

Retro Compendium



Sean McManus
www.sean.co.uk

Introduction

My Amstrad gave me opportunities to explore new universes, build my own tiny worlds, and learn skills that would last a lifetime.

Through my contributions to Amstrad Action and Amstrad Computer User, the CPC 464 and 6128 also gave me my first published work.

In this ebook, I've collected my writing about the machine and retrogames generally. Mostly, it's from the revived Amtix CPC magazine which ran from 2021 to 2024.

There's an article from Crash magazine (which is still going) and a listing from Amstrad Addict.

I've included the two tutorials I wrote for Amstrad Computer User magazine, about colour swapping and making 6128 programs work on a 464.

This ebook also collects my listings together, from Amstrad Action, Amstrad Computer User, Amtix CPC and Amstrad Addict.

You can download my programs as an emulator disc, and even play most of them in your browser. I've updated many of them to make them faster and to improve their presentation.

My Amstrad programming book The Basic Idea is online here.

Because this ebook compiles pages from various magazines, the page numbers are not always in sequence.

I've inserted black title pages between the sections to help you find your way around.

Many thanks to Chris Wilkins, Colin Bell and the team at Amtix CPC. Back issues of Amtix CPC, Crash and ZZAP! are available at fusionretrobooks.com.

Thanks also to David Crookes and the team at Amstrad Addict, which you can find at www.addict.media.

About the author

Today, I write inspiring articles and books about technology, and work as a copywriter for tech companies.

My books include [Web Design in Easy Steps](#), [100 Top Tips: Microsoft Excel](#), [Mission Python](#), and [Scratch Programming in Easy Steps](#).

You might also enjoy [my free Coding Compendium](#), collecting my articles about [Raspberry Pi](#), [Python](#), [Scratch](#) and electronics.

[Download free chapters from my books](#) and [join my newsletter](#) at www.sean.co.uk.

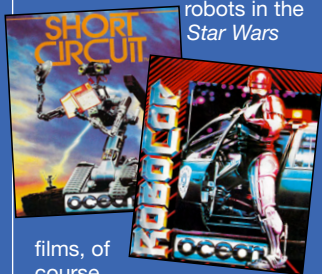
CAN ChatGPT WRITE AMSTRAD GAMES?



ChatGPT has been making a lot of headlines recently, with its ability to write articles, poems and even books. But can it triumph in the ultimate challenge and write Amstrad games? Sean McManus tests whether the latest artificial intelligence (AI) tool knows its Locomotive BASIC.

Article by Sean McManus

Hollywood loved AI in the 80s. We saw military robot Johnny 5 come alive in *Short Circuit*. *Robocop* policed the mean streets. And there were charming and quirky robots in the *Star Wars*

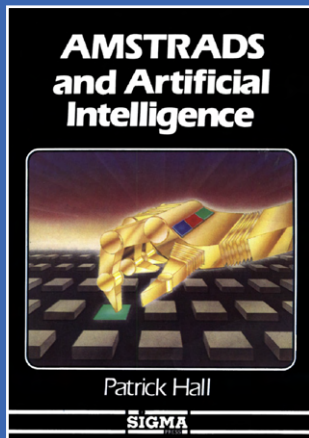


films, of course.

As computers entered our homes, putting us in control of these robots in games, it seemed like a new era of artificial intelligence was just around the corner. Could

a computer really be as smart as a human?

Some Amstrad BASIC programs showed us what was possible. The book *Amstrads and Artificial Intelligence* by Patrick Hall demonstrated how the computer could write poems and stories, translate German to English, and learn from answers to questions. These were great simulations of intelligence, but they were inflexible and limited by the amount of



data you could put into them.

In *Amstrad Action* issue 13 (October 1986), JC Ablett gave us the GIGO type-in, short for Garbage In, Garbage Out. With a brilliantly drawn *Max Headroom* character at the top of the screen, the program

spouted randomly chosen phrases in response to what you typed in. Sometimes, the conversation felt quite human, but it was just coincidence and wishful thinking, albeit great fun.

Recently, you might have



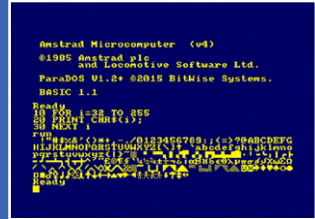


that we can understand. One thing ChatGPT is pretty good at, is writing programs in modern programming languages. It's been used to write simple games in JavaScript, the language that makes the web interactive, and it also does well in Python, a beginner-friendly language that has many real-world uses.

So, I thought: How would it fare writing Amstrad BASIC programs for me?

My first test was to ask it to write a program that prints out the Amstrad character set, creating a three-line program with a for-next loop. However, it printed characters from 0 to 255, which includes a lot of non-printable characters.

“Does that include control



which is a space, the first printable character. But the program stopped at CHR\$(126), which is the tilde character. In standard ASCII (as documented on Wikipedia), that is the last character, but we're missing all those fun symbols up to CHR\$(255), including my favourite, the square smiley face.

I asked it to try again. “That's good,” I said, patiently, “but the Amstrad has symbols up to CHR\$(255).

Can you modify the program to print them all, but still only print the printable characters?”

I hoped it would give me a loop that went from 32 to 255,

printing each character. ChatGPT couldn't get its head around the fact that the characters 127 to 255 are printable in Amstrad BASIC. The resulting program is shown in Listing #1. It only prints out the characters from 32 to 126, but pointlessly loops from 0 to 255 anyway.

This program shows one of the drawbacks of ChatGPT. Because it has no

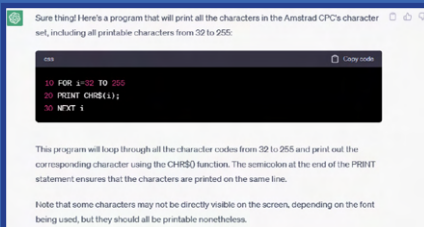


Above: GIGO - have a chat with 80s icon Max Headroom.

seen the headlines about ChatGPT, a new artificial intelligence tool that runs on a website (<https://chat.openai.com>). You type in your query, and it gives you an answer. Unlike those old Amstrad programs, its knowledge is extremely broad, and its output is incredibly diverse. It uses a technique called deep learning, which means it's basically taught itself based on a huge volume of web pages it's been fed. It doesn't attempt to actually understand anything; it just looks for patterns, and it generates patterns of text

codes? Can we only print out the displayable characters?” I asked it. “Good point!” ChatGPT replied, like a friendly programming partner. “The original program I wrote would indeed print out all 256 characters, including control codes. If you only want to print out the displayable characters, you can modify the program like this.”

The resulting program's loop started with CHR\$(32),



understanding of syntax, it can come up with programs that work (to some extent) but are somewhat inefficient or illogical.

I set it straight. “On the Amstrad, all the characters from 32 to 255 are printable. Can you write a program that prints them all please.” The “please” was a bit strained, but I don’t think ChatGPT noticed.

Finally, the computer gave me the program I wanted, shown in Listing #2. It runs as expected to display the character set.

As an experiment, this was a good starting point. It took four attempts to get the correct code, but ChatGPT was eventually able to take direction from me and generate the code I wanted. You might think that these results aren’t that great, but I was quite impressed. Remember, it hasn’t been taught Amstrad BASIC. It’s just learned it online. And it’s not a search engine, reproducing code from web pages: it’s producing new code that actually works using pattern matching.

It seems like ChatGPT doesn’t know much about the specifics of Amstrad BASIC, as opposed to other variants. That’s why my character set program turned out to be much more challenging than I expected. There isn’t much documentation for Amstrad BASIC online. Even where type-ins are published, it’s usually as a downloadable dsk file or as images in a PDF. Neither format is ideal

for training ChatGPT.

ChatGPT finds pens and inks confusing, another example of how it struggles with Amstrad-specific code. I asked it to print the character set in red, and it printed it in white by changing the ink in pen 1 to 26. I told it what happened, and it wrote me a new program that changed the paper colour to 1. The pen colour was the same, so the character set was now invisible.

I wondered whether ChatGPT could write me a simple game in BASIC, leaving aside any fancy presentation elements. I was looking for a game that would work on the CPC, but could be written by anyone with a reasonable understanding of BASIC from any machine.

Here’s the brief I gave it: “You are a programmer for the Amstrad CPC 464 computer. You write your programs in Amstrad BASIC. Can you write me a simple guessing game. The computer chooses a random number, between 1 and 100 and then asks the player to guess that number. If the player guesses right, the computer says “congratulations” and the game ends. If the player does not guess correctly, the computer tells the player whether they are higher or

lower than the number the computer has chosen and they are asked to guess again.”

I was excited to see the instructions unfold as they slowly appeared on my screen. The program used RANDOMIZE TIME at the start, a CPC-specific



technique for setting a random number seed. The variable names that had been used were meaningful, and the structure looked fine. The only problem was that it was trying to use INKEY\$ for the input instead of the INPUT command.

I told it, “That is excellent, thank you. The only thing you need to change is to use the INPUT command in line 40, instead of INKEY\$.”

Boom! On the second take, I had a program that worked perfectly, shown in Listing #3. It’s fairly generic

and would work on many 8-bit machines with minimal modifications. Nevertheless, ChatGPT had written me a working Amstrad BASIC game, almost by itself. Notice how it's embellished the messages shown to players, beyond what I wrote in my request.

It's not the most elegant solution. The program

There are lots of GOTOs. A WHILE structure would be cleaner, but that's implemented in different ways in different BASIC variants, so it's no surprise that ChatGPT couldn't get that. Importantly, it didn't attempt it and cause errors.

The program comes with line-by-line documentation, too, so it could help

is better because INKEY\$ only accepts one character, and you're asking players to enter a number up to 100. For this program, only INPUT works.

While ChatGPT can't understand the nuances of Amstrad BASIC (yet?), it can create generic BASIC programs that work on the Amstrad. If you're developing something simple, give it a

ChatGPT - Listings:

Listing #1

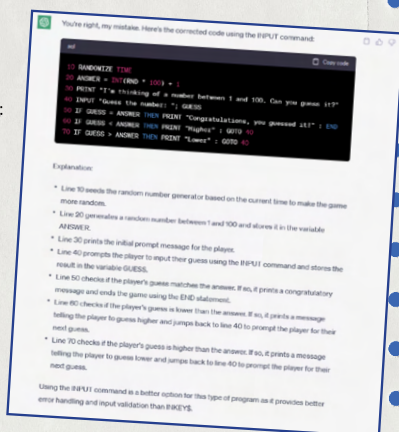
```
10 FOR i=0 TO 255
20 IF i>=32 AND i<=126 THEN PRINT CHR$(i);
30 NEXT i
```

Listing #2

```
10 FOR i=32 TO 255
20 PRINT CHR$(i);
30 NEXT i
```

Listing #3

```
10 RANDOMIZE TIME
20 ANSWER = INT(RND * 100) + 1
30 PRINT "I'm thinking of a number between 1 and 100. Can you guess it?"
40 INPUT "Guess the number: "; GUESS
50 IF GUESS = ANSWER THEN PRINT "Congratulations, you guessed it!" : END
60 IF GUESS < ANSWER THEN PRINT "Higher" : GOTO 40
70 IF GUESS > ANSWER THEN PRINT "Lower" : GOTO 40
```



structure matches my request exactly, although the order of instructions in my brief was not how a real programmer would write it. It was just the easiest way to brief ChatGPT. The program could simply check for higher and lower, and if it got beyond those statements, it would know the player guessed correctly.

beginners to learn BASIC. It's mostly helpful. Sometimes, ChatGPT makes up some real nonsense, though. It said: "Using the INPUT command is a better option for this type of program as it provides better error handling and input validation than INKEY\$." That's not correct: Using the INPUT command

go. Perhaps it can give you a helping hand, or at the very least, some sympathetic company on those long, dark evenings in front of a glowing screen.

If you'd like to explore ChatGPT for coding further, see my article for BBC News here: <https://www.bbc.co.uk/news/business-65086798>

In our second 'Flash Back' this Issue, Sean McManus looks at Ere Informatique's isometric adventure, *Get Dexter*. In its home country of France, it was known as *Crafton & Xunk*.



Your mission is to help Dexter the android, as he seeks the eight-digit code that will (somehow) prevent a war.

My first taste of *Get Dexter* came on the May 1986 Amstrad Action cover tape. It included a single room demo, but it was enough to show just how special this game was. There was plenty to do. You could bounce on the beds until they broke. Summon your companion Scooter and then jump on his head to reach up high. Destroy the robots, leaving just a puddle behind, and then slide around on the wet floor.

We'd seen isometric games before, but they felt very much like a sequence of single-room puzzles or a maze. *Get Dexter* feels like a real place you visit. While scenery necessarily repeats, the room designs are varied and make sense.

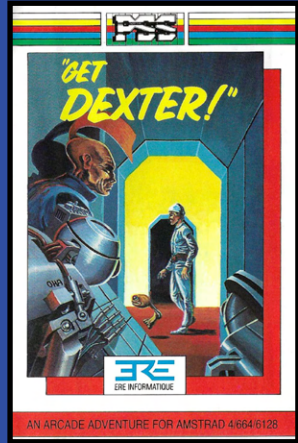
There's a dining room, a sick bay, and a kitchen area. There are open passages you can leave through, as well as doorways.

The environment behaves as you would expect in the real world, too. You fall off of spinning chairs and can drop objects onto a passing robot. If you make a mess, nobody magically clears it up when you leave the room. Sometimes enemies chase you, and sometimes they just roam around. They rarely



march up and down, as they mostly do in other games.

Humour is woven throughout the whole experience. Some other games had funny graphics, but this feels like you've stepped into a comic. The



main character's animation is great, and Scooter is a joyful addition. When you call him, he bounces over to you and makes a sound like a happy robot dog.

Try it today. It's spacebar to take, D to drop, and R to whistle for Scooter.

Sean



If you like isometric games, this is as good as they get. It's fast and teeming with life, with four or five enemies in some rooms. The game's hard, but you can use the booths scattered around to recharge your energy. With well-crafted graphics, superb attention to detail, and funky music, *Get Dexter's* quality has rarely been matched.

CRITICISM

SPEECH SYNTHESISERS



In this Issue's AM-TECH our newest member of the team Sean McManus takes a look at the world of Speech Synthesisers and chats to Richard Hanson of Superior Software, creators of Speech!

When I was a kid, there was nothing cooler than Knight Rider. I loved the turbo boosts, and the ejector seat, of course. But most of all, I loved KITT's slightly robotic voice. When the chance came to make my computer talk like that, I grabbed it.

My hardware of choice was the DK Tronics Speech Synthesiser. It was an expansion unit that plugged into the back of the CPC464, with a cable to connect to the audio output. One of the selling points was that it came with stereo speakers, all the better to enjoy the funky music from *Get Dexter*.

The DK Tronics unit



Above: The DK Tronics Synthesiser connected to the CPC peripheral port.

wasn't the only one available. There was the official Amstrad SSA-1 Speech Synthesiser, which was based on the same SP0256 chip, and also included stereo speakers. In France, TechniMusique et Parole Informatique (TMPI) made a speech unit, and there was a Spanish unit made by MHT Ingenieros with a built-in speaker. (If you're curious, the French unit did indeed have a French accent, and

could also sing).

Most surprisingly, there was a software-only package called Speech! by Superior Software, which I later obtained. It enabled you to build speech into your own BASIC programs, and



included a utility to copy the Speech! software to another disc containing your program.

We contacted Richard Hanson from Superior Software to find out more. "Speech! was the brainchild of David Hoskins, who programmed the BBC Micro and Commodore 64 versions," he said. "I remember being very impressed by the quality





of the speech that David had created purely using software – without any hardware add-on being required. We contracted a skilful Amstrad CPC programmer called Rick Peterson to code the Amstrad version based on David's original system."

Say Something, Say Anything!

All of the speech systems had an unlimited vocabulary, because they recreated the words using the sounds that make them up. Speech! had 49 of these phonemes, which covered vowel sounds, consonants, and special sounds. There were many more vowel sounds than vowels. For A, for example, there was AY (as in pale), AE (as in black), AA (as in car), and AI (as in fair). The special sounds included CH, CT, DR, and NG. There was an instruction to convert text into speech automatically, but you could improve the pronunciation by accessing the phonemes directly.

"I wrote a little spelling test program and a SAYFILE utility [which read files aloud], which we included in the package as an example of how Speech! could be used," said Richard. "The spelling test software has three categories (easy, medium, hard) with words like

'accommodation', 'desiccated' and 'isosceles' included in the hard category, while the easy category contains some

fairly simple words, such as 'tiger' and 'castle'. I remember one customer commenting that 'castle' had been given a northern England pronunciation. Well, I am a Yorkshireman!"

A nice feature in the spelling game is that it shows you a sentence with the context for the word you're guessing. That additional hint can be a big help. The speech quality is impressive, but some words can be hard to make out. I had to get it to say "parliamentary" a few times before I worked out what I was being challenged to spell.

Recorded Speech!

To create the original software, David Hoskins recorded himself saying the phonemes, and wrote software that manipulated the sound chip into playing them back. The sounds that used vocal

```

SPEECH 01.1 ©1986 Superior Software Ltd.
      Please wait

SPEECH HAS NOW LOADED

Commands available:
ISAY command
Format: ISAY, "HELLO"
ISPEAK command
Format: ISPEAK, "/BEHLLLOH"
IPEECH command
Format: IPEECH, number(1-20)

Ready to use "AMTIXCPC RULES"
Ready
  
```

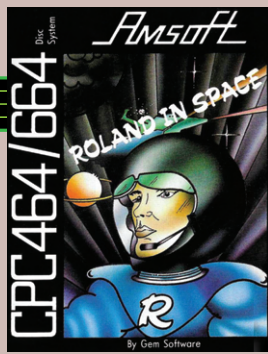
ords were said as long sounds ("aaaaa"), and then cut back to a smaller sample which could be repeated for playback. "It is very difficult to remember and

keep a steady pitch of voice throughout the recordings and it took many hours to complete [these] sounds," David wrote in 1986.

Plosive sounds like P and D were played once, with the software inserting the slight pause a person would make before the sound. For sounds like Z for zebra, David added in some S samples to give it the hiss it needed. You can read more about how the software was created in David's article in Your Computer magazine (January 1986).

I asked Richard how he rated the competition from DK Tronics and Amstrad. "If I recall correctly, the quality of speech was similar, but those hardware speech synthesisers were more expensive – so this helped the sales of our version," he said. "The Amstrad version of Speech! was a reasonable success in a different market for us. Superior's main software markets were the BBC Micro and Acorn Electron, followed by the Commodore 64, and those versions sold in higher numbers, but overall I remember being pleased with the level of sales achieved by the Amstrad version."

Sampled speech was widely used in games and there was a sampler type-in in Amstrad Action that used RSXs to replay sounds, but there were no other software packages for speech synthesis on the Amstrad. "There was probably enough market competition already with the hardware versions, but in retrospect it seems a little surprising that there



wasn't any other software competition," says Richard. "Maybe some of the other software companies attempted it but found it to be quite a complex project."

The hardware units and Speech! software all added RSX commands to BASIC that you could use in your programs. The hardware units both had the ability to intercept PRINT instructions and read out whatever is sent to the screen. Come to think of it, I really should have used that feature to check I'd entered programs correctly by getting the computer to read my listing aloud while I checked against the magazine. Reading a listing tended to be slow, though.

Surprisingly, the DK Tronics Speech Synthesiser, Amstrad SSA-1, and Speech! all used the same [SAY command to read a string aloud, creating a de-facto standard for using speech from BASIC. "I think David Hoskins looked carefully into the subject, and decided that SAY was important for our software speech synthesiser, and I think the authors of the hardware versions will probably have done the same - based on the published work that was available at that time," said Richard.

Games that talk back

Speech synthesis was rarely used in games. The most support came from Amsoft, Amstrad's own software arm. The software house had been set up because Amstrad understood the importance of using software to sell hardware and they did their best to stimulate demand for the

SSA-1. *Roland in Space* used speech to tell you how many lives you had left and which planet you were going to when you popped into the Tardis. *3D Stunt Rider* announced, "prepare to jump!" at the start of the game. *3D Boxing* gave you spoken prompts to enter your name and choose a menu item, and did the referee's countdown when a boxer was knocked down. The other Amsoft games including speech were *Alex Higgins World*



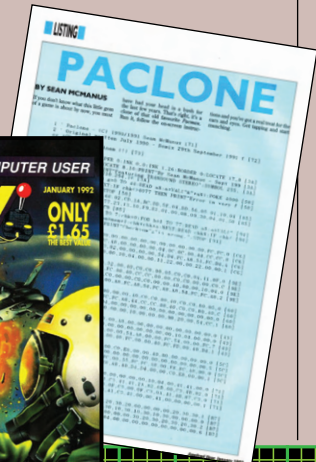
Glen Hoddle Soccer, Nuclear Defence, and Tubaruba.

