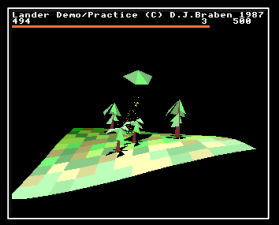
ROUGOL Talk August 2025

Mark Moxon

Contents



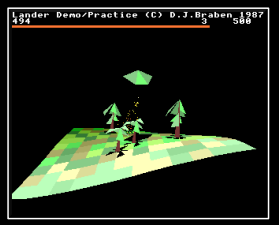
- Adding Econet support to Archimedes Elite
  - A quick overview of Econet Elite on the BBC/Electron
  - Detail about adding Econet support to Archimedes Elite
- Lander disassembly project
  - Landscape generation
  - Origins on the ARM1 and BBC Micro Elite II
  - BigLander
- Elite backporting projects
  - Enhanced sideways RAM version of Electron Elite
  - BBC Master Elite on the BBC Micro B+
- Software archaeology: The Elite source code family tree
  - The development history of 6502 Elite
  - Commodore 64, Apple II and NES source projects

Adding Econet support to BBC Elite

Adding Econet support to BBC Elite

- Fix memory clashes between Elite and Econet
  - All machines: 16 zero page locations from &90 to &9F reserved for Econet
  - On the Master, pages &B and &C are reserved for Econet
  - BBC Micro: Econet steals so much memory that there simply isn't room
    - Sideways RAM is available, otherwise drop docking animation and planetary detailing
- The BBC Master version needs to deal with NMIs from Econet
- Interface for loading and saving commander files needs update
- Loader programs
  - *Elite detects the system, checks for EliteConf, loads the correct Elite
  - *EliteB <param> allows loading of docked, flight, ship files

The multiplayer scoreboard



- Multiplayer like golf or darts, not tennis (it is not Arena Combat)

<menu> ELITE Stn 1.101
<sort SCOREBOARD Port 100
↔Page Page 1/2

	Mc	Station	C	Lgl	Player	Kills	Credits
	B	138.054	■	Fug	RIMMER	25/6	1226.3
	B+	4.200	■	Fug	HAN	24/10	1033.7
	M	42.056	■	Cln	MARVIN	23/11	957.5
	M	4.234	■	Cln	C3PO	21/3	1298.7
	SP	138.115	■	Off	LISTER	21/6	957.2
*	B	5.062	■	Fug	SPOCK	17/1	1053.6
	B	42.167	■	Cln	ZAPHOD	17/5	1189.3
	B+	42.011	■	Cln	FORD	17/5	900.3
	SP	1.070	■	Fug	ALEX	17/6	1056.9
	SP	1.245	■	Off	JAMESON	17/7	1273.6
	B	5.021	■	Off	KIRK	16/6	736.6
	M	1.187	■	Cln	RAXXLA	16/8	631.9
	B	1.214	■	Cln	RAFE	16/9	1019.5
	M	1.195	■	Off	ELYSSIA	15/4	982.9
	B+	42.128	■	Fug	ARTHUR	15/4	932.8
	SP	4.166	■	Off	R2D2	14/6	472.7
	M	4.244	■	Cln	LEIA	14/6	979.1
	B	5.140	■	Off	PICARD	14/8	750.3
	A	4.188	■	Cln	YODA	13/7	754.0
	SP	124.006	■	Cln	DECKARD	13/7	762.4

<menu> ELITE Stn 1.101
<sort SCOREBOARD Port 100
↔Page Page 2/2

Mc	Station	C	Lgl	Player	Kills	Credits
SP	66.099	■	Fug	RIPLEY	13/8	699.1
B+	2.001	■	Off	HAL	12/3	936.9
SP	7.111	■	Cln	ORAC	12/10	1081.2
SP	119.034	■	Off	FLASH	10/2	940.9
B	4.211	■	Cln	LUKE	7/4	536.6

The multiplayer scoreboard

- Data transmission, 20-byte block
  - #0-7: Commander's name, terminated by a carriage return
  - #8: Commander's legal status: 0 = clean, 1 = offender, 2 = fugitive
  - #9: Commander's status condition: 0 = docked, 1 = green, 2 = yellow, 3 = red
  - #10, #19: Commander's kill count (low byte, high byte)
  - #11: Commander's death count
  - #12-15: Commander's credits (low byte first)
  - #16: Machine: 0 = BBC+SRAM, 1 = Master, 2 = 6502SP, 3 = BBC, 4=Arc, 5=Elk
  - #17, #18: Commander’s station/network number for forwarded packets
- Send packet but do not wait for acknowledgement

Adding Econet support to Arc Elite

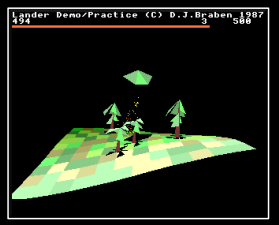
Adding Econet support to Arc Elite

- Arc Elite already loads over Econet (slowly, but it works)
- Adding support for the multiplayer scoreboard
  - Need to find and send the required data
- Disassembling Arc Elite is possible, but:
  - It is partly written in C, partly in assembler
  - It’s a mess of tiny getter/setter functions
- We need a better way

Relocatable module

- Relocatable modules are always resident in memory (in the RMA)
- Write a module that observes the game, which always runs at &8000
- Find the relevant memory locations
  - Dump out application memory in Arculator and look for strings
  - Disassemble existing cheat modules (credits, commander data block)
  - Examine code for commander editor applications
- What about kills and deaths?
  - We want one point per kill, but Arc Elite doesn’t work like that
  - Deaths are not counted anywhere

