari ambientato in diverse locazioni in qualche altra parte del mondo, perché, seppur breve, l'ho trovato simpatico, giocabile, carino e ricco di spunti che solleticano i ricordi.

Siete pronti a visitare e fuggire dalle tombe egizie direttamente dalla sedia di casa vostra?

di **Daniele Brahimi**



GIUDIZIO FINALE



» Giocabilità 80%

Precisione millimetrica nei salti, per il resto ci siamo.

» Longevità 60%

Carino, ma dieci livelli sono pochi e si finisce subito.





AXIS

Anno: 2026**Editore/Sviluppatore:** Anystone Games**Genere:** Action/Puzzle**Piattaforma:** Commodore 64/
Commodore Plus/4**Sito web:** <https://anystone.games/>

Dimenticate Doom, Duke Nukem o qualsiasi gioco in cui potete effettivamente difendervi. In Axis, il vostro protagonista è un astronauta che ha chiaramente dimenticato il kit di sopravvivenza a casa e ha deciso che l'unico modo per gestire un'invasione aliena è... correre!

Il concetto è di una semplicità che rasenta il bullismo psicologico. Non hai armi. Non hai poteri. Hai solo le tue gambe e una speranza di vita paragonabile a quella di un cubetto di ghiaccio in un vulcano.

Devi trovare una chiave e raggiungere l'uscita senza farti sfiorare dai residenti locali (gli alieni), che sono molto territoriali.

Il tutto in 10 settori pieni di nemici che sembrano programmati per leggarti nel pensiero. Se li tocchi, non perdi energia... esplodi! O muori di spavento! In ogni caso, ricominci da capo.

Il gioco non perdona, non dimentica e probabilmente ride di te sotto i baffi (se i chip avessero i baffi...). Passerai il 90% del tempo a fissare un alieno che blocca un corridoio stretto, aspettando il momento giusto per passare, solo per realizzare che avevi calcolato tutto tranne il tuo riflesso da bradipo morto da tempo.





GIUDIZIO FINALE

» Giocabilità 82%

I comandi sono così precisi che non potrai dare la colpa al joystick per i tuoi fallimenti. L'astronauta si muove con una reattività chirurgica, il che è fantastico finché non realizzi che sei stato tu, con le tue mani, a lanciarlo contro quell'alieno. È una danza millimetrica tra la gloria e l'esplosione spontanea.

» Longevità 65%

Dieci settori che durano quanto la tua pazienza dopo il quinto Game Over consecutivo. Una volta memorizzati i percorsi, lo finisci nel tempo di un caffè. È perfetto per chi ha i minuti contati o per chi vuole sentirsi un fenomeno del gaming prima che la realtà (o la cena) bussi alla porta.

Gli ambienti sono così puliti e geometrici che sembrano progettati da un architetto svedese con l'ossessione per l'ordine. La definizione è talmente alta che potrai goderti ogni singolo pixel del tuo astronauta mentre viene vaporizzato. È il trionfo del "less is more": meno fronzoli ci sono, più spazio resta.

Non ci sono armi, non ci sono scuse... ma ci sei solo tu e la tua imbarazzante capacità di andare a sbattere contro l'unico alieno nel raggio di tre chilometri.

di Chiara "Kika" Carrara

Il chip SID sputa fuori una colonna sonora che è essenzialmente ansia pura tradotta in onde sonore. Ha quel ritmo incalzante che ti fa sentire costantemente inseguito, trasformando la tua stanza in una base spaziale claustrofobica.

Axis è la dimostrazione che nel 2026 abbiamo ancora bisogno di giochi che ci ricordino quanto siamo scarsi. È un titolo onesto, veloce e frustrante quanto basta per farti sentire un genio se riesci a finirlo senza lanciare il computer dalla finestra.

il gioco perfetto per chi ha poco tempo e molta voglia di soffrire. È un porting pulito, spietato e così essenziale che quasi ti senti in colpa a non riuscire a finirlo al primo colpo.





GRAVITY DUCK

Anno: 2026

Editore/Sviluppatore: WPQREK

Genere: Platform

Piattaforma: Commodore 64

Sito web: <https://wpqrek.itch.io/gravity-duck-64>

Se pensavate che l'unica cosa in grado di sfidare le leggi della fisica sul Commodore 64 fosse la capacità dell'alimentatore di non esplodere dopo due ore di gioco, preparatevi a ricredervi. Gravity Duck 64 ci ricorda che, nel 2026, il modo migliore per divertirsi è far finta di essere nel 1984, ma con un papero che ha seri problemi di equilibrio.