Other independent games known to support speech synthesis were *Jump Jet* by Anirog and *Darkwulde* by Top Ten. My listing Paclone (a sluggish Pac-Man clone) from ACU January 1992 included a spoken

phrase if you had a DK Tronics unit connected, too.

In each game, the speech was an additional special effect, rather than a requirement to play the game. That's just as well. The market wouldn't support software that required speech hardware, and in some cases the speech was unintelligible. Although the speech synthesisers had nuanced controls for pitch, phonemes, and speed, it didn't often feel like they were being used.

Given how easy it was to support speech hardware, it's a shame that more games companies didn't do so. Those Amsoft titles are fondly remembered by many, but they're a long way from the best games on the machine. Imagine *Alien8* with amusing speech from the robot. Or *Gauntlet*, which included speech in the arcade ("Wizard needs food badly"). Or *Starstrike II*, which put you in the cockpit of a *Star Wars* style fighter craft. More support for speech synthesis would have added a whole



new dimension to Amstrad gaming.

With thanks to Richard Hanson. Superior Software is now called Superior Interactive and the company is publishing games for PC, iOS, and Android, including versions of their classic 8-bit titles including the *Repton* series. Visit www.superiorinteractive.com.

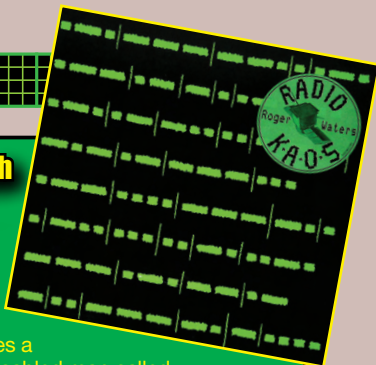
Doctor, doctor!

Here's a listing you can use to have some fun with speech on your Amstrad. It uses the |SAY command, so it should work with the hardware speech synthesisers or the Speech! software. I've written and tested it using Speech!. If you're using Speech!, run it first and then run this program.

This rather silly program

Magic Billy speaks through Speech!

Roger Waters, lead songwriter from Pink Floyd, used Speech! on a BBC Master computer in his concept album *Radio K.A.O.S.* The album features a radio DJ between the tracks. A disabled man called Billy uses speech synthesis to talk and calls in to the station. Over the course of an evening, he chats with the DJ about his life. The album memorably ends with Billy hacking the world's computers to fake a nuclear attack.



creates the illusion of having a conversation with your computer, who plays the role of a doctor listening to your woes. There have been lots of programs that attempt to analyse what you type in and give a sensible reply. This one just pretends to listen to you, and then says something random and supportive in reply.

You press space when you've finished talking, and press B when you've said goodbye. If you fancy you, can have a play and can easily extend this program to incorporate more random phrases.

Remember, it's only fun if you actually talk out loud.

```
++++ Listing Starts +++++
```

```
1 REM Doctor, Doctor! From AmtixCPC!  
2 REM by Sean McManus - www.sean.co.uk - February 2022.  
10 READ maxi:DIM answer$(maxi)  
20 FOR g=0 TO maxi:READ answer$(g):NEXT  
30 hello$="What would you like to talk about?":SAY,@hello$  
40 CALL &BB06:WHILE a$<>"B":SAY,@answer$(INT(RND*maxi))  
50 a$="":WHILE a$="":a$=UPPER$(INKEY$):WEND  
60 WEND  
70 bye$="Goodbye. Nice talking to you!":SAY,@bye$  
80 DATA 12,why,tell me more,how do you feel about that?,indeed  
90 DATA intreeging,why do you say that?,oooh,quite  
100 DATA do continue,hmmmm,is that right?,interesting,really?
```

```
++++ Listing Ends +++++
```

TOP of the CPC POPS

WILL YOU SPEND YOUR WOOLLIES VOUCHER ON THE LATEST SINGLE, OR AN AMSTRAD GAME? THE GAME, OF COURSE, BUT YOU NEEDN'T MISS OUT FOR A NUMBER OF GAMES HAD POP TUNES BUILT IN! JOIN US NOW AS SEAN MCMANUS COUNTS DOWN THE TOP 10 CPC HITS!

GREETINGS POP PICKERS!

Article by Sean McManus

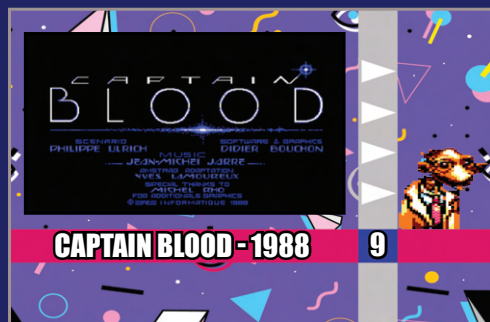
WELCOME TO ANOTHER EDITION OF 'TOP OF THE CPC POPS'. IF IT'S THE CPC THERE MUST BE KENNY LOGGINS, JEAN-MICHEL JARRE & DUANE EDDY. THEN THERE'S HAROLD FALTERMEYER, LIMAH! & MICHAEL JACKSON...

DANGER ZONE, EVERYBODY NEEDS SOMEBODY TO LOVE, NEVER ENDING STORY, TIME WARPI, BAD, THE HEAT IS ON... WE BEGIN IN 86, AT NO.10 WITH OCEAN SOFTWARE'S TOP GUN & DANGER ZONE!

This minor hit for Kenny Loggins (who previously charted with Footloose) featured in the movie and game *Top Gun*. Danger Zone charted at number 45. The Amstrad version confusingly uses the same sound for the voice and synth parts, but is otherwise a solid version of the song. The *Top Gun* game also features the *Top Gun* Anthem by Harold Faltermeyer, which won a Grammy for Best Pop Instrumental Performance in 1987. Of the two tunes, the anthem is the stronger one.



A NON MOVER AT NO.9 FOR CAPTAIN BLOOD & ETHNICOLOR



In 1988, Jean-Michel Jarre played landmark concerts in London's Docklands, with searchlights, fireworks, and projections turning the industrial landscape into his stage. That same year, *Captain Blood* was released, with *Ethnicolor* as its title music. It's the opening track on Jarre's pioneering sample-based album *Zoolook*. The tune is often masked by the distorted vocal samples that sound like the most wonderful alien jungle on the record. Think of the CPC version as a novel remix, stripping away the samples to uncover the core melody.

DOWN TWO AT NO.8 IT'S THE BLUES BROTHERS

If you missed out on The Art of Noise and Duane Eddy's cover of Peter Gunn, you can hear the tune in *The Blues Brothers* game. The loping bass line that holds it all together is double tracked when there's a channel free, so the bass sounds as fat as possible. The single charted at number 8 in 1986, but the tune originally dates back to 1959. The game also features Everybody Needs Somebody to Love, with some bluesy vamping behind the melody line and power chords for one of the vocal sections to break it up.



UP 20 TO NO.7 IT'S OUR HIGHEST CLIMBER – THE ROCKY HORROR SHOW



Hot Patootie! I really love that rock n' roll! *The Rocky Horror Show* is a musical about a transvestite alien who teaches two up-tight Earthlings how to party. Among other new experiences, Brad and Janet discover the Time Warp, a dance with its own song. Damian took his version to number 7 in the charts in 1989. CRL got there first, with *The Rocky Horror Show* game launching in 1985. A lot of Amstrad tunes sound like either synths or bells, and this one's a "bells" one in parts. It captures the rock n' roll spirit that underlies the original song, with each part feeling like it's performed independently, as if by a band. The only glitch is an odd low note just before the tune loops. If you don't know which way to jump or where to put your hands, see the title screen for dance instructions.

DOWN 4, IT WAS 2, IT'S NOW NO.6, THE NEVER ENDING STORY

The *Never Ending Story* was composed by Giorgio Moroder, performed by Limahl, and reached number 4 in the UK chart. The Amstrad version (like the single) starts with a gentle fade in that you don't notice it at first. At the end, it washes away, too, creating the illusion that the music – like the story – is eternal. There are no drums on this, but the synth arpeggios keep the momentum up, and the Amstrad hums the vocal lines beautifully. Full marks for capturing pretty much all of the single, including the solo. Bonus points for adding a pop song to a text adventure, which is rare. If you haven't seen the film, do watch it. It's magical.



AND NOW, THE TOP 5!

UP 7 TO NO.5 IT'S MICHAEL JACKSONS MOONWALKER BY US GOLD

Jackson was multimedia before there was multimedia. Already famous for his cinematic pop videos, Jackson starred in pixelated form in the Amstrad game *Moonwalker*, based on the film of the same name. Jackson's single *Bad* reached number 3 in the UK single charts. The Amstrad rendition in the game is not bad at all. The game also plays *Smooth Criminal* (a top ten hit at number 8), and *The Way You Make Me Feel* (number 3). That last one plays near the end of the game, but it's no reward. It sounds terrible, with a beepy synth voice and frequent laser blasts interrupting the flow.



OUR HIGHEST NEW ENTRY AT NO 4, WHO YA GONNA CALL? – GHOSTBUSTERS!



Ray Parker Jr took the *Ghostbusters* theme to number 2, with its memorable refrain of "Who you gonna call?". It's almost a rap in parts: tricky to do on a computer. The first *Ghostbusters* game gives you a karaoke rendition that displays the words for you to sing, a neat solution. *Ghostbusters II* remixes the song with oomph, and adds a melody for some of the rap parts. It's surprising in places, but it works. The *Real Ghostbusters* game offers another catchy interpretation. Those last two versions sound funkier, but it's most fun to sing along to the original. I ain't afraid of no ghost!

DOWN ONE TO NO.3 IT'S DOMARK'S 007: A VIEW TO A KILL

This Bond theme reached number 2 in 1985, performed by Duran Duran with a 60-piece orchestra. Not much to live up to then. The version in the *A View to a Kill* game uses the original melody and adds some nice trilling bass effects. As a b-side, if you like, you get a strong performance of the Bond theme in the game, too. The game itself is nothing special. If I'd bought it, I'd probably have played the music more than the game.



UP ONE TO NO.2 IT'S LAST WEEK'S NO.1, TYNESOFT'S BEVERLY HILLS COP



This song also spun out of a hit movie. The instrumental track Axel F was named after Eddie Murphy's character in *Beverly Hills Cop*



Hills Cop and hit number 2 in the UK chart in 1984. The Amstrad version in the *Beverly Hills Cop* game is such a perfect cover, you could almost believe it's the original. Sure, the drums sound scratchy (especially at the start). The Amstrad's sound chip, much as we love it, is no match for the Jupiter 8 and Moog synths that performed the lead and bass of the original. But, everything is where it belongs, and sounds great. It helps that the original has a minimalist arrangement, with perfect interplay between the bass and lead parts.



WHICH MEANS WE HAVE A NEW NO.1. RELEASED IN 1986 BY OCEAN SOFTWARE IT'S WEEK 1 OF 4 FOR FRANKIE GOES TO HOLLYWOOD!

Driven by tabloid headlines, controversial videos, and endless remixes, Frankie's first three singles all went to number 1 in the charts. Their game saw you completing challenges to become a complete person, with a murder mystery and a number of arcade-style minigames. There's a version of the band's hit *Two Tribes* while you play, which is our highest charting song in a CPC game. It's a short loop, but it captures the riff of the original nicely and overlays the melody well. As a bonus, the original game box included a tape recording of a live version of *Relax*. With either of these songs, you'll have the tune going through your head all day.



YOU CAN HEAR ALL THESE POPTASTIC HITS ON

SEAN'S SITE AT: [HTTPS://TINYURL.COM/6E4ESBV2](https://tinyurl.com/6E4ESBV2)

Retro Computer Festival 2023

Cambridge is the birthplace of the ZX Spectrum, Acorn Electron and, more recently, the Raspberry Pi. What better place for a museum dedicated to computing?



The Retro Computer Festival 2023 on November 5th gave vintage gaming fans an opportunity to see a wide range of Amstrad machines, including a rare CPC 472.

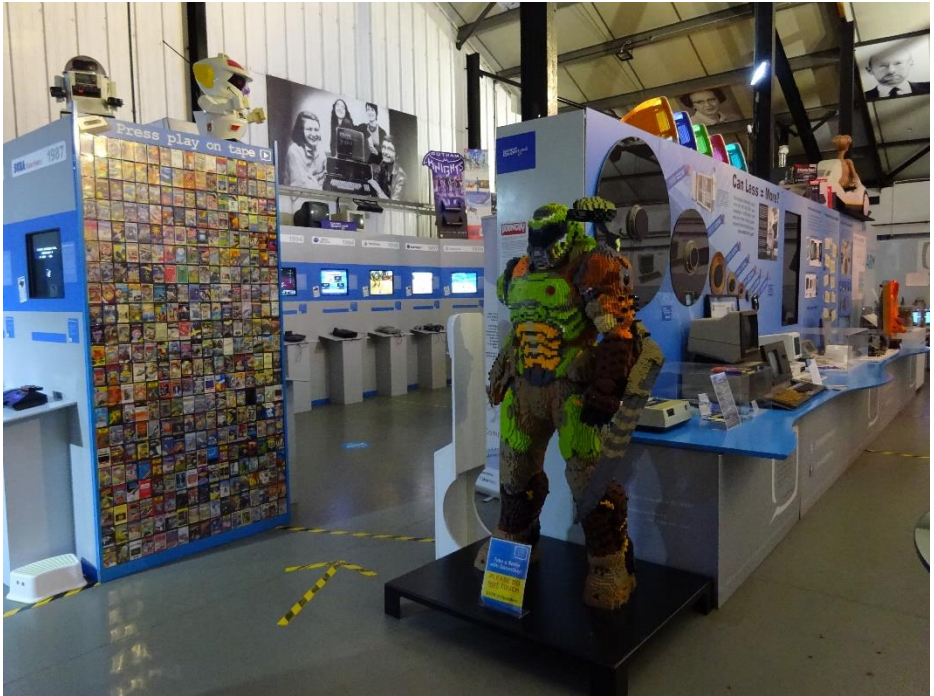
The event took place at the Centre for Computing History, which is tucked away in the back of an industrial estate in Cambridge.



This museum opened in 2013 and houses a wide range of computers and video game machines. There's an emphasis on British technology. Computers from Cambridge (including from Acorn and Sinclair) are proudly labelled.

For the Retro Computer Festival, about 30 collectors and hobbyists set up their own computers to complement the permanent displays. Pete Golding brought along a collection that included a Schneider 64K machine with brown keys where the green keys should be, a CPC 664 with its ridiculously chunky cursor keys, and the CPC 6128 Plus and GX4000. His CPC 6128 was running Get Dexter, one of the finest games for the Amstrad (see our Flash Back in issue 5).

The most intriguing machine in his collection was the CPC 472, which was exclusively sold in Spain. When the Spanish government imposed an import tax on computers with 64K or less in 1985, Amstrad simply added another 8K of memory. The catch? It wasn't connected to anything. "We had basically stuck two fingers in the air to the Spanish government," Alan



Sugar wrote in his autobiography. "As a result of this exercise, we managed to sell at least another 50,000 units in the Spanish market in the Christmas season," he added.

The CPC 472 at the museum was displayed with the case open, so you could see the extra board that was created with the redundant RAM chip. This one had a BASIC 1.1 (CPC 664) ROM chip on it, so it was one of the earlier CPC 472s. Later machines reverted to BASIC 1.0 for greater compatibility.

For Amstrad fans, there are several other exhibits of interest in the permanent collection. A CPC 6128 was equipped with a Dandanator providing instant access to some of the best games, including Arkanoid and (isometric) Batman. The PCW 8256 word processor was powered up, as was the PC 1512, both of which made serious computing much more accessible to homes and businesses. Also on were the portable but clunky PPC 512 and the E-mailer phone, as seen on The Apprentice and

almost nowhere else. I got to play a GX4000 for the first time, but the NC100 notepad wasn't turned on.

There's a wide range of other unique and production machines, aside from Amstrad relics. A prototype ZX Spectrum, a BBC



Micro signed by members of the development team, and one of Douglas Adams' old computers are on show. The machine that used to run London Underground's Bakerloo line is about the size of three large fridges. There's a row of home computers with games running for you to try. I loved playing Centipede and Space Invaders on original arcade cabinets, too.

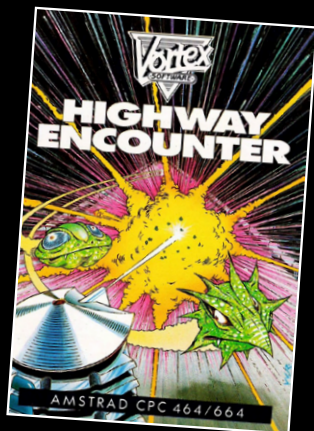
You'll probably find they have machines from your past. I enjoyed seeing the 380Z and Nimbus by Research Machines, similar to the computers I first used in school. We had an ICL One Per Desk on loan at home for a bit when I was a kid, so it was nice to meet that again here.

The museum is well worth a visit, and if you can tie it in with a retro weekend like this, all the better. It was a friendly event and the festival gave me a chance to see machines such as the Sharp MZ-80K (where you had to use the character set for all graphics), the Aquarius II (believed to be the only working one in the world), and a recreated Bywood Scrumpi, a single board kit computer from 1976. This was shown by Tim Gilberts (also known for his work at Gilsoft, which sold The Quill adventure writing system).

Thank you to everyone who shared their collections and expertise at the festival and to the staff at the museum.

AM CPC *FLASH* TITLES *BACK*

For our second 'Flash Back' this issue, Sean McManus and his Vortons are on a mission to save Earth from an alien invasion! Time to hit the road and get the Lasertron into position in Vortex Software's 3D isometric masterpiece, *Highway Encounter*.



It's 1985, and Colin Baker's in the Tardis. Someone's given Davros a bump on the head, and the Daleks have all turned friendly and want to help the humans. At least, that's how it seems.

In *Highway Encounter*, your mission is to save the Earth from an alien invasion (again). You look, move, and zap like a Dalek, although you're called a Vorton. There are five of you, and you must push a weapon called a Lasertron to the end of the last road, where the aliens conveniently await destruction.

You play one of the Vortons and the rest automatically advance, pushing the Lasertron. When there's just you left, you have to push the Lasertron yourself, and the mission gets much harder.



Along the way, there are obstacles to dodge, move, or destroy with your blaster, and plenty of aliens to shoot. An eye on a stalk spins around, surveying the highway. A giant mouth opens and shuts as it slides around the road on its slug-

like foot. These aliens are proper non-player characters: if you shoot them, they don't regenerate. They move even when you can't see them.



Above: *Vorton No.4* is go! Navigate the treacherous and seemingly endless highway.

You can shoot the aliens in the next zone if you blast off the screen and get lucky. And they can attack your Vortons if you leave them behind and an alien sneaks past you.

A good strategy is to block the Vortons' path while you go on to kill the aliens on later screens. There's a time limit, so you can't afford to leave them far behind. If your Vorton is destroyed, you take the next one in line, so you risk going back several screens if you die.

Highway Encounter earned an Amtix Accolade in issue #1 with 92%.

Sean



It's an original idea that's challenging but fun to play. The concept of multiple lives is justified within the game world, unusually, and the aliens seem like real creatures.

Highway Encounter is a design classic from start to finish. The loading screen reproduces the powerful box art. The title screen is well laid out, with the game's name emerging from the ground in 3D. As you advance along the road, you see different plants in the background and the architecture of the highway changes. The sound effects are sparse but good, and the explosions are beautifully animated. At the end, there's a rewarding 45-second completion sequence, where the Lasertron advances past hordes of aliens and destroys their giant ship. It's Dalek-table!

CRITICISM



CASSETTE 50



It inspired the Spectrum's crap games contest and has been called an early example of "shovelware", where software is sold on quantity over quality. For the Amstrad CPC Annual, Sean McManus bravely takes another look at Cassette 50.