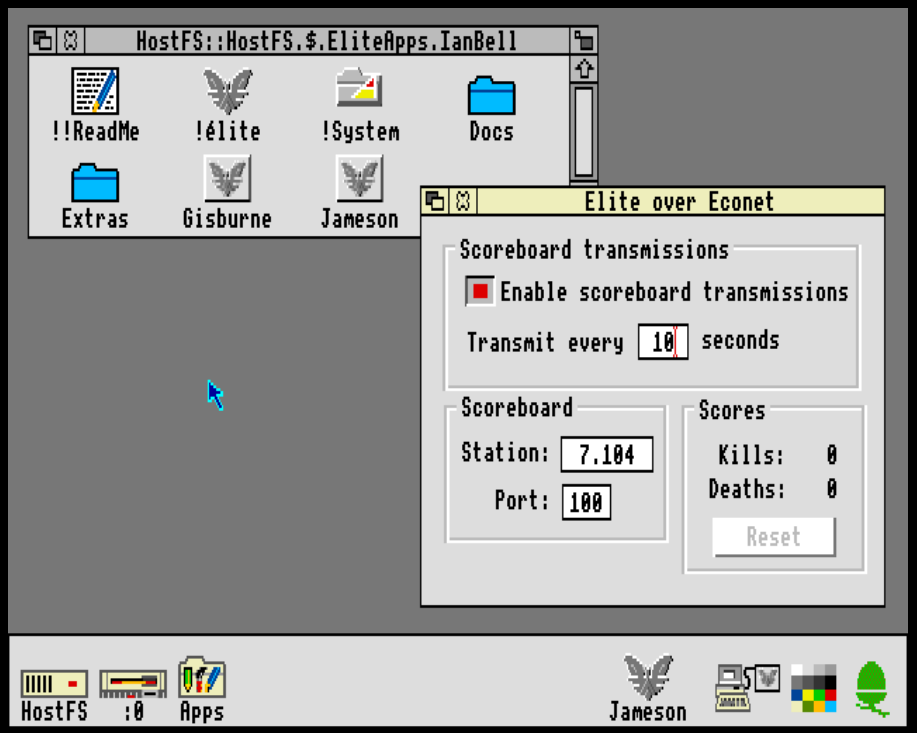
Module structure



- Build a module
  - Contains an observer that tracks rank and counts a kill when it goes up
  - Contains a routine that fetches data from the game and transmits it
- Call the module regularly using OS_CallEvery
  - Call the rank observer twice each second
  - Call the transmit routine every 10 seconds (configurable)
- On initialisation of the module, inject SWI calls into the title sequence
  - Insert SWI Elite_GameOver into the code that prints the “GAME OVER” text
  - Insert SWI Elite_GameRestart into the code that prints the title copyright text
  - First one is called every frame and sets a flag, second one records a death

Build the EliteNet application

- Add module configuration via *-commands:
  - *EliteStatus, *EliteTxStation
  - *EliteTxStart, *EliteTxStop
- Add a SWI interface:
  - Elite_GetStatus
  - Elite_SetStatus
- BASIC application using my own WimpLib (RISC User)
- Add configuration saving

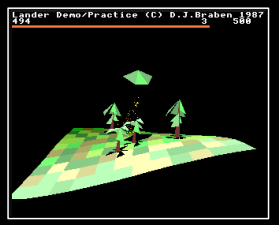


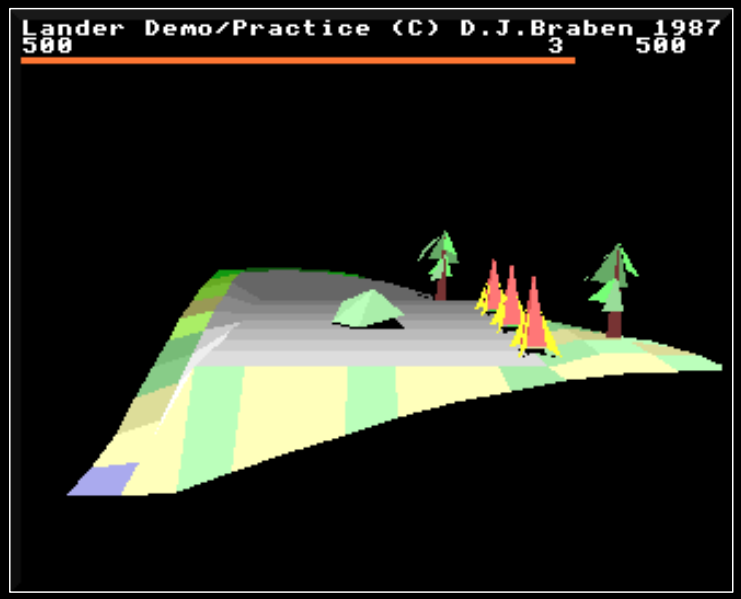
Lander disassembly project

What is Lander?

- What David Braben did next, after co-writing Elite
- 3D scrolling landscape shooter with beautiful solid 3D graphics
- First-ever game for the ARM platform
- Written in “about 2 months” late 1986/early 1987
- Started on an ARM Evaluation System and finished on a prototype A500
- Source appears to be in BBC BASIC assembler
- Bundled free with Arthur and RISC OS 2
- Works on RISC OS 3 but not on 3.5+
- Led to Zarch: smooth edges, more 3D objects, radar, enemies, virus, demo
- Lander-only: falling rocks at 800+, hover mode, launchpad is lethal

Lander vs Zarch



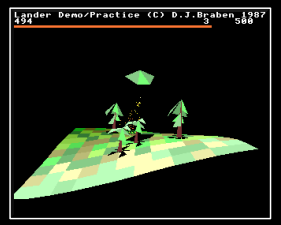


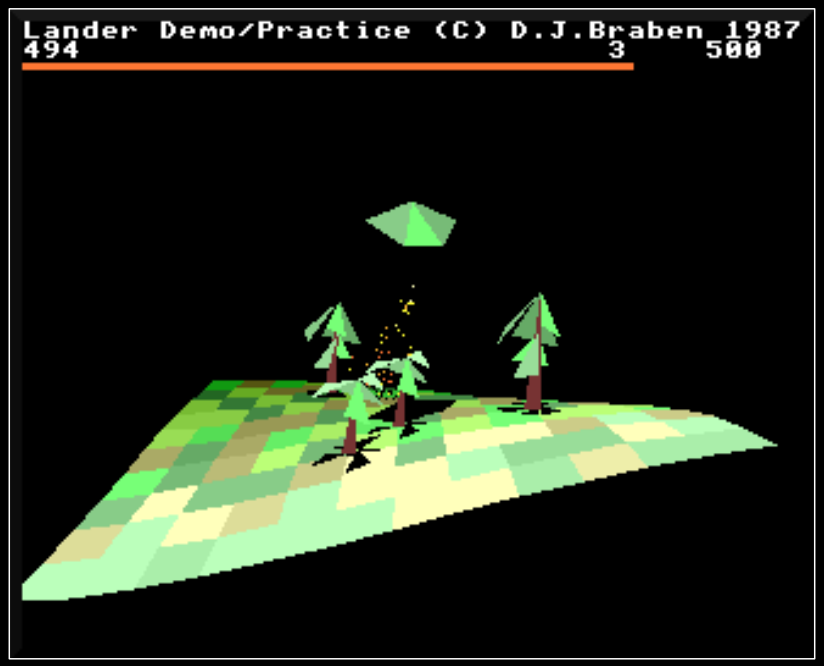
Documenting Lander

- Use Arthur binary as it is not encrypted (but is identical to RISC OS)
- Disassemble binary using Gerph's Python script
- Hand-build a pipeline using vasm and Python to support RO format
- Added labels and documentation using a text editor
- Deploy to Archimedes Live on Mac, or Arculator for deeper analysis
- Auto-generate website from GitHub repository
- Write 20+ deep dives – see lander.bbcelite.com
- Add script to produce BBC BASIC text file for building on RISC OS