Gravity Duck 64 è farina del sacco di un trio di sviluppatori della scena homebrew polacca, noti come wpqrek. Il protagonista è un papero giallo che ha chiaramente saltato il LEGDAY in palestra... perché non sa saltare! In compenso, possiede il potere divino di invertire la gravità...ma effettivamente perché saltare gli ostacoli quando puoi letteralmente prendere il soffitto a testate? Con la pressione di un tasto, inverti l'asse del mondo. È come cercare di arredare un appartamento mentre qualcuno scuote la stanza come un cocktail. Devi recuperare un uovo d'oro attraverso 40 livelli di puro delirio architettonico.

Perché? Beh probabilmente il protagonista soffre di una rara forma di feticismo per i metalli pesanti che lo spinge a ignorare le leggi della fisica pur di riempirsi il nido di lingotti a forma di uovo. In ogni caso, non farti domande: corri, cadi verso l'alto e recupera quel bel ovetto anche se la Pasqua è ormai passata.

Gli Sprite sono così nitidi che sembrano





GIUDIZIO FINALE

» Giocabilità 85%

Il controllo è così reattivo che il papero esegue i tuoi ordini prima ancora che il tuo cervello li formuli. È fluido, preciso e senza fronzoli. L'unico vero limite è la coordinazione tra il tuo pollice e il tuo senso dell'equilibrio, che dopo dieci minuti inizieranno a chiederti il divorzio.

» Longevità 70%

Con 40 livelli, lo finisci nel tempo che serve al tuo vicino per parcheggiare l'auto. È un'esperienza fulminea! Una volta raccolte tutte le uova, l'unico motivo per rifarlo è voler dimostrare a te stesso che non è stata solo fortuna. Ottimo per una sfida veloce, meno se speravi di occuparci l'intero weekend.

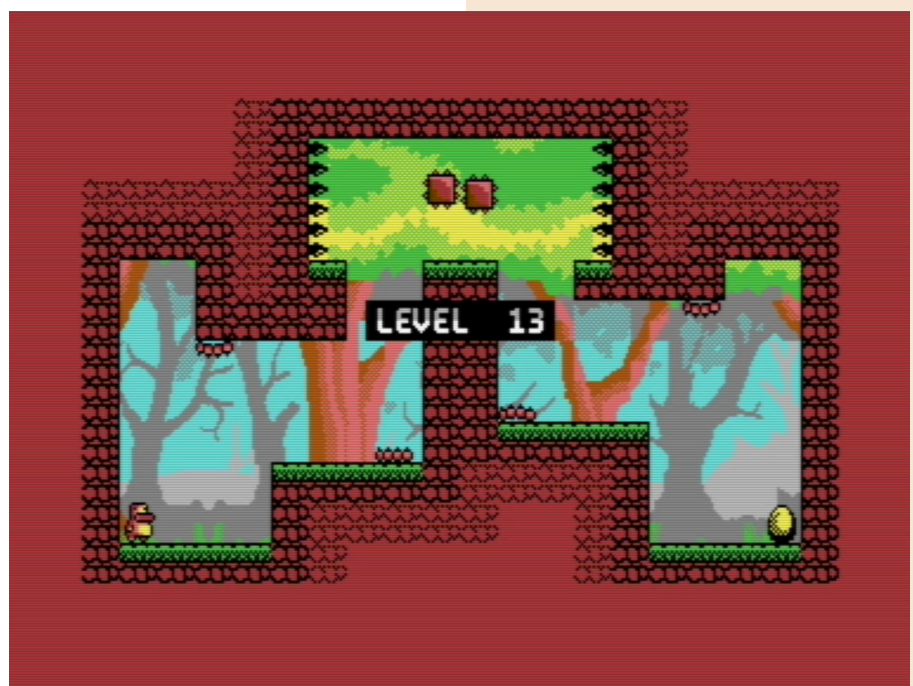
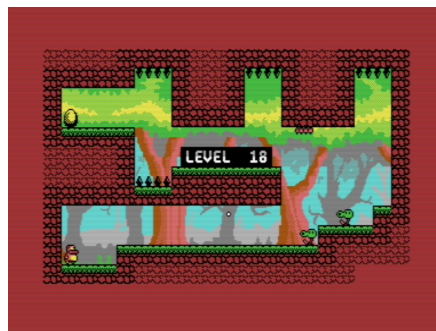
fatti al laser. È tutto così fluido che rischi di scordarti che il tuo computer ha meno memoria di quella che serve al tuo spazzolino elettrico per accendersi.