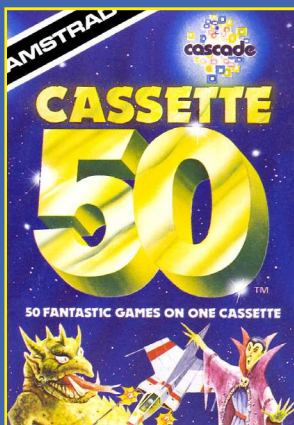
There were 50 Amstrad games waiting for me under the tree on Christmas morning, 1985. They were all on a single tape; the infamous mail-order compilation, *Cassette 50*.

I'd specially requested it, sucked in by the advert that promised "super" and "fabulous" games. Above all, I'd been swayed by the free gift of a calculator watch. These gizmos were the smartphones of their day: the pinnacle of portable technology. At the time, they cost more than £15 in the Argos catalogue. Software publisher Cascade was selling *Cassette 50* for £9.95,

postage free, and throwing in a watch. How did



they do it? Well, the first thing they did was to buy the games very cheaply. Matthew Lewis wrote *Galaxy Defence*, which ended up on the Spectrum version,



Above: The iconic *Cassette 50*.

and was paid just £10 for all rights. He didn't

even get a free copy of the tape. The second thing is that Cascade set the quality bar low. I would not be surprised to hear they published the first fifty they received.

Cassette 50 was marketed on most home computer platforms. There were different games on each version, and the Amstrad one was also available as *Disk 50*.

Cassette 50 then

My recollection is that I rather liked *Cassette 50*, which I realise is an unpopular opinion today. I was chuffed with the watch and considered the tape to be the free gift. Wikipedia says that the games were mostly BASIC, as if that's an inherently bad thing. For me, that was part of the fun: being able to peek at the code and tweak it. If you think of *Cassette 50* as a



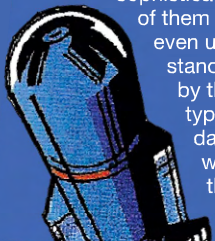
compendium of type-ins, that's more realistic than expecting commercial quality games for 20p a pop. Seen through that lens, *Cassette 50* wasn't so bad.

For the Amstrax CPC Annual, I took a fresh look. While *Cassette 50* has not aged well, I was surprised to discover a fascinating reflection of the Amstrad scene at the time.

Cassette 50 now

Imagine you could travel back to 1985 and pop into homes around the country to see what people were programming on their Amstrads. You'd find some people poring over the character tables at the back of the manual for symbols they could use in games. Maybe some were taking their first steps in machine code, surrounded by "Computing with the Amstrad" tutorials. One or two amateur programmers would be excited because they'd got their first game working.

Now, imagine you collected up their programs and put them in a time capsule. That's what *Cassette 50* feels like today: a snapshot of hobbyist programming, around 1985. The Amstrad had only been out for a year then, so it's little wonder that the games aren't especially sophisticated. Most of them weren't even up to the standards set by the lengthy type-ins of the day. But they were exactly the kind of thing



cascode CASSETTE 50

PLUS SPECIAL FREE WATCH OFFER...

50 COMPUTER GAMES ON ONE CASSETTE

YOURS FOR £9.95

Including FREE Watch and Postage

CASSETTE 50 IS AVAILABLE ON

ATARI DRAGON ELECTRON

VIC ORIC ZX Spectrum Amstrad

FREE calculator watch with every **CASSETTE 50**

EXPRESS DELIVERY ORDER NOW!

ONLY £9.95

WORTH OVER £10

20 KEY, MULTIFUNCTION CALCULATOR WATCH with every order for a Cassette 50

- Normal 12 and 24 hour time mode
- Calendar mode
- Alarm mode
- Calculator mode

NAME _____

ADDRESS _____

POST CODE _____

COUNTRY AU 3085

AMSTRAD ATARI ORIC ZX SPECTRUM

COMPTON AMIGOS DRAGON ELECTRON

CASSETTE 50 CAN BE ORDERED FROM ANY OF THE FOLLOWING:

AMSTRAD: NORTH YORKSHIRE, HE21 5BGL, ENGLAND. TELEPHONE: 04323 66666. POSTORDER NUMBER: 170004

AMSTRAD CPC 464

Read the rockets and anti aircraft game and CYRUS II, the hero, into the most skilled pilots succeed. Free yourself in different scenarios in this original adventure to search for the KING OF DRAGONS.

MEME is a sophisticated version of Othello with many colorful options and levels of play.

To mention only three of the 50 games on the Amstrad Cassette 50, which includes arcade type games, war games, adventure games, logical, tactical, and educational games, a flight simulator, and a business strategy game. Cassette 50 features multicoloured and user-defined graphics, graphics compatibility, sound and music utilising the amazing power of your dataset.

1. Maze Fester	17. Patton Bat	36. Submarine
2. Cyborg	18. Star Wars	37. Space Wars
3. Space Mission	19. 2D Maze	38. Space Wars
4. Lunar Landing	20. Chess 9	39. Space Wars
5. Fantasy Land	21. Force Field	40. Draxman
6. Soccer	22. Soft 9	41. Football
7. Tennis	23. Tunnel Escape	42. Day of the
8. Attack	24. Star Wars	43. Star Wars
9. Vampire	25. Star Wars	44. Star Wars
10. Galactic Attack	26. Star Wars	45. Star Wars
11. Vampire Attack	27. Star Wars	46. Star Wars
12. Knight and Crosses	28. Nemesis	47. Star Wars
13. Royal Race	29. Jet Flight	48. Adventure
14. King King	30. King King	49. King King
15. Crazy Castle	31. Invasion	50. King King
16. Amstrad	32. Invasion	51. Invasion

Clipped on cassette, compatible with 654 and 6128 Amstrad and game may vary for most type of computer.

Postage FREE for UK and £1.00 for overseas orders. Free watch with every cassette. Game will be dispatched within 7 days.

London & Home Office: **FRANCAIS** Ltd, 400 Broad Street, London W1P 1JF. **Calculator** cassette 50 is available from any of the following:

AMSTRAD: NORTH YORKSHIRE, HE21 5BGL, ENGLAND. TELEPHONE: 04323 66666. POSTORDER NUMBER: 170004

Above: Fancy a free calculator watch? Order your copy of *Cassette 50* now!

lots of us were writing in our bedrooms.

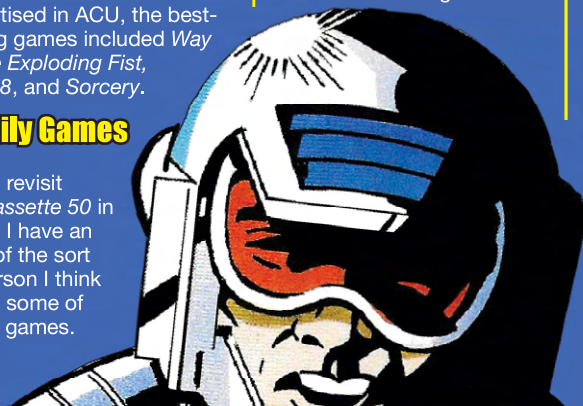
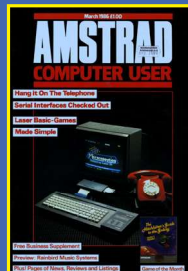
The inlay says: "Cassette 50 features multicoloured and user-defined graphics... utilising the amazing power of your Amstrad." That's quite sweet really: Bragging about using the PEN, INK, and SYMBOL commands. For historical context: when *Cassette 50* was first advertised in *ACU*, the best-selling games included *Way of the Exploding Fist*, *Alien 8*, and *Sorcery*.

Family Games

As I revisit *Cassette 50* in 2022, I have an idea of the sort of person I think wrote some of these games. They

probably had a good job, saw a had a few quid spare, saw the computer craze, and decided to buy an Amstrad. Of an evening, they'd do a bit of programming, discovering what the machine could do. As they learned new skills, they looked for projects they could build.

They wrote a simple maths quiz, and an anagram game



(they're both here). Then they ran out of ideas.

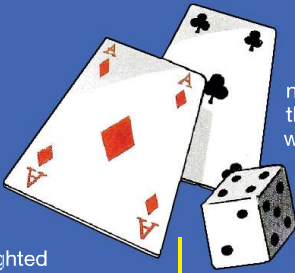
They looked around the room until their eyes alighted on a dusty box of games under the bed. Inspiration struck.

A good number of the games on *Cassette 50* are card, dice, or board games. There's a primitive version of Brucey's *Play Your Cards Right*, *Pontoon*, and *Three Card Brag*. For dice games, there's *Backgammon*, *Yamzee* (a version of Yahtzee), and *Craps* (snigger!). There's the board games *Draughts*, *Noughts and Crosses*, and *Solitaire* (where you must move balls to end up with only one at the end).

Most of the family games on *Cassette 50* aren't much fun. Some are little more than dressed up random number generators. It's particularly bizarre to use a £250 machine for *Pontoon* and *Play Your Cards Right*, when you could just use a 50p pack of cards.

The horse racing game and the business strategy games, such as *Trucking* and *Star Trek*, are also fancy electronic dice at their core. Gambling fake money on random numbers doesn't do much for me.

The *Hangman* game was (presumably) written by someone who hadn't learned to use arrays and random



numbers yet, so there's no built-in word list. It requires one player to enter a word and another to guess it. In that case, the Amstrad is playing the part of a piece of paper, albeit with letter checking built in.

There's a maze game with 'Dungeons & Dragons' style fighting added, but there aren't enough monsters to make it interesting.

A few of the family games were highlights of the set, though. I spent a long time on *Solitaire*, and enjoyed playing *Draughts* against the computer. The Othello clone *Nemesis IV* was fun, too. In the early days of computing, the artificial intelligence of a computer opponent was still quite impressive. Nobody would have bought these games as standalones but they're nice to have. They still stand up today.

Arcade Games

There's a selection of action games, of course. My favourite game of the lot (then and now) was *Fantasy Land*, a fast platform game written entirely in BASIC. It used colour swapping to change the position of the enemies, which was a brilliant touch.

Two games feature *Pac Man*, but neither of them plays like *Pac Man*. *Maze Eater* is more of a strategy game than an arcade game because the ghosts all home in on you and you

can only progress by trapping them behind walls. The game *Ghosts* should really be called *Portcullises*, because that's what they look like. This game uses fruit, ghosts, and *Pac Man* but doesn't have a maze or any other resemblance to the arcade classic.

Don't you hate sci-fi films that are packed with ear-wrecking explosions, even though they're in space where there's no air? Authenticity, people! *Rush Hour Attack* isn't making that mistake. It's silent. You point your cannon at the colourful aliens parading by in an orderly fashion, very much like cars in lanes on a road. Hence, the title, presumably. The aliens can't fire back, so you just blast for a bit until the time runs out. It's better than a lot of the other programs, but it seems more like a sprite demo than a fully realised game.

In *Hopping Herbert*, you have to jump through the moving gaps in the platforms above you to reach the top of the screen. It's infuriating when you fall all the way to the bottom, but if you can forgive that, it's not a bad game.

Inferno should be called *Inertia*, because of the way your ship moves with a sense of weight and gravity. It's a single-screen game, where you fly around shooting a target that's also moving, while you dodge the wreckage from past targets. This is one of my faves because it's a simple idea, nicely executed. There are just two moving objects, which is exactly the kind of strategy you need to write something fast and fluid in BASIC.

As you dig deeper in the collection, you find that the same ideas come up repeatedly. There are a



couple of games where you're supposed to fly your spaceship through a corridor. One's completely unplayable, and the other one's okay on its slow speed setting. Both of them use randomly generated levels that can be impossible to navigate. *Creepy Crawley* basically has the same game mechanics, rotated 90 degrees.

We all get a bit nostalgic when we play on the Amstrad, but it really hits me in the feels when I see games based on the rocket in the character set. *Royal Rescue* and *Lunar Landing* both use CHR\$\$\$(239) but do it a disservice with a weak game. It doesn't matter, though. *Space Pod Rescue* is pretty much the same idea, but a much better execution in machine code with colourful sprites.

With its day-glo colour scheme and single button play, *Whirly Bird* is like a Fisher Price My First Arcade Game. Perhaps the undocumented way to start the game (press Enter) is a parental lock? Major Blackthorne is low on helicopter fuel, so he must drop bombs on New York's skyscrapers to clear a landing path, it says here. Imagine the headlines! As helicopter games go, this one's more playable than *Airwolf*. And it's better than *High Rise*, which is a BASIC implementation of exactly the same idea, also on Cassette 50.

Is it any good?

I came into this article expecting to defend *Cassette 50* against its terrible reputation. I can't



Above: *Disk 50* - Instant access to erm... great games.

do that, sadly. Although I enjoyed a handful of the games this time around, they're mostly primitive and unimaginative. A lot of them feel incomplete, and some of them are unplayable. It made me laugh to see "Cascade" plastered all over them, like the company thought it was building up some kind of brand equity. *Attacker* is bugged in a way that shows even its creator didn't play it twice before it was released. Nonetheless, as a snapshot of bedroom programmers in 1985, *Cassette 50* is unique and has a certain charm. It's

a shame *Cascade* didn't repeat the exercise a few years later, when we'd all learned a bit more and had the tools to create better games.





Maze Eater



Cyclons



Handicap Golf



Rush Hour Attack



Royal Rescue



Star Trek



Whirley



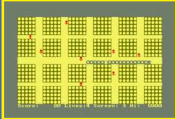
Attacker



Fighter Command



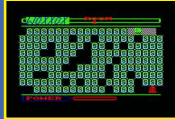
Draughts



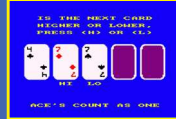
Evasive Action



Noughts & Crosses



The King's Orb



Play High/Low



Exchange



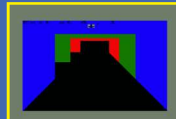
Hang Man



Pontoon



Firemen Rescue



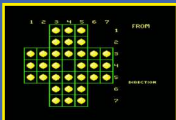
3D Maze



Colony 9



Backgammon



Solit



Yamzee



Craps



Creepy Crawly



Three Card Brag



Trucking



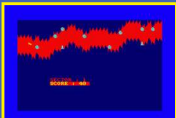
Rally 3000



Sitting Target



Nemesis IV



Space Ship



Jet Flight



Dragon Maze



Intruder



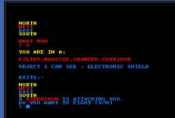
Inferno



Ghosts



Fantasy Land



Space Base



Planets



Hopping Herbot



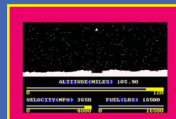
Dynamite



Timebomb



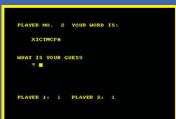
Day at the Races



Lunar Lander



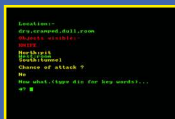
Space Mission



Rats



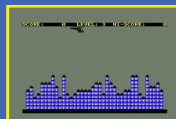
Motorway



Dungeon Adventure



Space Pod Rescue



Hi Rise

MISSED AN ISSUE?
WE'VE GOT YOU
COVERED!

ORDER

WWW.FUSIONRETROBOOKS.COM

SUBSCRIBE

WWW.PATREON.COM/AMTIXCPCMAGAZINE



FINAL ISSUE!

BINDERS



AVAILABLE FROM

WWW.FUSIONRETROBOOKS.COM

Picade arcade cabinet for the Raspberry Pi

Bring gaming to your desktop, with emulation of vintage arcade cabinets and classic home computers.

From its earliest days, one of the popular uses for a [Raspberry Pi](#) was retro gaming. The Raspberry Pi is powerful enough to run emulators for classic arcade machines, home computers, and consoles, and part of the appeal is that you can house the tiny computer inside a ZX Spectrum case or a cabinet.



Pimoroni makes an arcade cabinet called Picade that provides all the parts you need to build a desktop arcade cabinet.

There's a responsive joystick, six coloured arcade cabinet buttons for the top surface, a further two buttons on the front, and three more on the side (one for power).

The kit includes the cabinet frame, the Picade add-on board for connecting the controls to the Raspberry Pi, and an impressive lightweight HDMI display.

You'll need to add your own Raspberry Pi, micro SD card, and power supply.

Self assembly

I assembled the unit over a couple of sessions of two or three hours. You could probably do it in a single day, but I found I was ready for a break after a while. It's fiddly work, although it doesn't involve any soldering. It's satisfying to build, though, and the unit itself is beautifully designed. It feels like a substantial games machine, despite being much smaller and more convenient than a real upright arcade cabinet.

Classic arcade

For games, you use the open source RetroPie software. This supports a wide range of emulators, including classic home computers (such as the [Amstrad CPC](#) and ZX Spectrum), classic consoles (including the Game Boy, Nintendo 64, and PlayStation 2), and arcade machines.

My main interests are in the classic arcade machines and the home computers. Although the software is old, it's still protected by copyright, so you should be using software that you have a licence to use.

Nevertheless, the easiest way to get that software is to download it, rather than try to convert your old arcade ROMs or cassette tapes yourself.

The arcade games use the MAME emulator, and you'll need to find ROMs that match the version of MAME you're running. One gotcha is that some ROMs require other ROMs to run, so you might find you need to install a game you're not that fussed about to get a favourite working. Some ROMs also require sound effects to be downloaded and installed separately.

Arcade games play particularly well. They look brilliant on the screen, sound fantastic, and are a perfect match for the interface.

Home computer heaven

Classic home computers can be a bit trickier, including my favourite, the [Amstrad CPC 6128](#). Many games that you can play on a joystick require a keyboard to start them.

The right flipper button works as Enter, so it's easy enough to start games like Batman that use the Enter key.

You can just plug a Raspberry Pi (or any USB) keyboard into the back of the unit, but it means taking the back off the cabinet, and you have the clutter of a keyboard lying around. Part of the appeal of having an arcade cabinet is that everything is contained within it.

Fortunately, there is a virtual keyboard you can use for the Amstrad emulator (Caprice32), but it took some trial and error for me to get it working.

First, I had to add:

```
input_libretro_device_p1 =  
"513"
```

to \\retropie\configs\
amstradpc\retroarch.cfg,
as [described here](#).

Then I was able to show the keyboard using Start + B. To activate the joystick with the keyboard, I used Select + Joystick Right. Then I could use the joystick to point to a key on the keyboard, and the A button to select a key. Start + B hides the virtual keyboard again.

Remember to use the on/off button to switch the cabinet on after the power is connected. It doesn't automatically start like Raspberry Pi devices usually do. I thought mine had died when I

reconnected it after a long break and nothing happened.

Adding games

There are several ways to copy the files to RetroPie. You can insert a specially formatted USB drive.

The easiest approach, I found, was to access \\retropie over the local network and then drag and drop files from my Windows machine. The RetroPie interface only shows the emulators for which you've added games.

Top tip: It's better to have 100 games you love, than to have 1,000 unknown games clogging up the menu.

Saving the screen

You can customize the screensaver with your own images, by adding pictures to the folder \\retropie\configs\all\emulationstation\slideshow\image. I had to create the folder first, and then change Settings > UI Settings > Screensaver Settings. Set the screensaver behaviour to slideshow and change the slideshow screensaver settings to enable custom images.

AMTIX **FLASH BACK**

For our second 'Flash Back' this issue, Sean McManus travels back to 1986 and revisits the classic sliding puzzle game *Split Personalities*, by Domark Software.

SPLIT Personalities

Reagan's gone to pieces. Thatcher's in bits. Kinnock's all over the place. Yes, it's *Split Personalities*, the frantic sliding puzzle game where you put images of celebrities back together.

The game's original name was *Splitting Images*, and the choice of characters certainly looks inspired by the TV show *Spitting Image*. It's hard to imagine Reagan and Thatcher, let alone Kinnock, being the first choice for an



cept the frame starts empty. Pieces come in at the top left when you request them. You can fire them up, down, left, or right. They keep moving until they hit the wall or another piece. Putting the faces together requires both logic and luck because the pieces arrive in a random order.

There are doors around the edge of the screen that you can fire bombs or pieces through. There are also cracks in the wall that bounce the pieces back, and on later levels, the pieces bounce off each other, making them harder to position within the tight time limit.

Some special symbols arrive along with puzzle pieces. The bomb ticks down and explodes, taking one of your three lives unless you can hit it with a water tap or propel it through a door in time. On the



Charles and Diana screen, you get bonus points for combining a pair of big ears with a hair-dryer. There are lots of bonus combinations like that to discover. Smash the match against the fuel, though, and you're another life down.

It's a shame the game has little of the TV show's humour, but it remains a fun time capsule from the 80s.