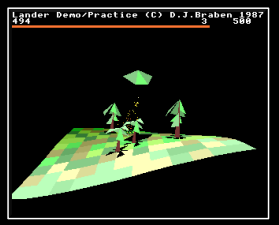
The procedural 3D landscape

Landscape 12x10 tiles

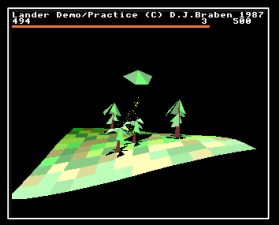




Big Lander 121x121 tiles



Memory is tight

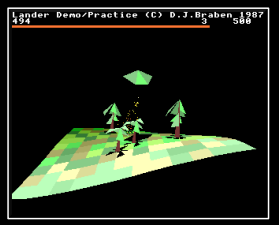


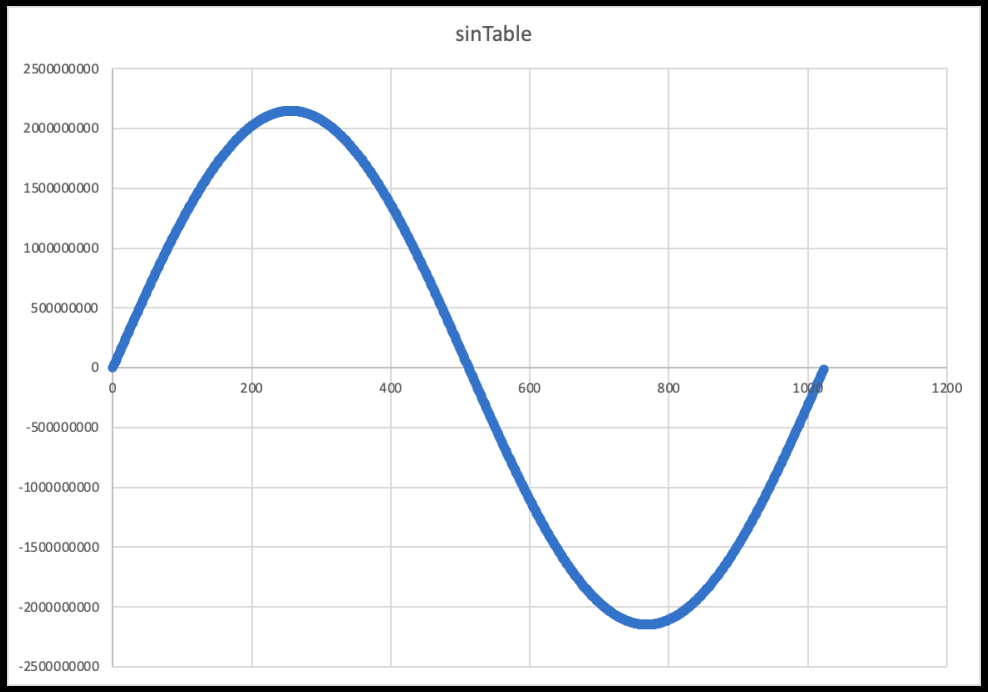
- Lander had to run on on the Archimedes A305, with 512K of RAM
  - Screen memory = 160K (2 x 80K)
  - System workspace = first 32K of memory
  - System heap & supervisor stack = 32K minimum
  - Cursor/system space/sound DMA = 32K
  - Lander requires 172K for game code and variable storage
  - Leaves 84K for relocatable modules, fonts, RAM disc, sprites
- To fit it into an A305, need to run USEGAME first:
  - Unplugs the modules containing the font and window managers
  - Disables the desktop

Procedural generation

- 256x256 tiles would take 65,536 values
  - 64K with 1-byte altitudes
  - 256K with 32-bit altitudes
- Lander implements an infinite landscape with 32-bit altitudes
- GetLandscapeAltitude routine = 46 instructions = 184 bytes
- Sine lookup table = 1024 x 32-bit words = 4096 bytes
- Total of 4280 bytes
- Only uses lookup tables, shifts and addition

Sum of sine waves (Fourier synthesis)





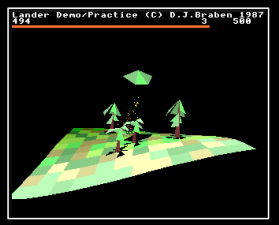
Sum of sine waves

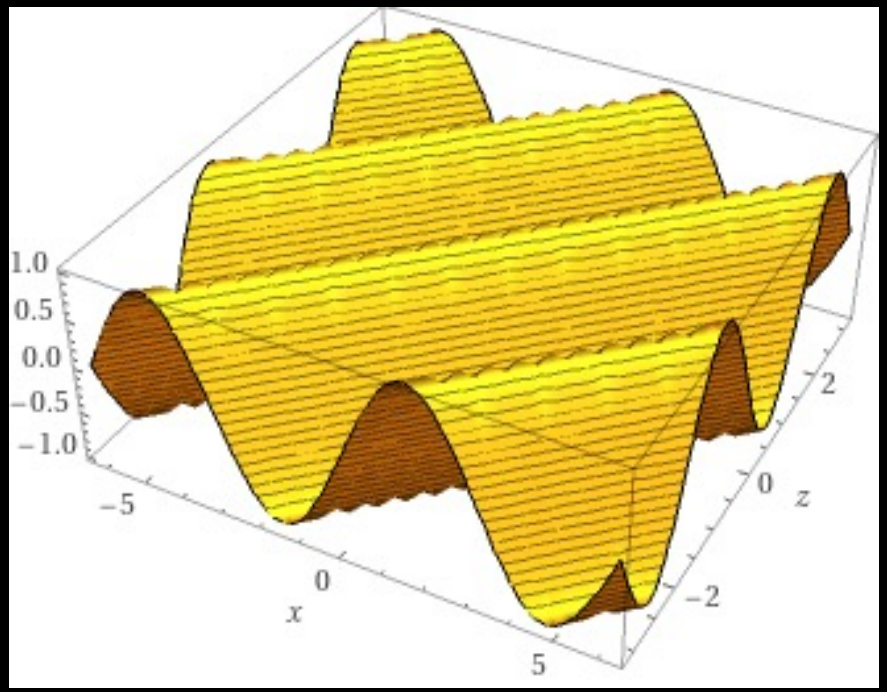
The altitude y of the point (x, z) on the landscape is given by:

```
LAND_MID_HEIGHT - (   2*sin(x - 2z)   + 2*sin(4x + 3z)
                    + 2*sin(3z - 5x)  + 2*sin(3x + 3z)
                    +   sin(5x + 11z) +   sin(10x + 7z)
                  ) / 256
```

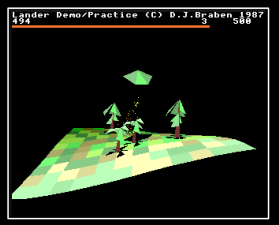
where LAND_MID_HEIGHT represents the altitude of the vertical mid-point of the landscape