Il chip SID sfoggia una colonna sonora ritmata e super orecchiabile, di quelle che ti entrano in testa e non ti abbandonano più. È un motivetto ipnotico che trasforma ogni livello in una danza gravitazionale, capace di farti fischiettare per ore anche dopo aver spento il monitor. Un piccolo capolavoro di sintesi sonora che dà al gioco quel tocco magico tipico delle grandi produzioni per C64.

Da lodare il sistema di controllo e la precisione dei comandi mentre segnalo che occorre un po' di pratica per gestire il sistema di gravità in gioco.

Ad ogni modo Gravity duck è un piccolo tesoro di cortesia e precisione. È il gioco ideale per coccolare il tuo Commodore e ricordarti che, ogni tanto, è bello guardare il mondo da un'altra prospettiva, anche solo per il tempo di un battito d'ali. E come dico sempre vale la pena di provarlo!

di Chiara "Kika" Carrara



Refurbished software & revisited games

Il retrocomputing nel 2026 è una vera e propria piattaforma culturale e tecnica, una realtà che collega il passato al futuro. In un'era di rapida obsolescenza e smaterializzazione del software, recuperare sistemi e programmi di 30 o 40 anni fa ha un valore che va oltre il semplice collezionismo. Si tratta di un atto di conservazione storica e, allo stesso tempo, di un laboratorio di innovazione. Preservare il software del passato significa salvare un patrimonio di idee. Agli inizi dell'informatica "domestica", gli sviluppatori e i programmatori lavoravano con pochi kilobyte di memoria, processori limitati e strumenti lenti e fragili. Eppure, da questi limiti sono nate soluzioni eleganti, algoritmi ingegnosi e tecniche di programmazione raffinate. Studiare quel codice è come leggere i classici della letteratura: si possono scoprire principi fondamentali che restano validi ancora oggi. Tuttavia, preservare gli oggetti digitali non è un processo semplice. I programmi originali spesso si trovano solo su supporti deteriorati e in forma cartacea. Passarli alla forma digitale significa renderli accessibili e disponibili, permettendo a chiunque di eseguirli su emulatori o hardware reale. È un lavoro paziente, potremmo dire simile a un lavoro archeologico, e richiede competenze tecniche e sensibilità storica.

Ma il retrocomputing non si limita alla conservazione. Negli ultimi anni, si è verificato un fenomeno sorprendente: la creazione di nuovi giochi e software per piattaforme obsolete. I programmatori ora utilizzano strumenti moderni per spingere quei sistemi oltre i loro limiti originali. Il risultato è una produzione che non solo eguaglia, ma talvolta supera quella dei "tempi d'oro". Questo dialogo tra passato e presente apre a scenari interessanti. Si cerca di rispettare l'estetica e le logiche dell'epoca, introducendo elementi innovativi che dimostrano quanto potenziale fosse ancora inespresso in quelle macchine. Il retrocomputing diventa così uno spazio di sperimentazione, dove i vincoli rappresentano una sfida stimolante. Un aspetto affascinante di questi processi è il "reverse engineering". Quando i codici sorgente originali non sono disponibili, lavorare su un programma significa analizzarne il funzionamento a partire dal codice binario. È un processo complesso che richiede competenze avanzate e pazienza, ma che può portare a risultati straordinari. In questo senso, la preservazione non è più un atto passivo, ma diventa dinamico. Non si tratta solo di salvare ciò che è stato pubblicato, ma anche di reinterpretarlo, modificarlo e, perché no?, migliorarlo. Una forma di continuità tecnologica che rompe la linearità del tempo e mostra come il passato possa ancora dialogare con il presente.

Naturalmente esistono questioni etiche e legali da affrontare, come la proprietà del software e i diritti degli autori. Tuttavia, è difficile negare il valore culturale di queste attività, che aiutano a mantenere viva una parte fondamentale della storia dell'informatica. La preservazione digitale (come ben dimostra il nostro progetto RetroLiPS) rappresenta un ponte tra generazioni di tecnologie e programmatori. È memoria e allo stesso tempo reinvenzione. In un'epoca che tende a dimenticare rapidamente, questa pratica ci ricorda che anche nei bit più datati può nascondersi un'innovazione in grado di sorprendere.

David La Monaca

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.