Above: Slide those tiles - Can you guess who it is yet?



Above: You'd think this was Prince right? Wrong! It's Mick Jagger!

arcade game without the rubbery puppets as a reference point. This game's name was changed following legal threats from the *Spitting Image* team, but publisher Domark went on to release an official *Spitting Image* game.

Split Personalities is a bit like a sliding puzzle, ex-

Sean



AMTIX was lukewarm on this game when it came out, but I've always loved it. With the tight time limit and the bombs, it's a fast-paced game that makes me sweat. The title screen music sounds like a fairground organ, and the mode 0 graphics are colourful and clear. Apart from Mick Jagger, the caricatures are easily recognisable. However, the difficulty is too hard on later levels, with more cracks in the walls and doors that never close. Has anyone ever seen Marilyn Monroe complete?

CRITICISM



SECRETS FROM THE PAST

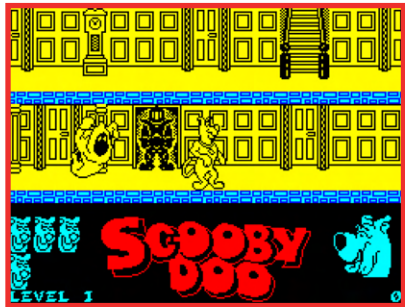
There are messages hidden in the code of some of our favourite games, like time capsules. Sean McManus does some digital archaeology.

If you ever studied the code of a Spectrum game, looking for POKEs and hints, you probably came across some hidden messages from the programmers, too. If there were bytes to spare, some coders had plenty to say.

Notes were often addressed to the hacker. “Due to the size of the game you may find it hard to hack without overwriting a part of the program with any kind of disassembler you may use,” writes Stuart Middleton in *Universal Hero*. “You will also notice there is no way any room for BASIC. To overcome this problem, you may erase the bottom part of memory (5B00-8000) as this is purely graphics. When any alterations have been made, they may be re-patched over again. I hope you have fun playing and hacking this game, and I hope you find some good pokes for it.”

He adds: “Due to the initial bad design of this program, the programmer wishes to apologise for the lack of sound which is

due to memory restrictions. To make up for this, the programmer suggests one of the Prince albums be played in the background at a large volume.” In *Bomb Jack*, Stuart recommends Purple Rain, especially on CD where “it sounds even better”.



Stuart often included a message in the games he worked on. In *Scooby-Doo* in the *Castle Mystery*, he writes: “This rather trendy loader was written for Elite by Stuart



Middleton of XCEL software, as were [the loaders for] Commando, Bomb Jack, and Ghosts and Goblins. QUESTION :- If Elite's programmers are the best in the business, why do they employ someone from another company to write their loaders?"

HACKER BEWARE

Messages addressed to hackers from other programmers weren't always so friendly. "Stop looking at my code, you twat!" says the message in Ocean's *Firefly*, while Simon Francis, author of Durell's *Critical Mass*, says: "Don't disassemble any of this you bast**ds..."

The main reason for the hostility seems to be that programmers were worried that hacking the code, and especially the loader, would make it easier to pirate the game. "Sod off and buy your own copy," says John Jones in *Soft & Cuddly*.

"If you have disassembled this code and intend making copies for resale, friends or others, bear in mind you are a thief," says the code in tennis game *Match Point*. "Four people spent a great deal of time, effort and sleepless nights producing this program. A few quid isn't much to pay."

Sometimes the tone descends into threats, from the silly to the unsettling. "Pokes can permanently bugged your Spectrum," says Martech's *The Planets*, in a feeble attempt to frighten away the kids. *Saboteur!* says: "If you pirate this the martial artists will get you!"

Moon Strike, published by Mirrorsoft, warns: "If you don't stop hacking into my sacred protection system, I will get a large axe and give you a hacking you will never forget!"

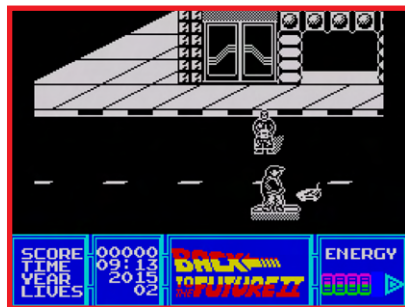
OVERWORKED, UNDERPAID

One reason the programmers are so defensive is probably that they were feeling overworked and underpaid. Steve Elward, author of budget title *Galletron*, wrote: "I did this conversion in less than two months. This in itself isn't amazing. It is, though, when you hear that I haven't ever touched a Spectrum, let alone tried to do a program on it." It must have been a challenging couple of months, presumably involving programming from another computer as a development system.

"Please don't copy this for your mates as I only get 5% of the price in royalties," he adds. "Tell them to go out and buy one, or no one will even bother to support the Spectrum any more."

Sonja Knight developed 1989 shoot-em-up *Task Force* and hid this message in the code: "My God, am I ever going to get any money out of Interceptor? I doubt it. I sold Task Force for only £800 to Interceptor. I get the feeling I was ripped..."

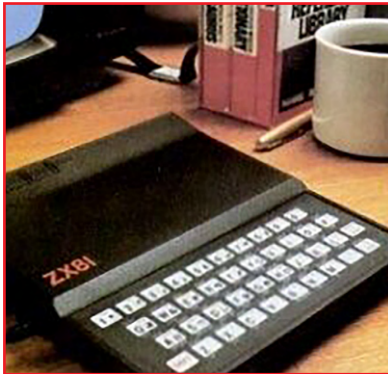
"I wonder if I'll get paid today," muses the programmer on *Back to the Future Part II*. "What do you think, Damian?"



Naaaaahhhh! The mastering hasn't been done!"

They've been pulling an all-nighter: "I don't believe it is ten past six in the morning! Yeah! And we've been typing this soddin' text in all night."

Flak was released by US Gold in 1984. There's a tiny snapshot of the work and reward involved in the project tucked away in the binary. "This program was adapted from a Commodore 64 version by Christopher Smith of Croydon, Surrey, England between September and December 1984, using an A.C.T. Apricot micro. It took approx 300 hours of programming and debugging time and bought the programmer's wife a new kitchen."



Sometimes there wasn't as much time as that. "We apologise for the game being boring but we were literally only given three weeks to write it," said Simon Brattel and Neil Mottershead, writing about *The Warlock of Firetop Mountain*. The game scored 73% in issue 1 of *Crash*, so just imagine what they could have done with an extra few weeks.

Meanwhile, the development of 1987 game *Eye* was frustrated by a lack of tools: "This program was HAND-CODED by Toni Baker, on behalf of Andromeda, who successfully managed to avoid giving me an assembler until a week before completion." Ouch! For your dedication to finishing the project against overwhelming odds, Toni, we salute you!

SCHOOL DAYS

We've heard about bedroom programmers before, but it's nice to see messages from them popping up in games. Elliot Gay wrote the Spectrum version of *Son of Bagger*, which we gave 80% for use of computer and 61% overall in issue 13 (February 1985). "Here's a bit about me," Elliot writes. "I was born on March 4th 1967. I live in Leeds and have my A levels in May 85. My first computer was a ZX81 equipped with 16K of wobble. I became superb at ZX Basic and soon tired of it due to its amazing speed (0.000000001 mph). This prompted me to teach myself Z80 machine code. Then one day I saw an advert looking for programmers to work for Alligata. I answered it, had an interview and now I'm working on a freelance basis for them."

He signs off as Elli Gay (Tor), a pun on his publisher's name. What happened to that ZX81? "I now use the ZX81 to heat my bedroom!" he writes.

Blolocop featured on the February 1989 covertape of *Unclear User* – sorry, *Sinclair User* (old habits die hard). Andy Severn and Jabba Severn were among the authors. "There's a bit of a story attached to this little gamete," Andy writes in a hidden message. "An early version of the game, with a working title of *Plurps*, was developed in a couple of days. At

```

4090H: 00 10 D0 00 7F 74 00 00
40A0H: 78 EF F6 80 00 0B 1A AA
40A3H: AA A0 00 00 41 0C 07 DD
40B0H: 55 DC 01 11 40 06 50 00
40B3H: 04 04 DB 00 70 EF 76 80
40C0H: DF FE 8F F8 00 DB 6E FA
40C3H: EA A8 00 00 90 A0 04 55
40D0H: 55 21 02 02 90 70 88 00
40D3H: 6A 72 DB 00 0F FE 8F 78
40E0H: 7F 7F FE FE 00 DB 11 AA
40E3H: AA 60 0C 02 48 14 08 D5
40F0H: 55 A4 03 C0 20 01 08 15
40F3H: D0 00 DB 00 7F 7F FE FE

```

ZX Spectrum Machine Code

that point, it had no plot or real name, but it was offered to the mags at a trade show. Sinclair User wanted it.”

“Then I HAD to finish it,” writes Andy. “After many long lunchtimes’ thought, I came up with the name BLOBOCOP, a parody of Ocean’s ROBOCOP. I had never seen the game and have been told that this bears quite a resemblance to ROBOCOP. Then, on the 23rd of December, having just got back from Southampton to collect Jabba’s new D50 synth (what a poser, huh?) I made a spectacular entrance at the pre-Xmas doo drinkies to the shouts of ‘you’re in deep trub...’ and ‘Ocean are going to sue!’”

“After a severe tongue lashing I was asked to interrupt my Xmas break



and make the gruelling journey from Cheshire to Hampshire on the 28th ‘just to fix the bits’. Well, bang go my heavy boozing sessions! Hope you have all had a smashing Xmas and a GREAT new year. I’m back off to Cheshire in a few hours to re-join the merriment!”

At least one of the messages was there to give players a better loading experience. Firebird’s *Kai Temple* contains this gem: “When my game is loading the screen buffer it makes a long whining noise due to all the 0s. So, to stop this I have inserted this really interesting message. This is my first ever machine code game. Looking back, I can see where I have gone wrong, and where the programming is not as good as it

could be. Such as the really slow sprite routine, and the scrappy detection routines. Well at least next time the programming will have improved.”

The message ends: “I dedicate this game to me, because I deserve it.”

If you’d like to dig into more hidden messages, check out *The Cutting Room Floor* (tcrf.net), where the community shared these discoveries.

Disclaimers:
 These messages only represent the opinion of whoever put them there: others involved might not even be aware of them. I’ve edited the messages for spelling, grammar, and flow in this article.

Oh, and check out my electronic music at: www.sean.co.uk/music.

Crashing & Zzapping into Your Galaxy NOW!

FOR SPECTRUM & COMMODORE ENTHUSIASTS WITH THEIR FINGER ON THE BUTTON!



ON SALE NOW



£4.99

The only magazines rampacked with vital info on Spectrum and Commodore games!

SUBSCRIBE NOW AT WWW.CRASHMAGAZINE.CO.UK

WWW.ZZAPMAGAZINE.CO.UK

NEWS REVIEWS ARTICLES COMPETITIONS HISTORY

FUSION RETRO BOOKS



Cartoons by *mch*
facebook.com/artbymch
m-harrison.co.uk

AMSTRAD ACTION



CHRISTMAS COVER-TAPES



It's Christmas! There's a new issue of Amstrad Action in the shops, and there's a tape on the cover. What a treat! Cheaper than a Mastertronic game (until Xmas '90), it was packed with type-ins, game demos, and sometimes entire games. Sean McManus looks at the software AA wrapped up for us.

Christmas 1985



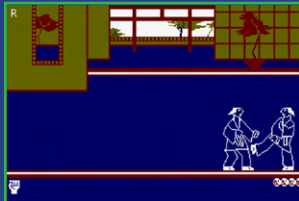
For issue #4, Amstrad Action treated us to its first Christmas covertape. The cover screamed "Mega Gift", which seemed a bit cheeky given the price went up

For you this Christmas

THE ALL ACTION COVER CASSETTE

Love, Amstrad Action and Ocean Software. Swapper or lover in every classic retail township the time comes to give. To give the person you care for something to be grateful. To give someone very attracted to you just as if he/she is not. To give someone who is not so much our readers over the leaf but we want to see you happy. Please, well, just this once, do us a favour. Buy this cassette from Ocean Software. In our humble opinion they're the only the real joy in life. We really do think you've got the standard of progress you carry proud for over the country. Sober or snobbish. Exciting action. Hours of entertainment for you this Christmas. From AA with love too.

by 50p to cover the cost of the tape. There were two



previously unreleased games from Ocean: *Kung Fu*, and *Number 1*.

Kung Fu might have had some appeal as a launch title for the Amstrad in 1984. With its four moves and lack of jumping, it seemed primitive at the end of 1985, even as a freebie. For context, that same issue

had *Yie Ar Kung-Fu* as its Mastergame. The music in



Kung Fu was atmospheric, though, in that it made you want to smash things up.

In *Number 1*, you rollerblade your way through space, collecting jewels and dodging obstacles. I'm guessing the developers had a sports sim and a space game that weren't quite working so they mashed them together.

The result? One highly original game that also doesn't quite work. The eccentricity and look of the game have a certain early-era charm, but your character blocks your view of the obstacles. The real Christmas number 1 that year was Shakin' Stevens with "Merry Christmas Everyone".

Christmas 1986



On the second AA of Christmas, my tape deck gave to me: some ads that unfolded tediously. I only loaded that bit once (and my tape's broken now), but I remember paging through seemingly endless text descriptions and screenshots of games, wondering when I got to join in.

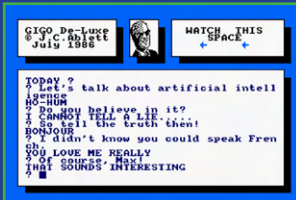
Eventually, I found the good bit: a playable level of the previous month's Mastergame, *Druid*. It's a



dungeon crawler similar to the Christmas Mastergame, *Gauntlet*. It must have irked US Gold to see their main rival on the tape in the same issue where they're reviewed.

Although *Gauntlet* remains my favourite, you spend less time wading through swarms of enemies in *Druid*, which is an improvement.

On the flip side were some of AA's best type-ins, including a database and ramdisk. *Blitter* bounced a large rotating ball around the screen, amusingly emulating a well-known Amiga demo. *Drumkit* was an excellent three-channel drum sequencer. As I write

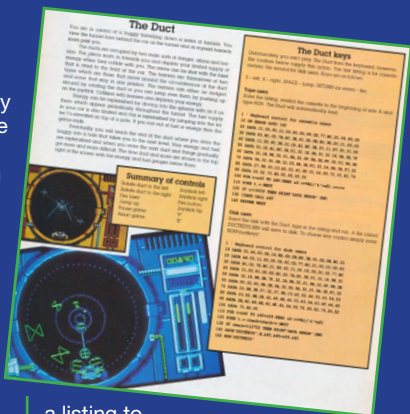


this, I can hear the default beat in my head. *GIGO* was a fun chatbot where you conversed with *Max Headroom* (see issue 8 of *AmixCPC* for more on this one). I'd already typed all of these in, but it's nice that those with lazy fingers didn't miss out.

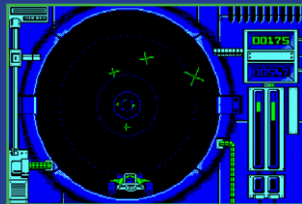
Christmas 1987



AA readers love type-ins, so let's have one to make the covertape work properly! This month's gift was a game called *The Duct*, an exclusive from Gremlin. Unfortunately, they forgot to program keyboard controls, so the mag included



a listing to get them working. In this simple but fun game, you navigate your buggy down a tube, jumping over obstacles and shooting things. The vector graphics look great when you spin the tube. (If you download it to try this Christmas, the keys are Q, E, Space, Fire).



This tape also included type-ins. *Thoughtlink* was a tool for organizing ideas, and there was another database to store your Christmas card list. As long you didn't know more than 125 people, that is.

The best type-in here was *Sound Digitizer*, which enabled you to sample audio tapes and replay them in your BASIC programs. Oh, I had so much fun with this. I created remixes of *Crazy Nights* by Kiss and *Doctorin' the House* by Coldcut and Yazz. It ate up all the memory, but I could loop the repetitive bits and get nearly

a minute of scratchy audio together.

Christmas 1987



What does a barber totally do? *Total Eclipse*. Never mind.

For Christmas 1988, AA gave away a demo version of the third Freescape game, *Total Eclipse*. Freescape was a technique for creating solid 3D worlds, viewed from a first-



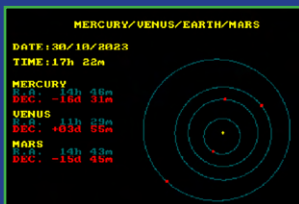
person perspective. It was graphically astounding. My favourite feature in *Total Eclipse* is that you can press a key to have a nice rest and slow your heartbeat.

Just what I needed after battling through Woolworths for the Christmas shopping.

The other side of the tape had some new



reader programs. *Isotopes* was a colourful and fast version of *Snake*. *Solar System* plotted the position of the planets for a given time. *Gibberish* spouted nonsense sentences, such



as "The flavoursome banana jestingly howled with the stupid running shoe." No idea why that program exists.

It wouldn't be Christmas without a new database, of course. You use this one to keep track of AA reviews, compiling your own index and noting important comments. Only two people ever thought that was a good use of time: the person who wrote the program, and the person who put it on the tape.

Christmas 1989



Dear Santa. Here's a letter I wrote using my new word processor from the AA Christmas tape."

Also in this month's selection box were a program for playing crosswords, and a program for avoiding them. That second one caught your errors in type-ins. *Gunslinger* gave you a two-player clone of cowboy shooting game *Boot Hill*.



The main attraction this year was Level 3 of *The Untouchables* by Ocean, based on the film of the same name.

Outside of sports sims, there aren't many games like this where you play a real, named person. In this case, it's Eliot Ness, an American agent trying to bring down gangster Al Capone. In this free level, you must shoot the baddies as they appear at the windows trying to shoot you. The game looks and sounds first class, but the targeting is slow, making it frustrating.



Christmas 1990

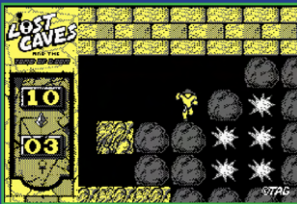
Lots of fun on this tape! First up, a demo of *Puzznic*. It's a pattern-matching puzzle game with gravity. Nowadays, there's an app for that. This playable demo starts off



teaching you the game mechanics but soon gets tricky.



While last year saw the third film in the original Indiana Jones trilogy hit cinemas, this Christmas saw a game inspired by the character on the covertape. *Lost Caves* was written by AA reviews editor Adam Waring before he joined the



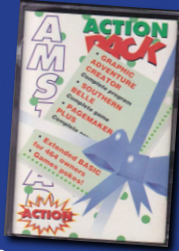
mag. It's a nicely presented *Boulderdash* clone, but with level designs that make it too easy to get trapped. Progress relies heavily on trial and error and replay.

Space Froggy sees our amphibian hero tackling the 6128 ROM shortage by collecting chips. It is a fast and colourful platform game, and a great showcase for Glenco's *Sprites Alive* package and BASIC compiler. This would have made a good budget game with a big enough map.



Christmas 1991

By now, Amstrad Action carried a tape every issue, so what would they do for Christmas? They led with *Graphic Adventure Creator*, which enabled readers to write adventure games without a deep technical knowledge. I remember being quite pleased to see it on the tape at the time, and I ordered the manual.



For most readers, it was probably an odd choice for a Christmas tape when you want a simple slice of fun to go with your turkey sandwiches. *Pagemaker Plus*, too, probably didn't tear anyone away from the Queen's speech.

The only game on the tape was *Southern Belle*, an AA Rave in November 1985. It enables you to



live the boyhood dream of generations past as you drive a steam train from London to Brighton. Working out how to use the cut-off, regulator, and vacuum brake is a challenge, but many of those who persevered loved the game. It's relaxing to watch the train puff away

on demo mode, and I was pleased to see the iconic Battersea Power Station when I pulled out from London.

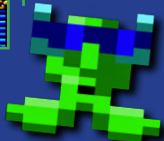
Around this time, other AA tapes featured more accessible games such as *Firelord*, *Impossaball*, *Tir Na Nog*, *Shockway Rider*, and *Sphinx Jinx*. If I'd been curating this tape, I'd have included one of these at Christmas to pump up the fun factor.

Christmas Future

Amstrad Action featured tapes every issue from here on until the magazine closed. In 1992 and 1993, there was no Christmas-themed issue, just business as usual, although *Elite* was a great covermount for '93. In 1994 there was a fun Santa cover, but the tape was rather dry: a demo of *Desktop Environment System*, a DOS copy utility, and some public domain games.