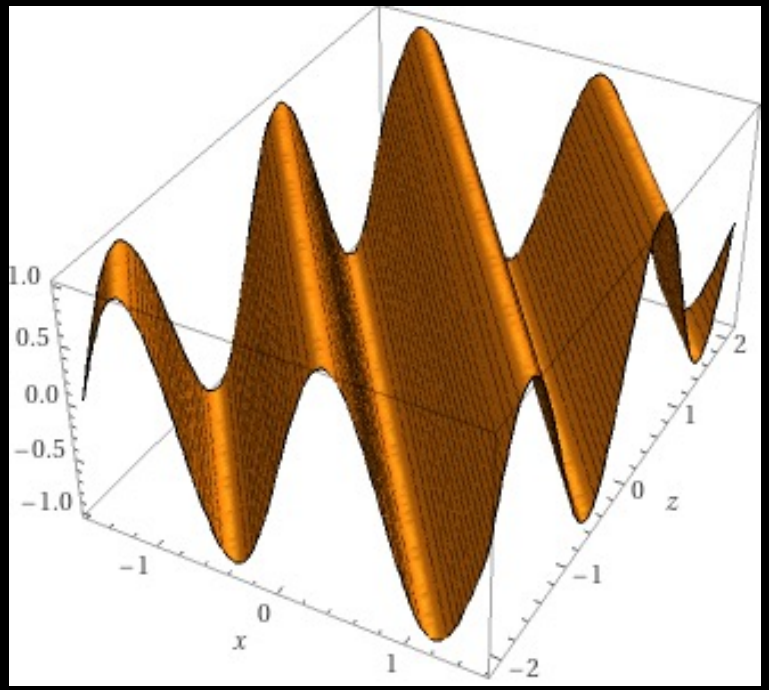
y = sin(x - 2z)