Through its monthly tapes, AA did a great job of getting classic games and powerful software into wider circulation. Considered like that, AA was the last budget software label for the Amstrad.

For subscribers, the covertapes were a lucky dip. You never knew whether you'd get a much-loved game like *Chuckie Egg* or a utility like *Tasword* in the post. Christmas is a time for surprises, so let's raise a glass and celebrate the joy of covertapes past!



AMTIX! CPC

FLASH BACK

For our second 'Flash Back' this issue, Sean McManus revisits a budget classic gem with Mastertronic's much-lauded *Finders Keepers*.

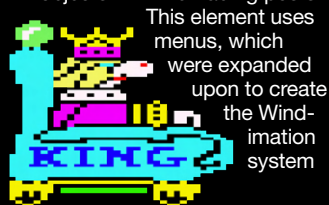


Can you help Magic Knight become a member of the Polygon Table? To do so, you must either escape from the castle or collect as much treasure as possible.

Finders Keepers is a splendid mash-up of several gameplay ideas. At first glance, it's a platform game, packed with enemies to avoid. But there are two top-down mazes to



explore as well, with objects strewn throughout. Also in the mix is a puzzle game, based on combining items to make new things, and trading objects with the trading posts.

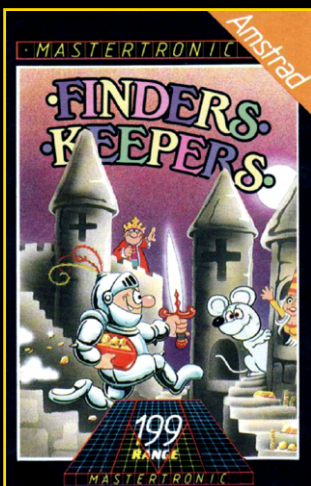


for Magic Knight's later graphic adventures (starting with *Spellbound*).

To enjoy *Finders Keepers*, you'll need to be comfortable with experimentation. You'll spend a lot of time combining objects to see whether they do anything interesting together. There's no obvious reason why the sage's stone and the lead bar should make a gold bar, but they do. Objects combine automatically if you're carrying both, but you aren't told when it happens, so you need to keep an eye on your inventory. There are lots of red herrings in the game (did you think that hairy dog would just frighten the cat away?).

A maximum of five items can be carried (of any size). You'll need to manage your inventory well and keep track of what you see where as you explore the castle. You can, at least, drop items anywhere to pick up again later.

Released by Mastertronic in 1985, *Finders Keepers* was one of the first games to show that budget titles could be high quality. Across all 8-bit formats, the game sold more than 330,000 cop-



ies. Although this game also appeared on the Spectrum, the graphics on the Amstrad were recreated in Mode 0, making them bold and colourful. There's a



jolly medieval-style tune to keep you going.

Sean



Finders Keepers certainly gave you value for your £1.99. The presentation is excellent, and the game feels large, although there are only about 20 rooms surrounding the mazes. There are some frustrations. Positioning Magic Knight to jump between platforms can be hard, and the nasties feel unavoidable.

Overall, the game works well, though. The multiple endings are a nice twist, and the platform rooms and mazes make a fun world to explore.

The object puzzles are a bit random and require a lot of trial and error. MK went on to more logical adventures in later games, but nothing with the pace and energy of *Finders Keepers*.

CRITICISM

GOODBYEEEEE!



As we publish our final issue, Sean McManus looks back at the life of the Amstrad CPC, as seen through the final issues of the major magazines.



AMTRAD
Issue: #18
April 1987

Readers must have wondered whether it was a joke. After 18

months, AMTRAD, the sparky games-focused title, was to be merged into Computing with the Amstrad, the serious magazine famous for its programming tutorials and listings. The planned publication date for the first merged issue? 1st April 1987.

Reviews in the 84-page, final issue of AMTRAD included *The Sentinel* ("98% - the best game ever"), *The Growing Pains of Adrian Mole* ("29% - for the asking price, extortionate!"), although that didn't stop Amtrid running a competition to win it), and *The Pawn*

("98% - probably the finest adventure released to date").

The results of the readers' poll, looking back at 1986, were in. *Spindizzy* won best game, *Get Dexter* best arcade adventure, and *Lightforce* won best shoot 'em up. Runners-up in various categories included *Batman*, *Gauntlet*, *Heartland*, *Ikari Warriors*, *Hitchhikers' Guide to the Galaxy*, and *Elite*.

The last page in the last issue was given over to a preview of *Head over Heels*. "We won't be around when this finally arrives on the streets, but don't hesitate to grab it," said



Right: *The Sentinel* by Firebird, which was awarded a whopping 98%!



Above: It's a fond farewell from the Adventure pages.

Below: A sneak preview of Jon Ritman's masterpiece, Head Over Heels



Richard Eddy. "It looks like this is going to knock the 3D games world for six."

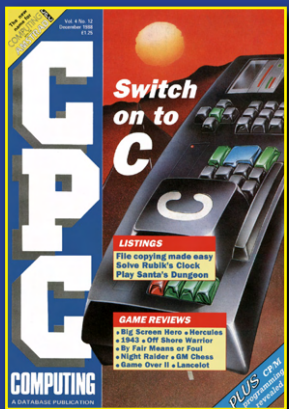
Although Amtix had a short run, it was perfectly timed. It launched in November 1985, when programmers had started to take the machine seriously and good

games were coming out, such as *Highway Encounter* and *Alien 8*. By the time of the final issue, many of the best games for the machine had been released.

COMPUTING WITH THE AMSTRAD

Issue: #48

Dec 1988



When the May 1987 issue of Computing with the Amstrad came out, there was a small AMTIX! logo on the front. That's when we knew for sure the merger was no joke.

In some ways, the marriage of the magazines was a great idea. Computing with the Amstrad previously carried game reviews, but the pictures were small and the layout was drab. AMTIX! arrived, bringing a much-needed splash of colour. The game reviews looked better and Am-Tips introduced POKEs to the magazine for the first time.

For AMTIX! readers, it wasn't the same, though:

Amstrad Action was probably a more natural home for them. Oli's art was no more.

August 1988 was the last issue of Computing with the Amstrad before it rebranded. It had 64 pages, and published tutorials on animating 3D graphics, building your own button-based game controller, and speeding up machine code. There was an epic two-and-a-half page listing called *House of Spiders*.

There were quite a few adverts for serious software, but the Amstrad market was already showing signs of decline. Although Ocean's *Karnov* won the final AMTIX! Accolade, Gallup's games chart was almost entirely



made up of budget titles, including reissues of *Way of the Exploding Fist*, *Ghostbusters*, and *Aliens*.

The following month, Computing with the Amstrad became CPC Computing, and all mentions of Amtix were gone. Inside, the mag wasn't that different otherwise. Perhaps the rebrand was to make clearer which Amstrad computer series they were catering for, now that there were multiple machines on offer? It was a plus that the new logo was vertical, so it would stick out

when the magazines were shelved overlapping in WH Smith.

The letters page revealed that 70% of readers had a disc drive and more than half had a printer. Most surprisingly, more than 1,000 people typed in a listing from the January issue of CWTA and sent in a competition entry based on it, suggesting a healthy



December 1988 issue of CPC Computing was to be the last, with the magazine being acquired by Amstrad Computer User. The last issue included a program to solve the Rubik's Clock puzzle, a disc copier program, and reviews of *Game Over II*, *The Games – Winter Edition*, and Level 9's *Lancelot*.

There were tutorials on CP/M and the C programming language, plus a long listing for a game called Santa's Dungeons. Your goal is to "save Santa from being blown up, stabbed in the back, and starved to death". I expect complaints were sent, but there was no follow-up letters page for them to appear in.

At the end, the magazine felt much like it



AMSTRAD COMPUTER USER

Issue: #90
Dec 1992

"The Amstrad CPC scene is as big as ever," claimed Amstrad Computer User (ACU), even as it announced its own demise in the May 1992 issue. The reason it had to close was that Amstrad had withdrawn its support for the CPC and shuttered the official magazine, too.

Amstrad Computer User was like Computing with the Amstrad in its commitment to type-ins. In the final issue, there was a meaty three-page game listing to get your fingers into, along with six pages of 10-liners. Other features covered MicroDesign, the PD scene, and on-line community CompuServe, which was accessed using a modem. The final issue felt light on technical articles, but if you loved Dizzy, there was a map and three-page guide to *Prince of the Yolkfolk*.

Surprisingly, the Amstrad games market still represent-

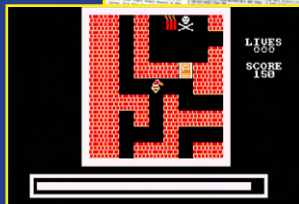


Above: Queen Gremla's back in the much anticipated *Game Over II* by Dynamic.



and engaged readership.

The rebrand was to be short-lived. The



Above: Can you save Christmas by helping Santa escape the dungeons?

did in the early days, with a focus on programming and serious software, lovely long listings, and comprehensive coverage of adventure games, including how to write them.



BOMB ALLEY

Super Seymour

Did you know? If you want to keep Seymour safe from the Phantom!

ROUND UP
of Seymour
Sales: Price: £3.99
30% 78%

ed 7%
of unit
sales at

this time, ahead of the Mega Drive, Nintendo, and Game Gear. But it was trailing the Spectrum (14%), Amiga (20%), and Commodore 64 (26%). By now, the Amstrad games market was all about budget titles and reissues, with budget titles typically costing £3.99. There wasn't much money around, although ACU still carried quite a few ads for mail-order companies. Romantic Robot's advert said they were leaving the 8-bit market and selling Multiface Twos at the knock-down price of £34.95. That seems quite pricey, looking back now.

Game reviews in this final issue of ACU included Code-masters' *Super Seymour* and budget re-releases of *Rainbow Islands*, *The Last Ninja*

2, *3D Pool*, *Quattro Fighters*, and *World Cup*. The only full-price title (£15.99 on tape, £19.99 on disc) was Domark's TNT 2, a compi-

pumped full of lead!" she said on the subs ad. All this "guns are kewl!" energy felt mismatched with the mag's great technical articles, including assembly language listings, which rarely appeared in other titles.

In the launch issue, there was a roundup of the Top 50 games, with *Lemmings*, *Prince of Persia*, *Smash*

BOMB ALLEY

TNT 2

Custom software of its kind level design enough to take a hit?

ROUND UP
of TNT 2
Sales: Price: £15.99
86% 82% 88%

Above: An explosive compilation! TNT 2 from software house Domark.

lation that included *Hydra*, *Robot Monsters*, *Skull & Crossbones*, *Badlands*, and *S.T.U.N. Runner*.

Not to be beaten, ACU publishers HHL Publishing launched unofficial mag CPC Attack the very next month to continue serving their readers.

CPC ATTACK Issue: #06 Nov 1992

Today, I learned that using an apostrophe instead of REM in a program takes more space in memory. Where did I learn this? The debut issue of CPC Attack, no less.

The magazine came crashing into newsagents in June 1992, with heavily armoured mascot Amy Strad popping up through-out. "Subscribe now or be prepared to have your butt

TV, *Rick Dangerous*, and *Rainbow Islands* topping the chart. It seemed like they'd missed some early classics, but they'd only included games that were

CPC ATTACK!

THE GREAT BUYERS' GUIDE

SCREENTEST GIVEN
20 GAMES REVIEWED

ERABIT!

DO YOU WANT TO FEEL THE WAY YOU FEEL? YOU WANT TO FEEL THE WAY YOU FEEL?

PLUS: ANATOMY OF A PROGRAMMER

still available at the time of publication, making for a great buyers' guide.

At times, it seemed like the mag didn't understand

Next Month
prepare yourself for...
CPC ATTACK!

AMSTRAD COMPUTER ACTION AT THE EDGE

ATTACK! challenges all aspects of home computing... from design, through to programming and hardware that...

ATTACK! will do the best CPC magazine for games... you will want to buy!...

BUY... NOW! Buy now or be prepared to have your butt...

SEND YOUR... NOW! Send your...

CALL... Call...

THE... The...

YOU NOT TO HANDLE...

Left: ACU may be over but prepare yourself, because CPC Attack is coming!

its readers. The free gifts, including posters and 3D glasses to view them, felt like the kind of thing you'd see on a kids' comic. The best games were given a Bogus Award, inspired by the 1991 film *Bill & Ted's Bogus Journey*, which made the editorial team look out of touch: Bogus was slang for bad, not good. A worse crime still: The launch issue dedicated five of its 68 pages to a feature about consoles, which felt like they'd given up on the Amstrad before they began. There were console features up until the survey results were in, showing readers hated them.

Amy was not the only member of the Stradivarius family to fea-



Above: Don't Mess! Or you'll have Amy Strad to answer to.



Above: The bold and colourful review pages of CPC Attack.



Above: See you next month! Erm, no, no you won't...

ture in the magazine. The serious articles, which were still plentiful, featured someone who looked like her spoddy brother, with thick glasses, zits, and a gormless expression. It was not an affectionate portrait of the readers, many of whom had had their Amstrad Computer User subscription transferred to CPC

Attack.

The "Dad sez" and "Mum sez" boxes on the reviews had them saying things like, "Destructive games shouldn't be allowed – do your homework!" It's as if the editorial team thought games were only for kids and didn't realise that many ACU readers were the dads and mums they were laughing at.

The magazine boldly claimed it had attracted new readers and didn't need audited circulation figures to prove it, but its run only lasted six issues. The final edition had a teaser for the next issue, so the axe fell fast and Amy never knew what hit her.

The last issue was 60 pages, compared to AA's 70 pages in the same month. The cover feature was about image processing using the Vidi Digitiser, Dart Scanner, and Videomaster hardware. There was an interview with Steve Denson from SD Microsystems, and a report from the second European demo party. Reviews were dominated



by football games, including *Psycho Soccer* (93%), and a Halloween roundup with

The Addams Family (94%) and Cauldron (72%) leading the list.



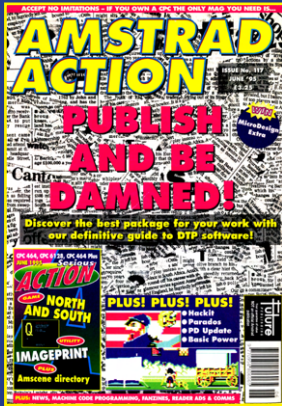
There were four pages of listings, including assembly language source code for a disc formatter. The magazine billed them as “our code-mungus collection of rebellious routines.”

At the time, the design of CPC Attack put me off a bit. Looking back, there is much to love. The programming coverage is excellent, and it is nice to see a publisher investing in original artwork for each issue, like AMTIX! had.



AMSTRAD ACTION

Issue: #117
Nov 1995



At the time of CPC Attack’s closure, Amstrad Action (AA) was publishing an audited circulation of 35,298. There was clearly life in the CPC market yet. And Amstrad Action went on to prove it, surviving nearly three more years, and lasting nearly ten years in total. Astonishing.

The secret to surviving so long? Probably the covertape. Starting with April 1991, every issue gave away games and utilities. These included highly rated titles such as *Elite*, *Exolon*, *Uridium*, *Zynaps*, *Dizzy*, and *Ikari Warriors*. For software houses, this was their last chance to get paid



(by AA) for games that nobody was buying any more.

The other secret to surviving so long was a committed readership. AA had an audited circulation of 15,168 for its final issue. It’s much fewer than the 27,090 readers it had two years earlier, or the 38,457 readers in 1988. But it showed there was still a sizeable community of people interested in the Amstrad, and willing to pay for a magazine about it.

Nevertheless, the 8-bit market was coming to an end. Issues of AA appeared without a single commercial game review. In AA’s final year, there were only four: *Masters of Space* (75%), *Black Jack and Cribbage* (64%), *Who Said That?* (85%), and *Super Bomberman* clone *Mega Blasters* (94%). The adventure scene was healthier, with *The Examiner*



columnist Debby Howard

reviewing seven new games in AA’s last year.

There wasn’t much money for advertising in the Amstrad market, so the page count dwindled. A year before AA closed (June 1994), it was 52 pages. The next month (July 1994), it dropped to 36. In November 1994, it was down to 24 pages.

AA had always champi-

oned reader contributions, especially through listings, but now the community really was the lifeblood of the magazine.

There were columns for public domain software and fanzines, both celebrating the work of hobbyists who were unlikely to make much money from their endeavours. The editorial urged readers to buy the things on offer. “Be one of the people who supports the CPC, not one of those who asks where all the support has gone,” it said.

The final issue of AA had American Civil War simulation *North and South* and a directory of Amstrad contacts on the tape.



Inside the 24-page mag, there were just three adverts, apart from the publisher’s own. Radical Software was advertising

RADICAL SOFTWARE

DOOMED!

AMSTRAD: 0784 1200 ext 6
 This will only run on the British Isles. We'll up to date most computers to 1000K. Super fast loaded shooting action and you can download games from our website. Contact us for more CPC information please.

119.95 128.95
 This will only run on the British Isles. We'll up to date most computers to 1000K. Super fast loaded shooting action and you can download games from our website. Contact us for more CPC information please.

023 95 3006 £13.95 (incl)

The CPC's greatest desktop system is fast becoming obsolete. It's time to get on with the future. A full desktop system, with 666 megabytes of RAM.

This is a collection of programs which run on the 666 operating system. Includes: Amstrad Desktop, Amstrad Forum, Amstrad Mail, Amstrad News, Amstrad Online, Amstrad Plus, Amstrad Plus Chat, Amstrad Plus Chat, Amstrad Plus Chat.

TITLE	PRICE	TITLE	PRICE
WHO SAID THAT	£13.95	UNDESIRE ADVENTURE	£9.95
AMSTRAD PLUS ONLINE	£19.95	HASTINGS 2000	£9.95
AMSTRAD MAIL	£19.95	STAR DRIBBLE	£9.95
AMSTRAD NEWS	£19.95	AMSTRAD ONLINE	£19.95
AMSTRAD PLUS	£19.95	AMSTRAD PLUS CHAT	£19.95
AMSTRAD PLUS CHAT	£19.95	AMSTRAD PLUS CHAT	£19.95
AMSTRAD PLUS CHAT	£19.95	AMSTRAD PLUS CHAT	£19.95

AMSTRAD: 0784 1200 ext 6
 This will only run on the British Isles. We'll up to date most computers to 1000K. Super fast loaded shooting action and you can download games from our website. Contact us for more CPC information please.

an upcoming *Doom* clone called *Abandoned*, along with *Fluff*, *Prehistorik II*, and *Super Cauldron*. That *Doom* clone sadly never appeared.

There was a page for Cheat Mode, including tips for *Biff* and *Chuckie Egg*, which had both been on the cover tape. Basically BASIC showed you how to create a string that contained machine code and call it, which was a quirky idea. The final cover story was about desktop publishing, with the headline “Publish and Be Damned!”

There was a sense it was becoming harder to create AA. The covertape had been stressful to put together, the mag hinted. After four years, it must have been difficult to track down good software that hadn't been on the tape yet. Republished content in the final issue included game tips and a hardware tutorial in Techy Forum. Disc operating system *Parados* was reviewed for the second time. It was hard for many people to find information in old issues, so perhaps there was some value in this. But it also seemed like there wasn't much new to say.

The magazine included a couple of teasers for the next issue, so there was no inkling it was the end. Tutorial series were cut short, incomplete. As the last mag for its machine, it's a shame Amstrad Action didn't get a chance to say goodbye properly, as Your Sinclair and Commodore Format did so magnificently.

AMTIXCPC

Issue: #12

June 2024



Amstrad Action, it turned out, was not to be the last regular CPC magazine after all. Amtix was revived in Summer 2021, with the blessing of founders Roger Kean and Oli Frey. Since then, we've published twelve issues and an annual, totalling about 750 pages. The games we've reviewed show the true potential of the Amstrad, and the brilliance of the CPC community, which goes on.

And so, the first Amstrad magazine to close was also the last. Thank you for reading. It's been a blast!

DID YOU KNOW?

My game *The Further Adventures of Fred* was due to appear on the tape of the axed issue 118 of *Amstrad Action*. If you're curious, you can play it online at:

<https://tinyurl.com/4z5emj8s>

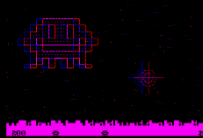


Left: We're all Doomed! An advert for Radical Software's CPC *Doom* clone *Abandoned*.

Listings

```
10 PRINT "Amstrads are the best!!!"  
20 GOTO 10
```

You can now play these games,
originally published as
listings, in your browser:



Anaglyph
Aliens



BAS Invaders



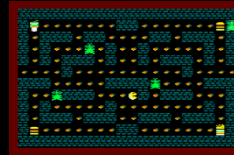
Fishtank



Alien
Intervention



Wallbuster



Paclone



3D Mapper



Chords
Tutor



Gribbet

[Download my disc file here](#)

Behind the listings

In this article, based on an interview I gave to Computer Shopper in 2019, I discuss the background to my listings and what I did to help them get published.

While I was still at school, I had several listings published in Amstrad Action and Amstrad Computer User magazines.

It was a huge buzz to see my programs in print. Often, that was the first feedback I'd get on them: that an editor considered them good enough to share.

I used to enjoy typing in the listings from magazines, but I was never really sure how many people (if any) were using mine. A couple of readers did write to me about them, though, and years later I met people who had typed some in. It was nice to know they had found an audience.

The money was good too. I was a teenager with a paper round and a couple of Saturday jobs, working on a market stall for a pound an hour and mowing a garden for a fiver. My effective hourly rate for writing some listings was much higher than I was earning elsewhere. ACU paid £50 per page, I think.

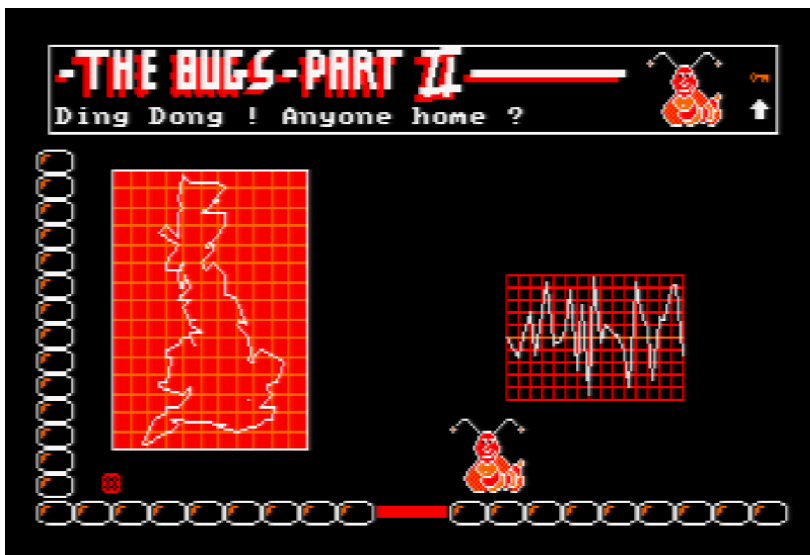
I believe I developed a good understanding of what the magazines were looking for. Basically, it had to be worth the effort of typing it in. AA preferred short listings that were fun, useful or impressive. ACU favoured epic multi-page game listings, which gave you a lot of scope to do something graphically interesting, with some music, and a bit more playability to it.



ACU also had a section dedicated to ten-line programs. I think the key to a good 10-liner was having an idea that you could naturally express in 10 lines, rather than trying to squeeze a big idea to fit. The best 10-liners taught you something new or provided a small but focused utility. My favourite one that I wrote was a Piano Chords tutor. It demonstrated a technique called colour swapping to highlight the piano keys, but also provided a useful program for budding Jean Michel Jarres. Others I wrote did things such as save the screen contents with all its colours and other attributes; and slow the computer down to help with debugging.

Listing ideas often emerged from my other programming projects. I created huge game programs that almost nobody saw, including The Bugs, shown below. The most interesting parts to other people were the tools I made to build them, and some of the core routines that they could use to make their own games.

One example was Easi-Sprite Driver (ESD), featured in AA. It added new commands to BASIC so that you could move sprites around the screen and make more colourful games. I sometimes came across games that were written using it, and it was exciting to know I'd played a part in helping make those games.



I only had one listing rejected, which was a rather slow fireworks demo. Thinking back, it wasn't very good. My programming skills and, equally importantly, my ideas were much sharper a couple of years later, when I started getting published more often.

I tried to make it easy for the editor to publish my work: I tidied up the line numbers, sent the program on disc, and provided a clear summary and instructions. I imagine for a busy editor these things made it easier to choose my listing over something arriving on tape with a post-it note.

When deciding what to code next, I found myself thinking of programs that editors might publish. It was good to know that there might be an audience for what I was coding. I spent a lot of time developing massive games that were creatively satisfying but of little interest to others. The type-ins gave me a chance to get my work in front of other CPC fans.

Ultimately, writing listings helped me establish my career. The listings gave me an opportunity to write a couple of tutorials and a book, and that enabled me to pivot into other areas of journalism. Today, I write [books](#) and [articles](#) to inspire children with coding, using [Scratch](#), Python and [Raspberry Pi](#). My book [Mission Python](#) is a listing dissection, with readers building a space adventure game, chapter by chapter. I hope that they feel the same excitement I had when I programmed the CPC in the 80s.



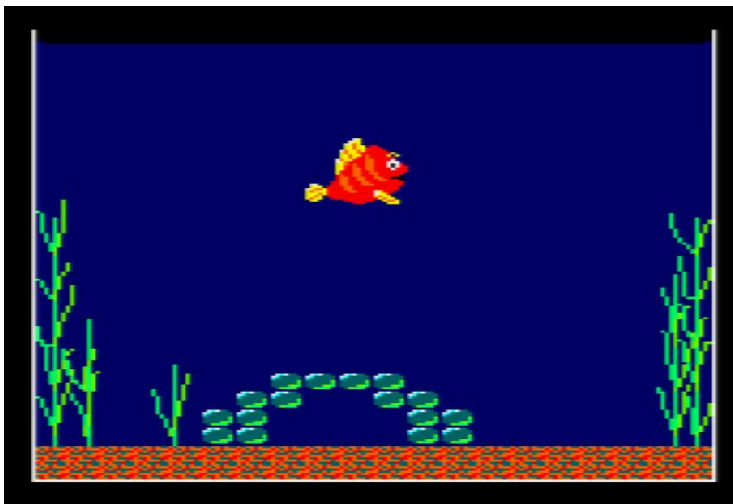
Looking back now, it would have been nice to have realised more type-ins and sent them in. I could come up with about a dozen ideas now that I could put together quickly. My school and social life were busy, though, and I'm pleased that I enjoyed all the programming projects I did do.

When I recently created listings for Amstrad Addict and Amtix CPC, I considered what kind of program would be interesting to read and read about. We can easily download any software now, so there isn't much point in a listing dominated by unintelligible hexadecimal data.

I'm surprised at how much better I am at programming the machine today, despite the fact I rarely touch it. I find it easier to solve programming problems and create effective and interesting code. I understand the importance of presentation and a good user experience in a way that I couldn't as a kid. I've updated many of the listings on my website to feel more polished, with instructions, more intuitive keys, and tidier transitions and screen layouts.

I learned a lot from early issues of ACU and Computing with the Amstrad, and I have happy memories of playing with type-ins from AA. I hope that readers found something useful or entertaining in my listings, too.

If you want to try any of them now, you can play many of them in your browser or download my disc file at [my website](#).



EYE-POPPING 3D EFFECTS ANAGLYPH ALIENS TYPE-IN SPECIAL



If you think shoot-em-ups are a bit shallow, this will change your mind. It's a listing by Sean McManus that uses 3D glasses to make the aliens pop out of the screen.

They're Coming for you!

The aliens emerge in the distance and fly towards you. Your mission? Blast them down. If three get past you and land, it's game over.

You'll need 3D red/blue glasses (called anaglyph glasses) which you get with comics. Alternatively, you can buy them in fancy plastic frames for a couple of quid on eBay.

The 3D effect works because the red lens lets red images through and blocks blue, while the blue lens lets blue through and blocks red. Because each eye sees a different view, there is a sense of depth. The game also works with the older red/green glasses. Just change the ink2 variable in line 50.

I'm not aware of any anaglyph games on the Amstrad. CPC Attack gave away 3D glasses

on its launch issue, but only for viewing the Lemmings poster inside. Mike Singleton wrote 3 Deep Space for the Spectrum and other machines, but Crash (issue #1) rated it poorly for both the 3D effect and the gameplay.

In October 1983, C&VG had a 3D issue with listings for the Spectrum,





BBC, Atari, and Commodore. These are fascinating experiments but appear to be hampered by the limited colour

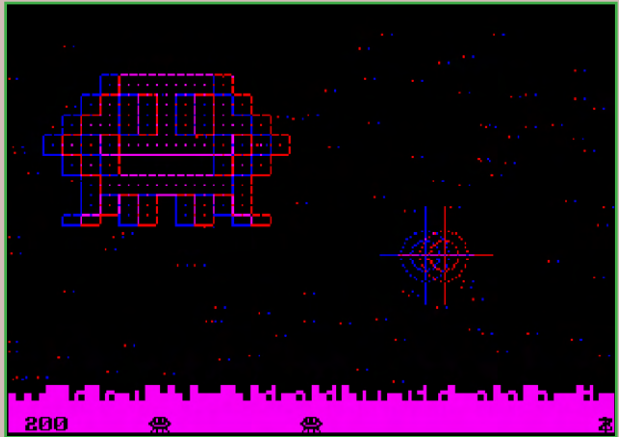
palettes on the computers.

I set out to see what's possible with Amstrad BASIC. Using Winape, the Amstrad can match the red, blue, and green colours used by 3D glasses really well. For best results, run this program there. On a real Amstrad, you might need to adjust the brightness, which you can do while the aliens are being drawn. The 3D effect works well with the distant aliens, but colour leaking becomes a problem as they get closer.

The alien images are created in BASIC by enlarging two redefined characters (UDGs). I then use a tiny machine code routine to turn them into sprites. I wish I'd had the idea of drawing sprites in BASIC when I was writing listings in the 90s. It enables this game, for example, to have more than 10K of sprite data, with huge images, although there is very little hex data to type in.

Give it a try!

You can download the code and try the game in your browser at: <https://tinyurl.com/2r4up4y5>



Above: Fancy a game with a little more depth? Then look no further than Anaglyph Aliens!

```

10 REM Anaglyph Aliens - (C) 2024 Sean McManus - www.sean.co.uk
20 REM ----- SPRITE CODE - ESD2 - 1993
30 MEMORY 9999:FOR mem=4010 TO 40221:READ a$:a=VAL("&"+a$):PRINT:POKE
mem,a:NEXT:DATA DD,6E,0,DD,66,1,DD,5E,2,DD,56,3,CD,1D,BC,E5,DD,5E,4,DD,56,5,EB,
4E,23,46,23,EB,E1,C5,E5,1A,AE,77,23,13,10,F9,E1,1,0,8,9,30,4,1,50,C0,9,C1,D,20,E8,C9
40 DATA DD,6E,4,DD,66,5,DD,5E,6,DD,56,7,CD,1D,BC,EB,DD,6E,8,DD,66,9,DD,4E,0,DD,
46,2,71,23,70,23,C5,D5,1A,77,23,13,10,FA,D1,EB,1,0,8,9,30,4,1,50,C0,9,EB,C1,D,20,E7,C9
50 ON BREAK GOSUB 480:MODE 1:PAPER 0:OPEN 1: BORDER 0:ink2=2:INK 0,0:INK 1,0:INK
2,ink2:INK 3,6:REM If you have green glasses change to ink2=9
60 SYMBOL 250,7,13,29,55,24,15,10,18:SYMBOL 251,224,176,189,236,24,240,80,
72:SYMBOL 255,48,254,22,60,116,210,254,16:a$=CHR$(250)+CHR$(251):LOCATE
1,25:PRINT a$:CHR$(23):CHR$(1):WINDOW 1,40,1,24:PRINT SPACE$(10):a$:
" Anaglyph Aliens ";a$
70 PRINT SPACE$(3):CHR$(169):"2024 Sean McManus - www.sean.co.uk"
:PRINT:PRINT"Near red/blue 3D glasses to make the".PRINT"aliens pop out!
Cursor keys move, Space fires. You must hit them in the centre. It's their
only weak spot. Good luck!"
80 LOCATE #1,39,25:PRINT #1,a$:WINDOW 1,40,10,24:INK 1,8:addr=10000:sprite=1:
size=2:GOSUB 370:FOR size=4 TO 10:GOSUB 370:NEXT
90 LOCATE #1,8,25:PRINT #1,SPACE$(3):"Arming defence forces":SPACE$(6):PLOT
320,250,2:DRAW 0,-100:PLOT 270,200:DRAW 100,0:PLOT 340,250,3:DRAW 0,-100:PLOT
290,200:DRAW 100,0
100 DEG:FOR a=0 TO 360 STEP 10:PLOT 320+25*SIN(a),200+25*COS(a),2:PLOT
320+15*SIN(a),200+15*COS(a),2:PLOT 340+25*SIN(a),200+25*COS(a),3:PLOT
340+15*SIN(a),200+15*COS(a),3: SOUND 2,a,1,15,0,0,1:NEXT:CALL 40164,addr,135,125,
65,50:sights=addr:addr=addr+3300
110 DIM r1(360),r2(360):FOR g=1 TO 360 STEP 10:r1(g)=RND*20:r2(g)=RND*20:NEXT:
INK 1,0:INK 2,8:INK 3,8
120 MODE 1:FOR i=2 TO 3:FOR g=1 TO 360 STEP 20:o=275+(i*20):PLOT
o+(20*SIN(g)),200+20*COS(g),i:DRAW o+(50+r1(g))*SIN(g+5),200+(50+r2(g))
*COS(g+5),i:DRAW o+(20*SIN(g+10)),200+20*COS(g+10):NEXT g,i:CALL 40164,addr,
115,145,45,90:ERASE r1,r2:bang=addr
130 REM ----- GAME BEGINS

```

Program listing continued on next page

```

140 MODE 1:LOCATE 1,23:PEN 1:blgd$=" "+CHR$(143)+CHR$(140)+CHR$(148)+CHR$(132)
+CHR$(141)+CHR$(150)+CHR$(142)+CHR$(156)+CHR$(157):FOR x=1 TO 40:PRINT
MID$(blgd$,INT(RND*10)+1,1):NEXT
150 PAPER 1:PEN 0:WHILE 1:LOCATE 1,24:PRINT SPACE$(79):CHR$(255):INK 1,8:INK
2,0:INK 3,0:FOR g=1 TO 100:x=RND*640:y=RND*350+50:PLOT x,y,2:PLOTR -10,0,3:
NEXT:INK 2,ink2:INK 3,6
160 score=0:landed=0:x=160:y=100:CALL 40110,sights,x-25,y+33:GOSUB 300:GOSUB
320:EVERY 20,1 GOSUB 340:WHILE landed<3:oldx=x:oldy=y
170 IF INKEY(2)=0 THEN y=y-20
180 IF INKEY(0)=0 THEN y=y+20
190 IF INKEY(8)=0 THEN x=x-20
200 IF INKEY(1)=0 THEN x=x+20
210 IF INKEY(47)=0 THEN GOSUB 450
220 IF x>280 THEN x=280 ELSE IF x<20 THEN x=20
230 IF y>160 THEN y=160 ELSE IF y<40 THEN y=40
240 IF x<>oldx OR y<>oldy THEN CALL 40110,sights,oldx-25,oldy+33:CALL
40110,sights,x-25,y+33:SOUND 2,x+y*20,5,15,0,0,8
250 WEND
260 WINDOW #2,15,27,12,14:PAPER #2,0:CLS #2:TAG:PLOT -50,-50,2:MOVE 250,204:
PRINT"GAME OVER":PLOT -50,-50,3:MOVE 260,204:PRINT"GAME OVER":PLOT 224,
174,1:DRAWR 208,0:DRAWR 0,48:DRAWR -208,0:DRAWR 0,-48:TAGOFF
270 FOR h=1 TO 6:FOR g=350 TO 500 STEP 25:SOUND 1,g,h*2,15:SOUND 2,g*2,h*2,
15,0,0,8:SOUND 4,g/2,h*2,15:NEXT g,h:SOUND 1,956,150,15:SOUND 2,239,150,15:SOUND
4,468,150,15
280 FOR g=1 TO 3500:NEXT:WINDOW #2,1,40,1,22:PRINT #2,STRING$(24,CHR$(11)):
WEND
290 REM ----- SET UP NEW ALIEN
300 alx=(INT(RND*8)*20)+80:aly=(INT(RND*5)*20)+80:distance=1:RETURN
310 REM ----- DRAW ALIEN
320 alienw=PEEK(Addr(distance)):alienh=PEEK(Addr(distance)+1):CALL 40110,addr
(distance),alx-(alienw/1.3),aly+alienh/2:RETURN
330 REM ----- MOVE ALIEN
340 GOSUB 320:distance=distance+1:IF distance=9 THEN landed=landed+1:LOCATE
(landed*10),25:PRINT a$::IF landed=3 THEN g=REMAIN(1):RETURN ELSE GOSUB 300
350 GOSUB 320:SOUND 1,900-(50*distance),5,7+distance,0,0,10:SOUND 1,900,5,7+
distance:SOUND 2,900,5,10,0,0,8:SOUND 2,900,5,10,0,0,10:SOUND 4,956,10,15,0,0,3:RETURN
360 REM ----- CREATE ENLARGED ALIEN
370 CLS:LOCATE #1,8,25:PRINT #1,"Making alien size: ";size;" at &":HEX$(Addr):
FOR x=0 TO 32:FOR y=1 TO 16:IF (TEST(x,y) AND size>3) OR (TEST(x,y) AND
x MOD 2=0 AND y MOD 2=0) THEN GOSUB 420
380 NEXT y,x:sprite=(200+(size*2))/2:spritey=(size*16+100)/2:spritew=(size*30+
(20-size*4))/8:spriteh=size*8:CALL 40164,addr,sprite,spritey,spritew,spriteh
390 FOR g=1 TO 3:CALL 40110,addr,sprite,spritey:FOR h=1 TO 100 STEP 4:SOUND
1,h*15,1,15:SOUND 4,h*25,1,15:SOUND 2,900-h*5,1,15:NEXT h,g
400 addr(sprite)=addr:addr=addr+(sprite*spriteh)+50:sprite=sprite+1:RETURN
410 REM ----- PLOT ALIEN PIXELS
420 PLOT 200+x*size, y*size+100,3:DRAWR size,0:DRAWR 0,size:DRAWR -size,0:DRAWR
0,-size:PLOT 200+x*size+(20-size*4), y*size+100,2:DRAWR size,0:DRAWR 0,size:
DRAWR -size,0:DRAWR 0,-size
430 SOUND 1,size*x+300,1,15:SOUND 2,size*x+400,5,15,0,0,7:SOUND 2,size*z+400,1,
15:RETURN
440 REM ----- PLAYER FIRES
450 DI:FOR g=150 TO 50 STEP -20:SOUND 1,g,2,15:SOUND 4,g*2,2,15:NEXT
460 IF x=alx AND y=aly-20 THEN CALL 40110,bang,x-45,y+45:FOR g=1 TO 10 STEP
2:SOUND 7,956+(10*g),10,15,0,0,g:NEXT:GOSUB 320:score=score+10:LOCATE 1,25:PRINT
score::GOSUB 300:GOSUB 320:CALL 40110,bang,x-45,y+45
470 EI:RETURN
480 CALL &BC02:PAPER 0:PEN 1:MODE 2:PRINT"You have been playing...":PRINT:
LIST

```

Type-in: BAS Invaders

Stretch those fingers and find a comfortable chair because you're about to have a blast! Not only has Sean McManus written a new type-in program, he's also explaining how you can use colour swapping in BASIC to enable rapid arcade action.

Written entirely in BASIC, *BAS Invaders* is certain to look familiar. That's because the game – pronounced BAS to rhyme with space because it's short for BASIC – is styled on Taito's *Space Invaders*. And, while it's not a full clone, it is surprisingly fast with up to 18 objects moving at the same time.

The trick is that the aliens are animated using colour swapping. All the alien positions are drawn at the start, using pens with the same colour as the background so you can't see them. Using the INK command, I changed the colour in the pens to make the aliens appear or disappear at the different positions.

There are four rows of four aliens. Each alien moves left and right, across five screen positions. For all the aliens, position 1 uses pen 1, position 2 uses pen 2 and so on. To make the aliens appear in a particular position, the pen at those positions is switched to a visible ink and the pen at the old positions is changed to the background colour. The movement is instant. When an alien is hit, its five positions are overwritten with spaces.