Lander Demo/Practice (C) D.J.Braben 1987
434
3
508

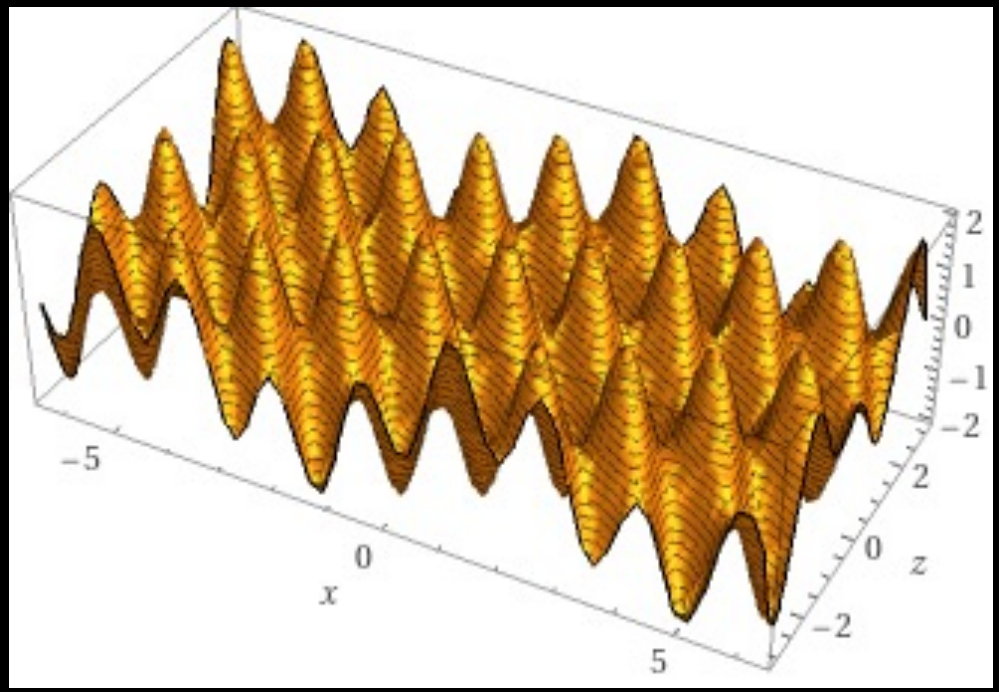

1.0
0.5
0.0
−0.5
−1.0
−5
0
5
x
2
0
−2
z

$y = \sin(4x + 3z)$

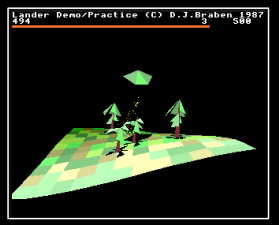

Lander Demo/Practice (C) D.J.Braben 1987
434
3
588

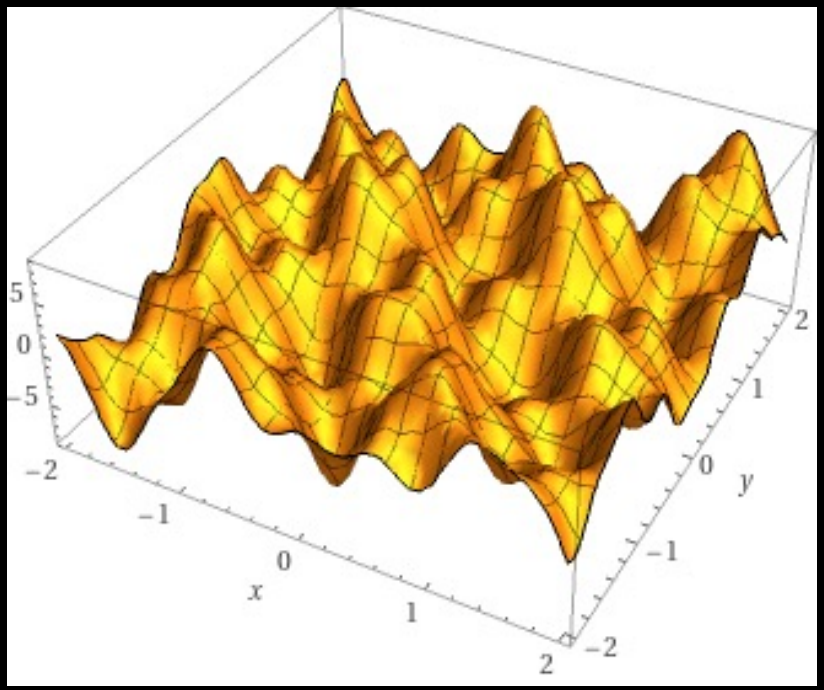

1.0
0.5
0.0
−0.5
−1.0
−1
0
1
x
2
1
0
z
−1
−2

Sum of two sine waves



Sum of six sine waves

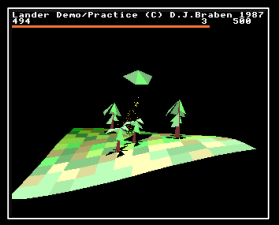


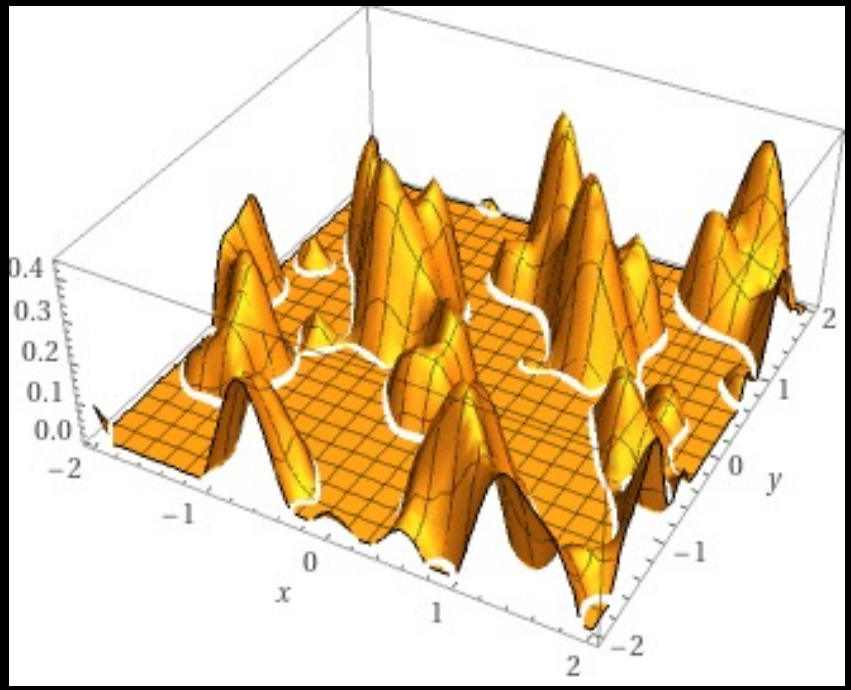


Add water and launchpad

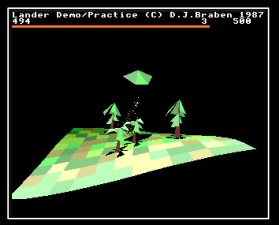
- Altitude is capped to a minimum value of SEA_LEVEL
- Altitude on the launchpad is set to LAUNCHPAD_ALTITUDE
- Launchpad is defined as (x, z) where 0 <= x < 8 and 0 <= z < 8
- So that's tiles 0 to 7 along each axis (8x8 square)
- Origin is at the front-left corner of the launchpad

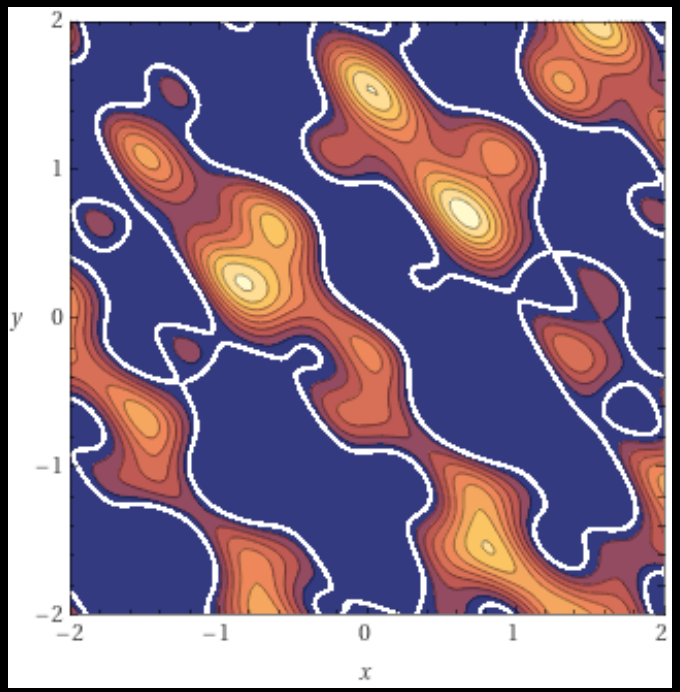
Add water


Lander Demo/Practice (C) D.J.Braben 1987
434
3
508

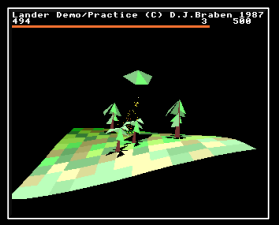

0.4
0.3
0.2
0.1
0.0
−2
−1
0
1
2
x
2
1
0
−1
−2
y

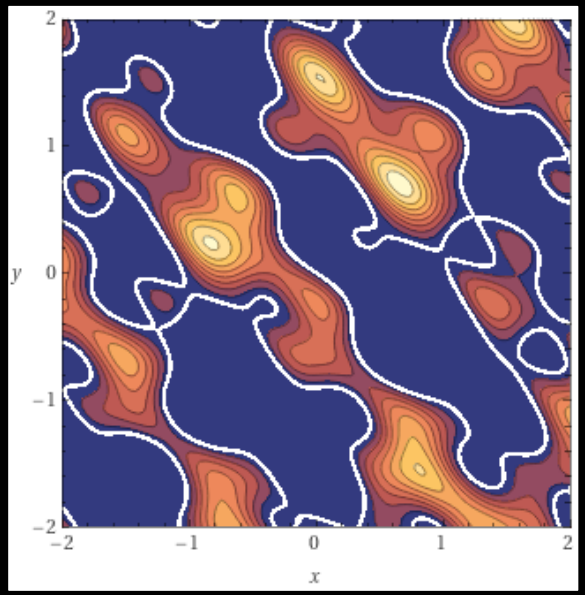
Contour view

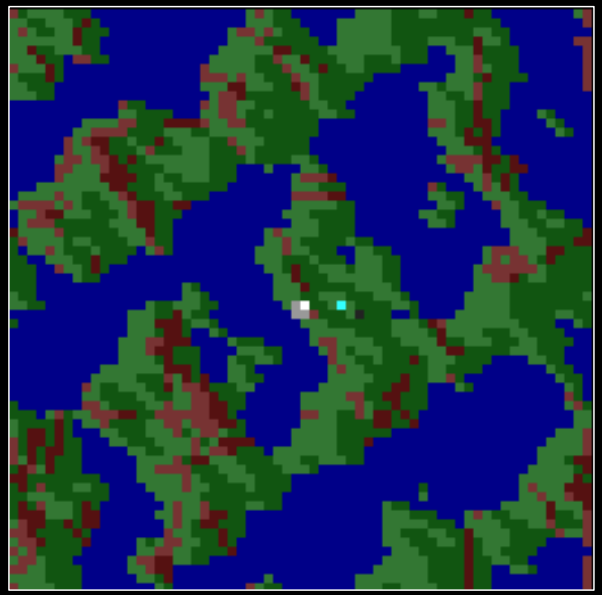




Compare with Zarch’s contour map







Origins on the ARM1 and Elite II

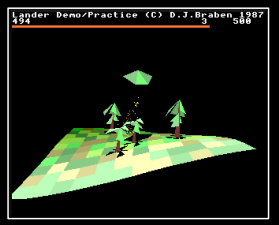
ARM1 influences

- Lander was started on the 1986 ARM1-based ARM Evaluation System
- Only moved to an ARM2 when Braben was lent an A500 prototype
  - “It didn't even have a disk drive, that's how basic it was”
- ARM1 does not have the multiply instructions (MUL, MLA)
- There are no MUL/MLA instructions in Lander (or Zarch)
- Inline shift-and-add algorithm instead (inserted via macro?)
- No ADR directive, Lander uses vectors instead
- No SBC or RSC instructions, or ROR shifter
- Whole game uses just 21 different instructions – truly RISC
- Only game ever written for the ARM1?