Although the pens used for the aliens are 1 to 5, the program counts them from -5 to 5, adding 1 each time. The code treats a negative number as a positive

number and skips 0. The result is that the aliens move from positions 5 to 1 and back to 5 again.

This approach means the program doesn't need to keep track of whether aliens are moving left or right: the pen number just keeps increasing until it reaches 5, when it loops back to -5.

There are some additional effects. I've used two alien shapes so the nasties have crude animation like a Nintendo Game & Watch device. When they move sideways, they change their pose. I've also drawn each alien in two colours. Both colours are turned on and off together, so we have the appearance and speed of multicolour sprites, implemented in pure BASIC.

I'm using the TEST command to check for aliens on the screen as the player's missile travels. As a result, I don't need to store the alive/dead status of each alien. The screen stores that. I only need to keep track of how many aliens are left on each row, so I can tell when the aliens win by reaching the bottom of the screen.

There are a couple of speed optimisations. I found using DEFINIT to set all variables to integers speeds up programs considerably. I also discovered that using INKEY with key numbers is much faster than using INKEY\$. Have fun! 

[TYPE IN]

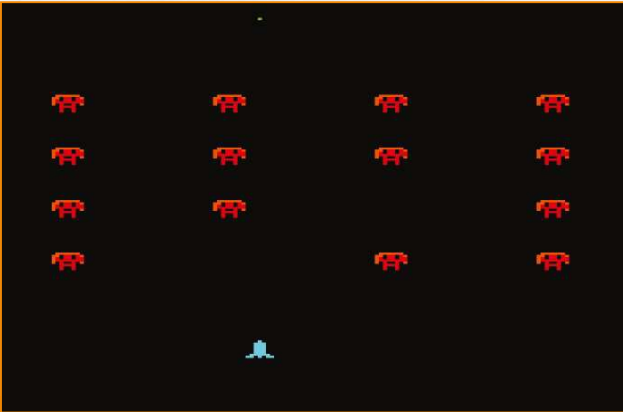
```

• 10 REM --- BAS INVADERS
• 20 REM --- (C)2022 Sean
  McManus - www.sean.co.uk
• 30 REM
• 40 ON BREAK GOSUB 380:SYMBOL
  250,126,219,219,255,165,60,6
  6,129:SYMBOL
  251,126,219,219,255,165,60,3
  6,36:SYMBOL
  252,126,193,128,128,129:GOSUB
  B 400:'Two On Breaks are
  needed because the program
  loops by RUNNING from the
  next line
• 50 ON BREAK GOSUB 380:MODE
  0:PAPER 0:INK 7,20:DEFINT a-
  z:LOCATE 10,25:PEN 7:PRINT
  CHR$(239);:PRINT
  CHR$(22)CHR$(1)CHR$(23)CHR$(
  1):'XOR and transparency on
• 60 DIM
  aliens(25):alienrow=1:winner
  =0:position=-
  5:posink=5:ship=10:counter=1
  :missiley=0
• 70 FOR row=1 TO 10 STEP
  3:aliens(row)=4:LOCATE
  1,row:FOR col=1 TO 4:FOR
  cell=1 TO 5:INK cell,0:INK
  cell+8,0:PEN cell:PRINT
  CHR$(250+(cell MOD 2));:PEN
  cell+8:PRINT
  CHR$(8);CHR$(252);:NEXT
  cell,col,row
• 80 PRINT CHR$(22)CHR$(0):'
  Transparency off
• 90 WHILE winner=0:LOCATE
  ship-1,25:PEN 7:PRINT "
  ";CHR$(239);" ";:' 239 is
  the rocket
• 100 IF INKEY(63)=0 AND
  ship<19 THEN ship=ship+1
• 110 IF INKEY(71)=0 AND
  ship>2 THEN ship=ship-1
• 120 IF INKEY(47)=0 AND
  missiley=0 THEN
  missilex=ship:missiley=25
• 130 counter=counter+1:IF
  counter MOD 5=0 THEN GOSUB 190
• 140 IF missiley>0 THEN GOSUB
  330 ELSE GOSUB 320
• 150 WEND
• 160 REM --- Game Over
• 170 INK 14,0:PAPER 14:PEN
  0:LOCATE 7,7:PRINT
  SPACE$(8):LOCATE 7,8:PRINT "
  ";CHR$(winner);" WINS
  ":LOCATE 7,9:PRINT
  SPACE$(8):PAPER 0:INK 14,20,21
• 180 gap=250:FOR note=1 TO
  7:SOUND 4,956-
  (g*50),5,15:SOUND 1,239+

```



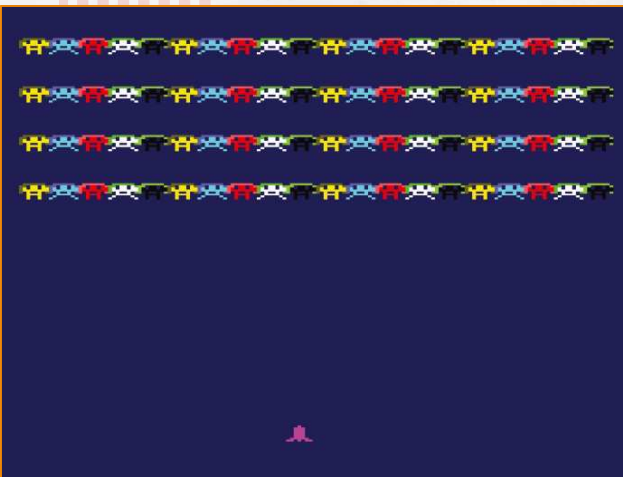
The title screen enlarges the alien sprite to create impact with just a few lines of code.



BAS Invaders is a simple but fast space shoot-em-up written 100% in BASIC.

```
(g*50),5,15:SOUND
2,478,10,15:FOR delay=1 TO
gap:NEXT:gap=gap*1.25:NEXT:S
OUND 1,239,120,15:SOUND
2,478,120,15:SOUND
4,956,120,15:FOR delay=1 TO
4500:NEXT:RUN 50
• 190 REM --- Move aliens
left-right
• 200 position=position+1:IF
position=6 THEN position=-
5:IF counter>250 THEN GOSUB 250
• 210 IF position<0 THEN INK
posink,0:INK
posink+8,0:posink=ABS(positi
on):INK posink,6:INK
posink+8,15 ELSE IF
counter>250 THEN GOSUB 250
• 220 IF position=0 THEN GOTO
190
• 230 RETURN
• 240 REM --- Aliens move down
• 250 IF missiley>0 THEN GOSUB
320
• 260 LOCATE 1,1:PRINT
CHR$(11):LOCATE ship-
1,25:PRINT " ";CHR$(239);" ";
• 270 IF missiley>0 THEN GOSUB
320
• 280 alienrow=alienrow+1:IF
aliens(26-alienrow)>0 THEN
winner=250:' 250 is alien
• 290 SOUND 4,956,5,13:SOUND
1,239,5,13:SOUND 2,239,5,13
• 300 RETURN
• 310 REM --- Show/hide missile
• 320 PLOT (missilex*32)-18,400-
(missiley*16)+4,14:RETURN
• 330 REM --- Missile movement
• 340 GOSUB
```

```
320:missiley=missiley-1:IF
TEST ((missilex*32)-18,400-
(missiley*16)+4) <>ABS(positi
on) THEN GOSUB 320:RETURN
• 350 INK
ABS(position),24:LOCATE
(INT((missilex-1)/
5)*5)+1,missiley:PRINT
SPACES(5):score=score+1:SOUN
D 2,956,20,15,0,0,7:IF
score=16 THEN winner=239
• 360 aliens(missiley-
alienrow+1)=aliens(missiley-
alienrow+1)-1:missiley=0:INK
ABS(position),6:RETURN
• 370 REM --- On Break
• 380 CALL &BC02:CALL &BB06:PEN
1:PAPER 0:MODE 2:LIST
• 390 REM --- Title Screen
• 400 MODE 1:CALL &BC02:INK
0,0:BORDER 0:INK 2,0:INK
3,0:INK 1,24,0:PAPER 0:PEN
1:LOCATE 1,25:PRINT CHR$(250)
• 410 FOR x=0 TO 16:FOR y=0 TO
16:IF TEST(x,y) THEN PEN
2:LOCATE x+13,16-y:PRINT
CHR$(143):PEN 3:LOCATE
x+14,17-y:PRINT CHR$(207)
• 420 NEXT y,x:INK 1,0:LOCATE
1,25:PRINT" ":PEN 1:LOCATE
14,20:PRINT CHR$(250);" BAS
INVADERS ";CHR$(250):LOCATE
4,22:PRINT"A 100% BASIC demo
of Space Invaders":LOCATE
5,23:PEN 2:PRINT"By Sean
McManus - www.sean.co.uk"
• 430 PEN 1:LOCATE
4,25:PRINT"Keys: Z, X, Space.
Any key to start":INK
2,15:INK 3,6:INK 1,24:CALL
&BB06:RETURN
```



Here is the screen display with the default inks, showing all the aliens in position.

I developed and tested the game using the *WinAPE* emulator. If you don't want to type the program in, you can copy the code at www.sean.co.uk/books/amstrad/play_bas_invaders.shtm and paste it into *WinAPE*. The web page also includes an in-browser emulation, although the keys are less responsive in the online emulator.

About Sean McManus

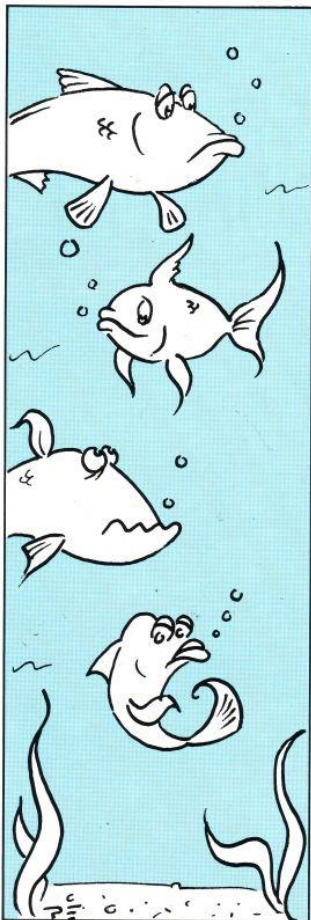
Sean wrote type-ins for *Amstrad Action* and *Amstrad Computer User*, and he contributes to the relaunched *Amix*. He also writes books to inspire children and adults with technology. They include *Scratch Programming in Easy Steps*, *Raspberry Pi For Dummies* (with Mike Cook), and *Mission Python*, which shows you how to build a space adventure game. His website includes free chapters from his books, plus his Amstrad type-ins, tools, and tutorials. Visit: www.sean.co.uk/books/

FISHTANK

BY SEAN MCMANUS

This one may not be the longest of listings but boy are you going to love it. Just like the latest video craze from the States, this excellent routine creates a simulated fishtank environment. Run it and relax, with your own personal fish swimming around in the background. Very clever and a totally brilliant proggy to amuse your friends.

When you are typing it in, don't forget to omit the proofcodes at the end of the lines, as these are only for error checking purposes.



```

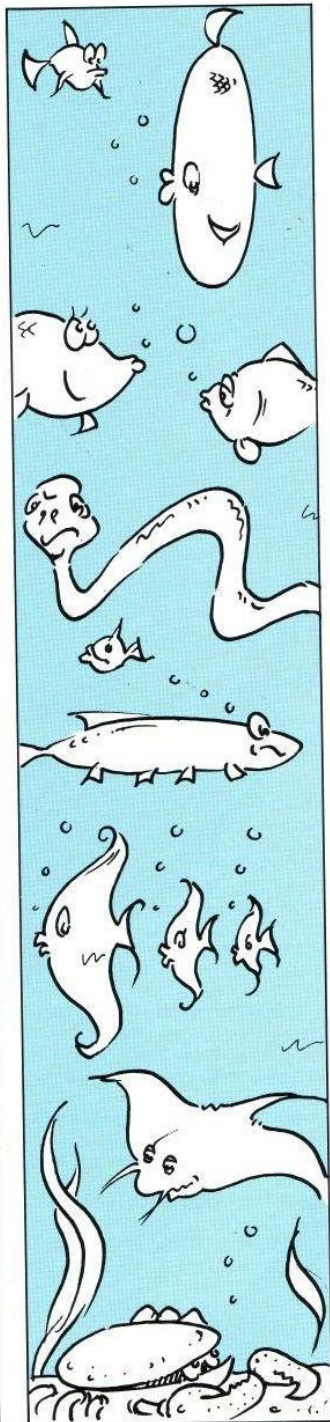
10 Fish Tank Simulator [F1]
20 (C) 1991 Sean McManus - 28th July 1991 [F2]
30 [F3]
40 It's Madness - with greetings to Sue. [F4]
50 [F5]
60 MODE 1:CALL &BC02:INK 0.0:BORDER 0:LOCATE 12.5: [04]
PEN 1:PAPER 0:PRINT"Fish Tank Simulator":LOCATE 11 [04]
.7:PRINT(C) 1991 Sean McManus:LOCATE 12,17:PEN 2 [04]
:PRINT"Hello. How are you?":SYMBOL 255,48,254,22 [04]
.60,116,210,254,16::LOCATE 40,25:PRINT CHR$(255): [04]
70 LOCATE 15,15:PRINT"Polite Notice:":ON BREAK GOS [95]
UB 380:MEMORY 17663:mem=17664:FOR g=1 TO 14:chk=0: [95]
FOR h=1 TO 77:READ a$:a=VAL("&"+a$):POKE mem,a:chk [95]
=chk+a:mem=mem+1:NEXT:READ chk$:IF VAL("&"+chk$)< [95]
chk THEN PRINT"error in line"30+(g*10):STOP [95]
80 NEXT:CHK=0:FOR g=1 TO 77:READ a$:a=VAL("&"+a$): [48]
POKE 39999+g,a:chk=chk+a:NEXT:IF chk>&1D07 THEN P [48]
RINT"Error in 180":STOP [48]
90 DATA 01,01,00,1D,08,00,00,00,54,00,00,00,00,00, [92]
00,00,54,00,00,00,00,00,00,00,FC,A8,00,00,00,00, [92]
,00,AC,A8,00,00,00,00,00,04,0C,0C,00,00,00,00,0 [92]
C,0C,0C,08,00,00,00,04,0C,A6,A6,0C,00,00,00,04,4C, [92]
8C,CC,0C,00,00,00,04,CC,26,66,8C,00,00,00,913 [92]
100 DATA 00,04,0C,0C,0C,0C,00,00,00,04,1D,0C,1D,0C [3A]
,00,00,00,0C,3F,2E,3F,2E,08,00,00,0C,3F,2E,3F,2E,0 [3A]
8,00,00,0C,7F,2E,7F,2E,08,00,00,0C,7F,2E,7F,2E,08, [3A]
00,04,0C,3F,2E,3F,2E,0C,00,04,0C,1D,5C,1D,0C,0C,00 [3A]
,04,0C,0C,EC,8C,0C,0C,00,04,0C,0C,EC,8C,A36 [3A]
110 DATA 0c,0c,00,04,0c,84,cc,8c,84,0c,00,04,0c,84 [ED]
,4c,0c,84,0c,00,04,48,48,0c,48,48,0c,00,ac,0c,48,c [ED]
0,c,0c,0c,ab,fc,0c,48,cc,c8,0c,5c,a8,dc,0c,0c,4c, [ED]
84,0c,5c,88,a8,0c,0c,48,0c,0c,08,a8,a8,04,0c,0c,0c [ED]
,0c,00,a8,00,00,0c,0c,0c,08,00,00,00,00,132c [ED]
120 DATA 04,0c,0c,00,00,00,1D,0c,00,00,00,00,00,40 [2A]
,a8,00,00,00,00,00,00,00,00,00,d4,a8,00,00,00,0 [2A]
0,00,00,00,00,40,ec,a8,00,00,00,00,00,00,00,00, [2A]
00,54,ec,fc,04,08,00,00,00,00,00,00,00,54,ec,fc,0c [2A]
,0c,00,00,00,00,00,00,00,c4,fc,dc,0c,0c,bcd [2A]
130 DATA 08,00,00,00,00,00,00,00,0c,fc,8c,0c,0c,0c,ab [67]
,00,00,00,00,00,fc,dc,ac,8c,0c,5c,54,00,00,00,0 [67]
0,fc,dc,0c,8c,0c,0c,08,00,00,00,40,dc,ac,0c,cc, [67]
0c,0c,80,00,00,00,00,40,ec,0c,0c,cc,0c,48,c0,00,00 [67]
,00,00,54,ec,4c,0c,cc,0c,48,68,00,00,00,1328 [67]
140 DATA 00,54,ac,4c,cc,cc,0c,48,68,0c,00,00,00,54 [1E]
,ac,4c,8c,cc,0c,0c,84,0c,00,00,00,54,0c,4c,8c,4c,0 [1E]
C,0c,0c,0c,00,00,00,54,0c,4c,8c,4c,8c,0c,0c,08,00 [1E]
00,00,04,8c,4c,8c,0c,8c,0c,0c,00,00,00,00,04,cc,0c [1E]
,cc,0c,cc,0c,cf,8a,00,00,00,04,cc,0c,cc,11c1 [1E]
150 DATA 0c,4c,8c,0c,8a,00,00,00,0c,cc,0c,4c,8c,0c [BE]
,0c,0c,08,40,80,00,0c,0c,0c,4c,cc,0c,0c,0c,08,54,f [BE]
C,04,0c,4c,8c,0c,0c,0c,0c,0c,d4,cc,ac,0c,4c,8c, [BE]
0c,0c,0c,0c,0c,00,ec,fc,ac,0c,0c,cc,0c,0c,d4,0c,08 [BE]
,00,fc,ec,ac,0c,0c,0c,0c,d4,ac,00,00,14f2 [BE]
160 DATA fc,dc,04,0c,0c,0c,0c,0c,ec,fc,00,00,0c,ec,fc [2E]
,00,0c,0c,0c,0c,0c,5c,dc,a8,00,54,a8,00,00,04,0c,0 [2E]
C,0c,00,ec,dc,00,00,00,00,00,00,04,0c,00,00,54,fc, [2E]
00,1d,0c,00,00,00,00,00,54,80,00,00,00,00,00,00,00 [2E]
,00,00,00,54,e8,00,00,00,00,00,00,00,e89 [2E]
170 DATA 00,00,54,dc,80,00,00,00,00,00,00,00,04,08 [15]
,fc,dc,a8,00,00,00,00,00,00,0c,0c,fc,dc,a8,00,0 [15]
0,00,00,00,00,04,0c,0c,ec,fc,c8,00,00,00,00,00,54, [15]