ARM1 instructions in Lander

- ADD ADC
- SUB RSB
- MOV MVN
- AND ORR EOR BIC
- CMP CMN
- TST TEQ
- LDR STR
- LDM STM
- B BL SWI

Elite II



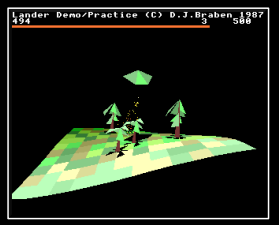
- Aborted 1985 sequel to the original BBC Micro Elite
- For the 40th anniversary of the release of Elite in Sept 2024, Ian Bell released 8 discs containing the sources for Elite II
- On disc 2015-08.048 ELITE II
  In file ELITE2D
  There is this line:

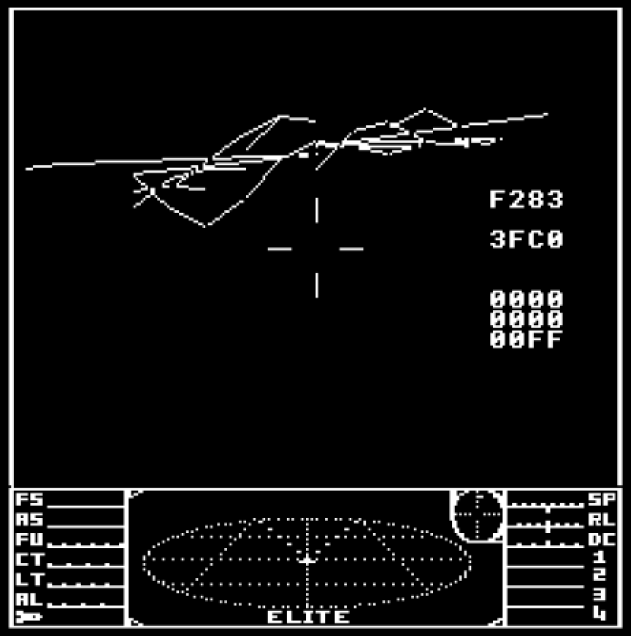
  `8255JSRMAS5:\JSRSCAPE:\RTS`
- And in file ELITE2E
  There is a subroutine called .SCAPE
  That contains a familiar bit of code...

Elite II: disc 2015-08.048, file ELITE2E

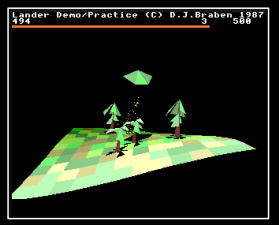

```
5000.LFN LDA#0:STAR:STAS     \ SR = Lscape(K1) (2b twos)
5005  LDAK1+6:LSRA:LDAK1+3:RORA:STAK1p3:LDAK1+1:RORA:STAK1p0
5006  LDAK1+8:LSRA:LDAK1+5:RORA:STAK1p5:LDAK1+2:RORA:STAK1p2
5010  LDAK1p0:ASLA:STAP:LDAK1p3:ROLA:STAII:ASLP:ROLA:STAII+1:ASLP:ROLA:STAII+2:ASLP:ROLA:STAII+3
5020  LDAK1p2:ASLA:STAP:LDAK1p5:ROLA:STAJJ:ASLP:ROLA:STAJJ+1:ASLP:ROLA:STAJJ+2:ASLP:ROLA:STAJJ+3
5030
5040  LDAK1p3                                :LDX#4:JSRLFN1  \16sinx
5050  LDAK1p5                                :LDX#4:JSRLFN1  \16sinz
5060  LDAK1p3:CLC:ADCK1p5:STAP:ASLA:CLC:ADCP:LDX#3:JSRLFN1  \8sin3(x+z)
5070  LDAII:CLC:ADCK1p3:CLC:ADCK1p5          :LDX#3:JSRLFN1  \8sin(3x+z)
5080  LDAK1p5:CLC:ADCJJ:SEC:SBCK1p3          :LDX#3:JSRLFN1  \8sin(3z-x)
5090  LDAII+1:SEC:SBCJJ                      :LDX#2:JSRLFN1  \4sin2(2x-z)
5100  LDAII+1:CLC:ADCJJ+1                    :LDX#2:JSRLFN1  \4sin4(x+z)
5110  LDAII+2:SEC:SBCJJ+1                    :LDX#2:JSRLFN1  \4sin4(2z-x)
5120  LDAII+2:CLC:ADCII:SEC:SBCJJ+2          :LDX#1:JSRLFN1  \2sin(10x-8z)
5130  LDAII+3:CLC:ADCJJ+3:CLC:ADCJJ+2        :LDX#0:JSRLFN1  \1sin8(2x+3z)
5140  LDAII+3:SEC:SBCJJ+3                    :LDX#1:JSRLFN1  \2sin16(x-z)
5150  RTS
5160
5170.LFN1 LDY#0:STYP \STZ++:JSRsine:SEC:BPLLFN2:EOR#FF:ADC#0
```

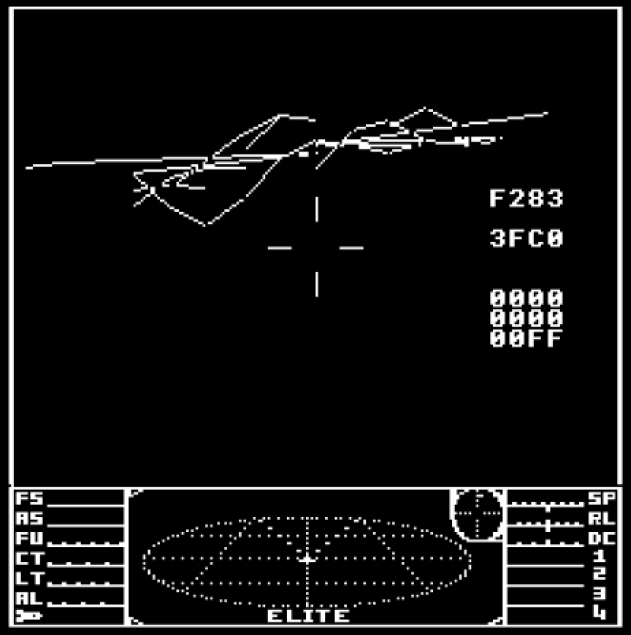
Elite II

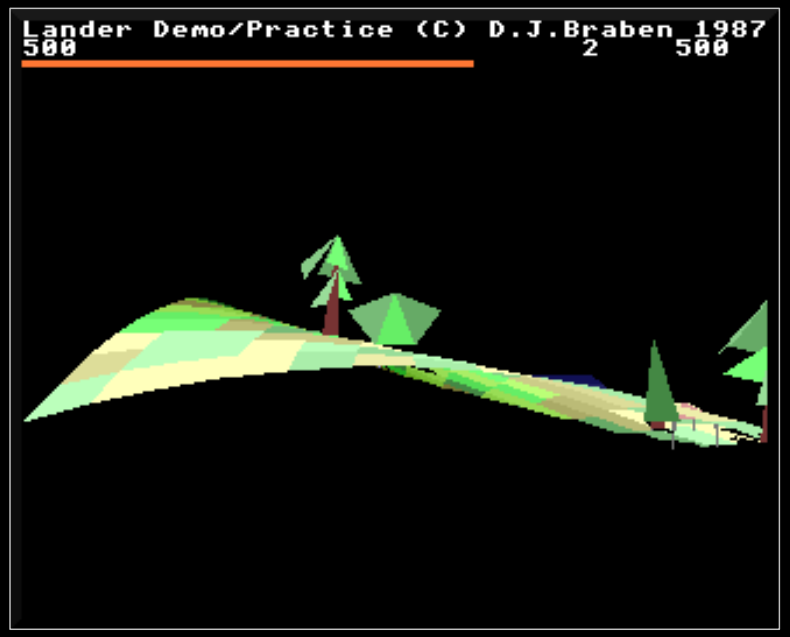

Lander Demo/Practice (C) D.J.Braben 1987
434
3
588


F283
3FC0
0000
0000
00FF
FS
AS
FU
CT
LT
AL
SP
RL
DC
1
2
3
4
ELITE

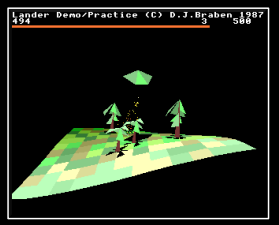
Elite II vs Lander



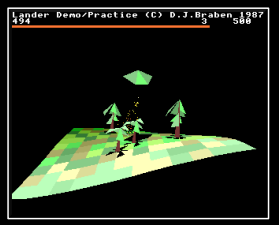




Big Lander



Micro User November 1987

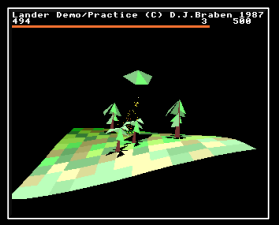


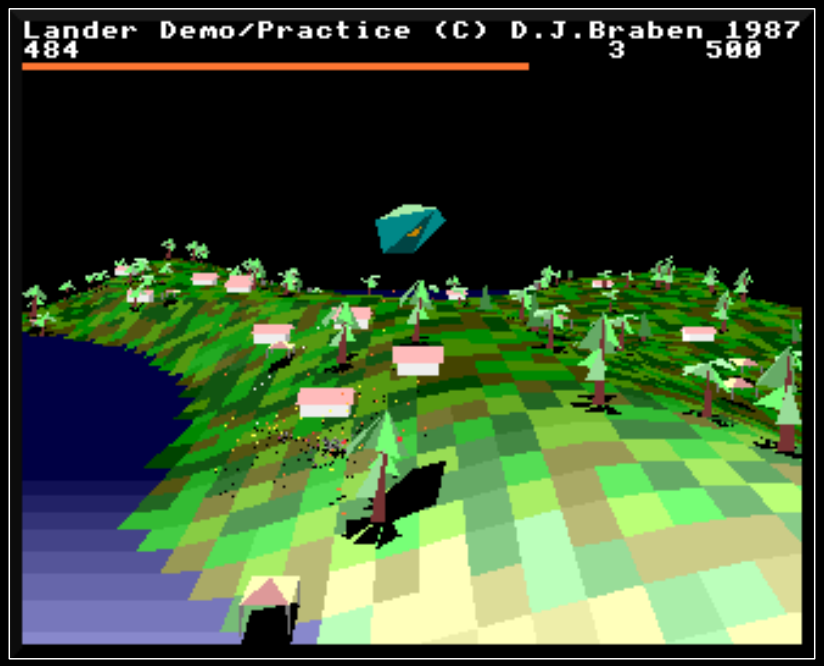
How Zarch might have looked: A display technique that was rejected

“With the display technique chosen, the frame rate dictated the approximate number of tiles which could be used and also the amount of scenery.

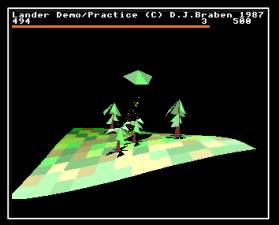
“The display looks much better if these are greatly increased - as the picture on this page shows - but sadly this reduces the frame rate to only one or two frames per second - a succession of stills.”

Big Lander 64x64 tiles

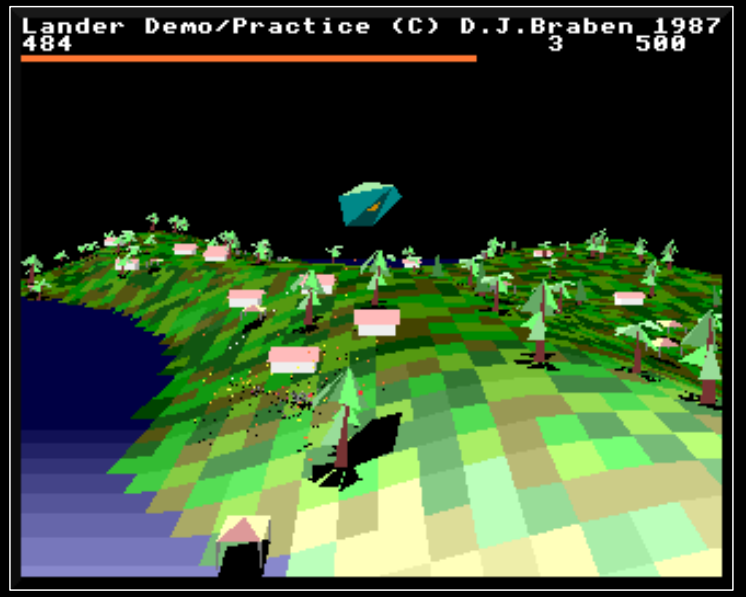




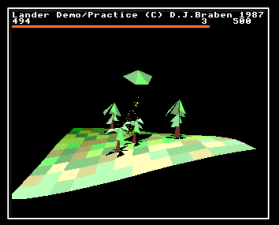
Comparing Big Zarch and Big Lander



How Zarch might have looked: A display technique that was rejected



Big Lander 121x121 tiles



Big Lander

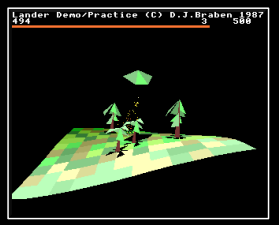
- Extract all tile-related values into run-time constants, e.g.
  - LANDSCAPE_Z_FRONT, CAMERA_PLAYER_Z, LANDSCAPE_X_HALF
- Code calculates all other constants from:
  - TILE_X, TILE_Z
- Code sets the size of tables and buffers accordingly
- Hack the tile colour routine to cope with arbitrary depths
- Fix a couple of issues with RISC OS 3.5+
  - Hard-coded screen addresses – use XOS_ReadVduVariables instead
  - Disable code that detected available memory by writing to location 0
- Pass tile size as build variables

Compendium Electron Elite

Speeding up Elite with sideways RAM

- On the Electron, memory is accessed at different speeds
  - &0000-&7FFF is accessed at 1MHz
  - &8000-&FFFF is accessed at 2MHz
- Move speed-critical code into sideways RAM to speed up the game
- Add logarithm-based multiply/divide routines from 6502SP and C64
- Add dedicated and unrolled line-drawing routines (HLOIN, LOIN)
- With sideways RAM, add all features from the BBC Micro disc version
  - Thargoids, suns, planetary details, missions, ship files, docking sequence
  - Plus Compendium features: flicker-free, joysticks, Trumbles

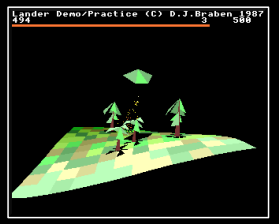
Speed comparison video

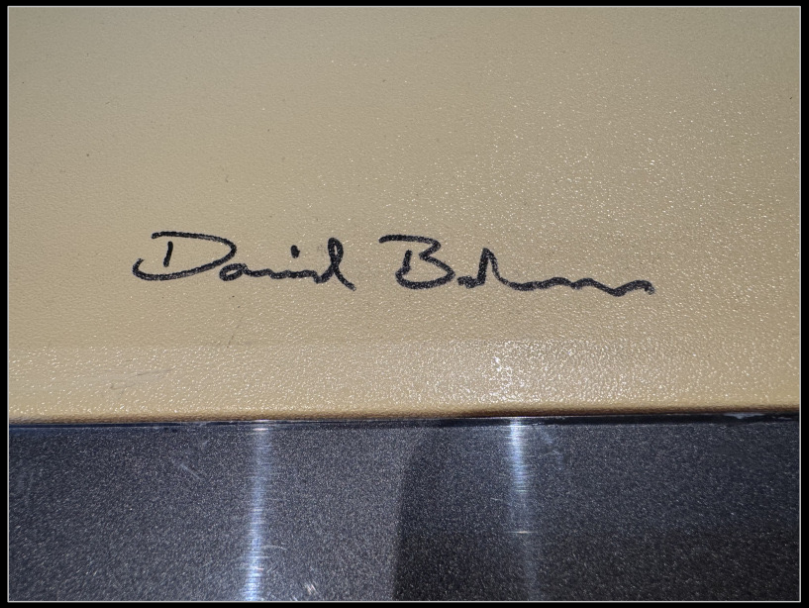


Master Elite on the BBC B+


ELITE
Load New Commander (Y/N)?
(C) Acornsoft 1986

I bought a B+


Lander Demo/Practice (C) D.J.Braben 1987
434
3
508

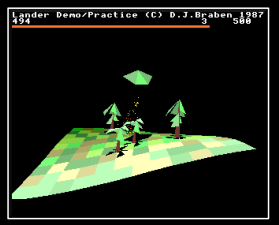

David Braben

Shadow RAM and Private RAM

- The B+ adds 32K of RAM to the BBC Micro Model B
  - 20K of shadow RAM (screen memory, &3000-&7FFF)
  - 12K of private RAM (&8000-&AFFF)
- Shadow RAM is limited, you can only write to the visible screen
- Private RAM is in addition to sideways ROMs, can't run languages
- Only VDU driver code can see shadow RAM
  - Private RAM &A000-&AFFF, MOS ROM &C000-&DFFF
- Fitting Master Elite into the B+
  - Enable shadow and private RAM and move screen-poking code to &Axxx
  - Shuffle workspaces and zero page to fit the B+ memory map

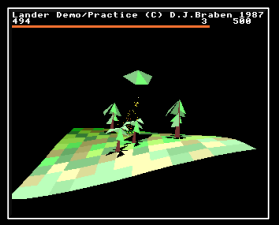
The Elite Family Tree

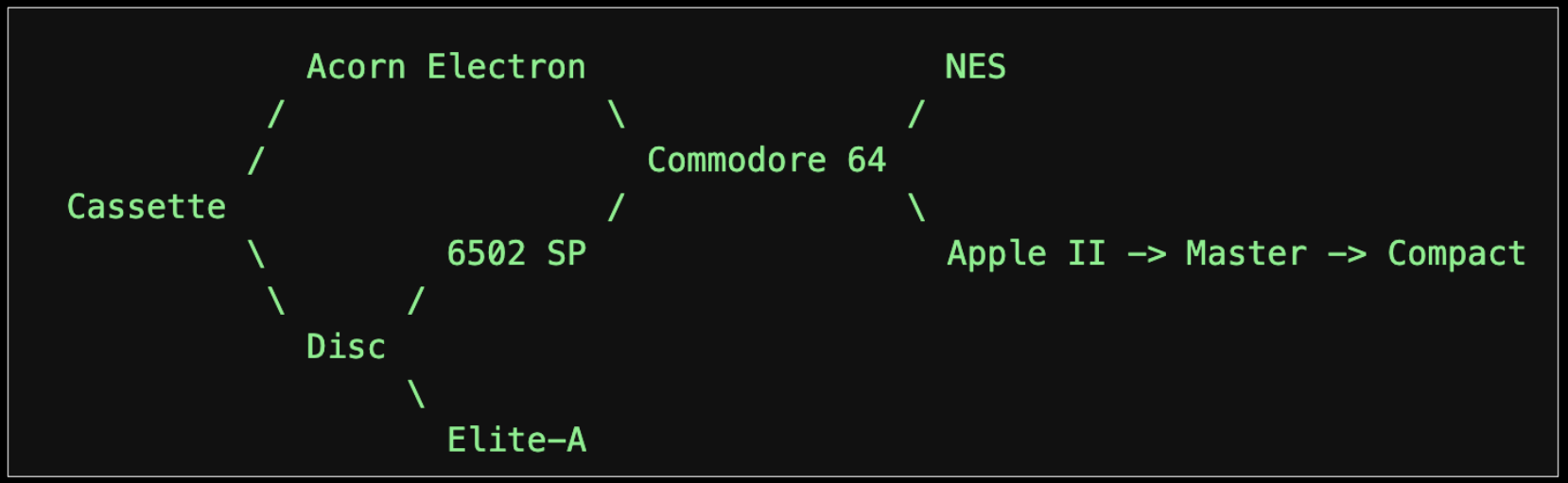
More sources released



- Ian Bell sources for 40th anniversary of release of Elite in Sept 2024:
  - Commodore 64 (built on a BBC Micro)
  - Apple II (built on a BBC Micro)
  - NES (built using Programmer's Development System)
- DFS doesn't contain date or version data, only a simple write count
- But source has clues as to which source was derived from which
  - Persistence of 6502SP code (DOXC, DOYC in parasite, SETXC, SETYC in I/O)
    - BBC, Electron, Elite-A -> 6502SP -> C64, Apple II, Master, Compact, NES
  - Scaling of the system charts for Apple II (SCALEX, SCALEY)
    - C64 -> Apple II -> Master, Compact

The Elite family tree





Finding out more

- Visit www.bbcelite.com and follow the links
- Follow me on Bluesky and Mastodon
- See you on Stardot