```

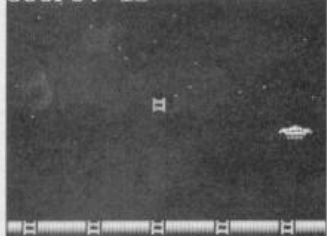
```

0C,0C,0C,4C,FC,DC,00,00,00,00,00,A8,AC,0C,4C,5C,EC [15]
.FC,00,00,00,00,00,04,0C,0C,4C,0C,EC,FC,1288 [15]
180 DATA 00,00,00,00,00,00,40,0C,0C,CC,0C,5C,EC,80,00 [0B]
.00,00,00,00,00,0C,CC,0C,0C,DC,80,00,00,00,04,94,8 [0B]
.4,0C,CC,0C,8C,DC,A8,00,00,00,0C,94,84,0C,CC,0C,8C, [0B]
5C,A8,00,00,00,0C,48,0C,0C,CC,4C,8C,5C,A8,00,00,00 [0B]
.0C,0C,0C,0C,8C,4C,8C,0C,A8,00,00,00,04,1390 [0B]
190 DATA 0C,0C,4C,8C,4C,8C,0C,A8,00,00,00,00,0C,0C [5E]
.4C,0C,4C,8C,4C,08,00,00,00,45,CF,0C,CC,0C,CC,0C,C [5E]
C,08,00,00,00,45,0C,4C,8C,0C,CC,0C,CC,08,00,00,00, [5E]
04,0C,0C,0C,4C,8C,0C,CC,0C,CC,00,00,00,04,0C,0C,0C,CC [5E]
.8C,0C,CC,0C,00,40,80,00,0C,0C,0C,0C,FE1 [5E]
200 DATA 4C,8C,0C,08,FC,A8,00,0C,0C,0C,0C,4C,8C [E3]
.0C,5C,CC,EC,00,04,0C,EB,0C,0C,CC,0C,0C,5C,FC,DC,0 [E3]
0,00,5C,EB,0C,0C,0C,0C,5C,DC,FC,00,00,FC,DC,0C, [E3]
0C,0C,0C,0C,08,EC,FC,00,54,EC,AC,0C,0C,0C,0C,0C,0C [E3]
.FC,DC,00,EC,DC,00,0C,0C,0C,08,00,00,00,54,17E8 [E3]
210 DATA A8,00,FC,A8,00,00,0C,08,00,00,00,00,00,08 [56]
.04,98,CC,64,30,4C,64,4C,CC,8C,8C,30,24,98,30,98,C [56]
C,4C,CC,4C,8C,98,4C,30,0C,CC,4C,CC,64,8C,98,24,30, [56]
07,04,40,EC,30,00,D0,30,30,20,B0,30,20,30,30,30,30 [56]
.0A,30,30,30,82,30,30,0F,82,10,4B,C3,00,182A [56]
220 DATA 08,04,00,00,E8,00,00,40,00,A0,00,40,00,A0 [EF]
.00,00,D0,00,54,80,00,00,80,50,00,00,80,50,00,00,5 [EF]
0,A0,00,00,07,04,54,00,80,A0,00,80,A0,00,00,00,00, [EF]
E0,C0,A0,00,00,00,00,A0,00,00,80,80,A8,54,00,80,40 [EF]
.00,00,00,00,00,00,00,00,00,00,00,00,00,00,F23 [EF]
230 DATA DD,6E,00,DD,66,02,DD,7E,04,F5,CD,1A,BC,F1 [E0]
.E5,6F,26,00,29,01,7D,9C,09,4E,23,46,50,59,69,60,4 [E0]
E,23,13,13,46,E1,C5,E5,1A,AE,77,23,13,10,F9,E1,01, [E0]
00,08,09,30,04,01,50,CO,09,C1,0D,20,E8,C9,00,45,03 [E0]
.45,ED,45,4B,47,A9,48,CB,48,E9,48,0B,49 [E0]
240 OUT &BC00,65:OUT &BD00,0:MODE 0:INK 0,1: BORDER [27]
0:INK &B,0.6:INK 15,0.15:INK 14,26,15:INK 13,15,6 [27]
:INK 12,6,15:INK 6,0:INK 1,26:INK 2,6:INK 3,15:INK [27]
4,10:INK 5,20:INK 7,24:INK 8,1:INK 9,19:INK 10,21 [27]
:SPEED INK 205.5:FOR g=399 TO 390 STEP -1 [27]
250 PLOT 0,g:DRAWR 640,0.6:NEXT:PLOT 12,388:DRAW 6 [EE]
26,388:FOR g=0 TO 19:CALL 40000,4,g,24:CALL 40000, [EE]
4,g,23:NEXT:FOR g=1 TO 16:READ x,y:CALL 40000,5,x, [EE]
y:NEXT:DATA 5,22,6,22,6,20,7,20,5,21,6,21,7,19,8,1 [EE]
9,10,20,9,19,10,19,11,20,12,21,11,21,11,22,12,22 [EE]
260 FOR l=1 TO 5:READ x,h:PLOT x+2,32,9:DRAWR 0,h: [FA]
FOR g=1 TO h/40:PLOT x,g*40:FOR h2=5 TO 1 STEP -1: [FA]
DRAWR -RND*(h2*2),RND*(h2*5):NEXT h2,g:FOR g=1 TO [FA]
h/40:PLOT x,g*40,10:FOR h2=5 TO 1 STEP -1:DRAWR RN [FA]
D*(h2*2),RND*(h2*5):NEXT h2,g [FA]
270 NEXT:PLOT 1,399,1:DRAW 1,1:DRAW 639,1:DRAW 639 [DF]
.399:DATA 20,200,600,150,620,200,50,110,130,70 [DF]
280 x=5:y=5:dir=2:CALL 40000,2,x,y:OUT &BC00,65:OU [4B]
T &BD00,40:AFTER RND*150 GOSUB 360:WHILE 1:odir=di [4B]
r:ox=x:oy=y:DI [4B]
290 IF RND>0.8 THEN dir=dir XOR 1 [94]
300 IF dir=3 AND x>1 THEN x=x-1:IF x=1 THEN dir=2 [E3]
310 IF dir=2 AND x<13 THEN x=x+1:IF x=15 THEN dir= [22]
3 [22]
320 IF RND<0.5 AND y>3 THEN y=y-1 [E7]
330 IF RND>0.4 AND y<15 THEN y=y+1 [B8]
340 CALL 40000,odir,ox,oy:IF dir<>odir THEN CALL 4 [D6]
0000,1,x,y:FOR h=1 TO 100:NEXT:CALL 40000,1,x,y [D6]
350 CALL 40000,dir,x,y:EI:FOR h=1 TO 400:NEXT:WEND [7D]
360 CALL 40000,dir,x,y:CALL 40000,1,x,y:FOR g=1 TO [A0]
1000:NEXT:CALL 40000,6,x-1,y+1:FOR g=y TO 1 STEP [A0]
-1:CALL 40000,6,x-1,g+1:CALL 40000,6,x-1,g:FOR h=1 [A0]
TO 50:NEXT h,g:CALL 40000,6,x-1,1:CALL 40000,7,x- [A0]
1,0:FOR h=1 TO 50:NEXT:CALL 40000,7,x-1,0 [A0]
370 FOR g=1 TO 1000:NEXT:CALL 40000,dir,x,y:CALL 4 [20]
0000,1,x,y:AFTER RND*1200 GOSUB 360:RETURN [20]
380 MODE 2:INK 1.26:INK 0.0:PEN 1:PAPER 0:LIST [18]

```



Score: 10



Ah, now this really is the Manhattan skyline, or somewhere like that.

Alien Intervention

This month's final contribution is a game called Alien Intervention by Sean McManus of Stevenage in Herts (nice signature by the way). All instructions are included within the game.

```
HAGN 10: 'Alien Intervention
NAME 20: '(c)1991 Sean McManus--1st-March
DCLJ 30-MEMORY-844FF:RESTORE-180:FOR-gD-TO-6
6:READ-af:a:VAL('A'+a):POKE-40000+g,a:c
bk:chk+a:NEXT-IF-6436(C)chk-THEN-PRINT"Er
ror-in-280":STOP
FDFE 40-men:17604:FOR-g=1-TO-3:chk=0:FOR-h=1
-TO-77:READ-af:a:VAL('A'+a):POKE-men,a:in
en:men+1:chk:chk+a:NEXT:READ-chk:IF-chk
(>VAL('A'+chk))-THEN-PRINT"Checks-un+a:f
ter-line-220-is-wrong":STOP
DEI 50-NEXT:MODE-1:INX-2,0:INX-0,0:BOARD-0:
PEN-2:PAPER-0:FOR-g=1-TO-16:READ-as:LOC
ATE-20-0,5:ENL(en+a),g:5:PRINT-as:NEXT:INX
-3,3:INX-1,25:PRINT-CHR(23)CHR(1):FOR
-g=334-TO-50-STEP-2:PLOT-1,g,3:DRAM-639
,g:NEXT:CALL-8B86
PDCB 60-MODE-0:INX-0,0:BOARD-0:INX-14,2:INX
-6,2:INX-0,1:x:40:y:250:af:"Alien-Inte
rvntion" :GOSUB-250:x:300:y:330:af:"By" :GOS
UB-250:x:150:y:310:af:"Sean-McManus" :GOS
UB-250:x:200:y:220:af:CHR(184)+"MCMXCI"
:GOSUB-250:x:120:y:70:af:"Press-any-key"
GDMX 70-GOSUB-250:SYMBOL-255,48,254,22,60,116
,210,254,16:ia:CHR(255):x:300:y:160:GOS
UB-250:INX-15,1:PLOT-1,1,8:DRAM-1,394:DR
AM-634,394:DRAM-634,1:DRAM-1,1:PLOT-5,5,
6:DRAM-5,399:DRAM-639,399:DRAM-639,5:DR
AM-5,5:WHILE-!INKEY(")")WEND:CALL-8B86
IDNH 80-MODE-0:INX-1,26:INX-2,6:INX-3,15:INX
-5,20:INX-6,24:INX-7,24:INX-8,1:INX-9,19:I
NK-15,15,8:INX-14,24,15:INX-13,0,26:INX
-12,26,4:DEFINT-a-z:PAPER-0:FOR-gD-TO-1
9:CALL-40000,3,g,24:CALL-40000,3,g,17:FL
OT-RND*640-RND*250,8:PLOTR-4,0,5,FL
OTR-4,0,0
```

```
LDBF 90-PLOTR-4,4:PLOTR-0,8:FOR-h=17-TO-17*(
RND*4)-STEP-1:LOCATE-g+1,h:PRINT"-":JNE
XT-h,g:PAPER-0:PEN-1:LOCATE-1,11:PRINT-5
PAGE*(20):FOR-g=1-TO-19:STEP-4:CALL-400
00,3,g,17:CALL-40000,2,g,17:NEXT-PRINT-C
HR(23)CHR(0):PLOT-1,1,0:DRAM-1,359:DR
AM-639,359
FRKH 100-DRAM-639,1:WINDOW-#1,1,20,1,1:PAPER
-#1,0:PEN-#1,1:PRINT"#1,".....GOOD-LOCK-1
....."CHR(23)CHR(1):
MCLF 110-x=9:CALL-40000,1,x,10:GOSUB-240:EVEN
Y-30,1-GOSUB-190:SPEED-KEY-4,2:WHILE-p(f)
(<f:INT(RND*5):r=(f*4)+1:CALL-40000,2,
c,0:FOR-h=1-TO-23-p(f):CALL-40000,2,c,h:
CALL-40000,2,c,h-1:FOR-g=1-TO-5
ACCE 120-as:UPPER(I)NEXT:IF-as(")-THEN-CALL
I-40000,1,x,10:x:=(a+"B")-(a+"M"):X=X
-CX-20):X=X+(CX-13):CALL-40000,1,x,10:IF-a
=")-THEN-GOSUB-210
ICKN 130-NEXT:IF-h=9-AND-(cx-0)<cx+1)-THEN-
CALL-40000,2,c,h:CALL-40000,5,c,h:score=
score+10:PRINT"#1,"Score":score:GOSUB-260
:CALL-40000,5,c,h:p(f)-p(f)-1:h=70
FBAC 140-NEXT:(f)-p(f)+1:5OUND-7,950-p(f)*10
0,5,15:WEND:IF-q-1-THEN-g=REMAIN(2):GOS
UB-240
CCPC 150-CALL-40000,4,MX,23*(MX/2-MX/2):g=REN
AIN(1):h=c-mx:nd=(h(0)+ABS(h(0)):FOR-g=1-
TO-ABS(h):mx=c+nd:CALL-40000,4,MX,23*(M
X/2-MX/2):FOR-t=1-TO-250:NEXT:CALL-40000
,4,MX,23*(MX/2-MX/2):NEXT
IDH0 160:FOR-g=23-TO-16-STEP-1:1:CALL-40000,4,
MX,g:CALL-40000,2,mx,g:FOR-t=1-TO-250:NE
XT:CALL-40000,4,MX,g:CALL-40000,2,mx,g:N
EXT:PAPER-0:LOCATE-mx+1,17:PRINT"-":JNE
L-40000,4,mx,16:FOR-gD-TO-15:PLOT-x*32+
12,20,g:DRAM-x*32+4,142:PLOTR-0,0
PCLD 170-DRAM-x*32+46,220:FOR-h=1-TO-50:NEXT-
h,g:CALL-40000,1,x,10:CALL-40000,5,x,x,10:
CALL-40000,5,x+1,10:FOR-h=1-TO-150:NEXT:
CALL-40000,5,x,10:CALL-40000,5,x+1,10:LO
CATE-3,7:PRINT"G-a-m-e-o-v-e-r" :GOSUB-2
60
OCGN 180-CALL-8B86:RUN:0:DATA-D1,EE,00,DD,6
6,02,DD,7E,04,FS,CD,1A,BC,11,ES,07,EE,06
,28,01,77,9C,09,4E,23,46,50,53,65,60,4E,
23,13,13,46,E1,CS,ES,1A,6E,77,23,13,10,F
9,E1,01,00,00,09,CL,DD,20,EE,C9,00,45,03
,45,45,45,67,45,09,45,09,45
HDEI 190-IF-q(-1)-THEN-CALL-40000,4,MX,23*(MX
/2-MX/2):MX-MX+D:CALL-40000,4,MX,23*(MX
/2-MX/2):IF-MX=0:OR-MX=19-OR-RND>0.7-TH
E-N:MD=MD:DATA-01,01,00,00,00,00,00,01,56
,01,02,00,00,01,03,56,DD,00,81,03,02,56,
34,00,20,FO,FO,FO,AL,CO,DD,78,3C,04,34,C
0,FO,AL,ED,CO
EDRE 200-RETURN:DATA-C0,CO,CO,C3,52,02,43,C3,
02,02,C3,83,01,00,01,03,03,03,03,02,00,01
0,00,0F,0F,0F,0F,00,00,04,04,02,00,01
50,FC,1DD2,F4,89,54,F9,E0,81,40,03,03,0
8,40,02,00,01,50,E0,F4,89,40,F0,34,81,54
,03,03,85,06,04,F0,70,F0,F0,D4,D4,D4,D4,
D4,D4,D4,D4,FC
KCDI 210-PLOT-x*32+12,220,15:DRAWR-0,-224:PLO
T-x*32+48,220:DRAWR-0,-224:GOSUB-270:DAT
A-FC,FC,FC,EC,EC,EC,CC,CC,CC,CC,8C,8C
,8C,8C,08,08,08,08,08,04,40,94,68,00,CO,
CO,CO,CO,91,E2,91,E2,68,94,68,2007
IDKE 220-IF-x=mx-OR-yt=zm-THEN-gc-1:CALL-400
00,4,MX,23*(MX/2-MX/2):CALL-40000,5,mx,2
3*(MX/2-MX/2):score=score+100:PRINT"#1,"S
core":score:FOR-g=1-TO-30:NEXT:CALL-4000
0,5,mx,23*(MX/2-MX/2):nxc-1:AFTER-150,2:
GOSUB-240:GOSUB-260
FDBG 230-PLOT-x*32+12,220,15:DRAWR-0,-224:PLO
T-x*32+48,220:DRAWR-0,-224:RETURN:DATA-9
```

```
4,16,56,89,29,29,0C,1C,16,16,16,03,29,29
,69,34,16,00,04,00,46,00,01,01,06,00,00
,06,0C,0C,03,4C,F8,EC,8C,C8,50,20,C4,05,F
8,AC,3C,04,DC,8C,03,01,8C,89,00,00,00,00
,00,00,00,00
MCHG 240-mx:INT(RND*15):nd=1:CALL-40000,4,MX,
23*(MX/2-MX/2):r=0:RETURN:DATA-00,00,00,
00,00,00,00,00,00,00,00,00,00,00,00,00,0
0,00,101C
HEAI 250-PRINT-CHR(23)CHR(1):TAG:GRAPHICS-
PEN-6:MOVE-x,y:PRINT-af:GRAPHICS-PEN-8:
MOVE-x-5,y-5:PRINT-af:TAGOFF:RETURN:DAT
A-Alien-Intervention--Instructions,,Lon
g-ago-banished-to-be-You-the-planet,"surf
ace,the-Rivanves-are-trying-to"
EENI 260:FOR-g=15-TO-6-STEP-2:5OUND-1,956,p
3,g,0,0,g:5OUND-2,456,p3,g,0,0,g:5OUND-0
4,239,p3,g,0,0,g:NEXT-RETURN:DATA-00,00
-using-ladders-dropped-in-by-the-Nemas
-Your-job-is-to-intercept,"these-delive
ries,smashing-the-ladders."
EECC 270-5OUND-1,239,5,15:5OUND-1,479,5,15:5O
UND-1,239,5,15:FOR-g=1-TO-15-STEP-2:5OUN
D-2,239-g*3,2,15:5OUND-4,956,2,g:NEXT:RE
TURN:DATA-"The-first-job-of-any-escaping
-Rivanve",will-be-your-extermination,You
-u-ay-zap-the-little-darlings-for
PBJO 280-DATA-"extra-points-and-great-satisfa
ction,","Keys,"B=Left;N=Right):SPC-Fi
re","Press-any-Key"
```

WALLBUSTER

BY SEAN MCMANUS

If you are one of those people who got hooked on Arkanoid and all the other derivations of bat-and-ball-knocking-blocks-out-of-the-wall games, then this month's epic type-in is going to be right up your street.

Taking a break from the serious stuff, it's time to get gaming again, and this

one by Sean McManus is guaranteed to hook you from the first bounce.

Using the cursor left and right keys to guide your bat, there are four different levels to beat, with a fantastic stereo soundtrack feature to keep you entertained whilst you battle it out against the blocks.

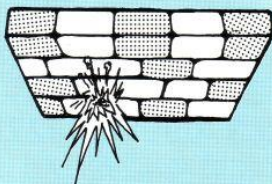
Be warned, this is no piece of cake and will have you pulling your hair out even before you've beaten the first wall.

Apart from that, it's great fun. Type it, save it and, ensuring that you've rebooted your machine so that the memory is fresh, get wallbusting straight away.

```

10 WALLBUSTER - (C) 1990 Sean McManus
20 (30th July - hi 2 every I saw on the 28th &
30th !)
30
40 This version (veery carefully) remixed 2nd M
ay 1991.
50
60 PRINT"Poking m/c":FOR g=40000 TO 40045:READ a$:
a=VAL("&"+a$):POKE g,a:chk=chk+a:NEXT:IF chk<>$126
2 THEN PRINT"Error in the very first data line":EN
D
70 DATA DD,6E,00,DD,66,02,CD,1A,BC,DD,5E,04,DD,56,
05,EB,4E,23,46,23,EB,C5,E5,1A,AE,77,23,13,10,F9,E1
,01,00,08,09,30,04,01,50,C0,09,C1,0D,20,E8,C9
80 PRINT"Poking Sprites - Please Wait":SYMBOL AFTE
R 255:MEMORY &7FFF:ADDR=&8000:FOR G=1 TO 20:CHK=0:
FOR H=1 TO 14:READ A$:D=VAL("&"+A$):POKE ADDR,D:CH
K=CHK+D:ADDR=ADDR+1:NEXT:READ CHK2$:IF VAL("&"+CHK
2$)<>CHK THEN PRINT"ERROR IN LINE"G*10+90:END
90 NEXT:DATA 08,04,00,40,A8,00,00,D0,FC,00,40,FC,C
C,88,550
100 DATA 54,EC,CC,08,54,CC,CC,08,54,CC,8C,08,00,CC
,688
110 DATA 0C,00,00,04,08,00,08,08,44,E4,CC,CC,CC,CC
,480
120 DATA 8C,08,D8,E8,FC,C4,C0,EC,0C,0C,CC,D4,CC,CC
,910
130 DATA CC,8C,0C,26,CC,EC,4C,4C,4C,0C,0C,26,CC,CC
,5FC
140 DATA 8C,8C,8C,0C,0C,26,44,CC,0C,0C,0C,0C,19,08
,343
150 DATA 44,8C,26,26,33,33,26,08,00,0C,0C,0C,0C,0C
,1EC
160 DATA 0C,00,08,04,FC,EC,EC,8C,A9,03,03,8E,A9,03
,561
170 DATA 47,8E,89,03,8B,8E,89,47,47,8E,89,8B,8B,8E
,64C
180 DATA CD,CF,CF,8E,8C,0C,0C,0C,08,04,E1,C3,C3,87
,6A3
190 DATA D6,FC,FC,85,D6,FC,FC,85,D6,FC,FC,85,D6,FC
,BCB
200 DATA FC,85,D6,FC,FC,85,C2,C0,C0,85,87,0F,0F,0F
,84F
210 DATA 08,04,54,F8,D0,80,F8,E0,E0,E0,F8,91,33,32
,82E
220 DATA E0,33,33,30,C0,33,32,32,91,33,31,30,32,32
,456
230 DATA 32,30,10,30,30,20,08,04,C0,C0,C0,C0,84,0C
,48E

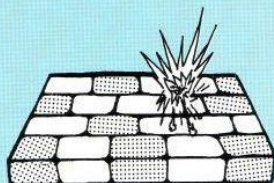
```



```

240 DATA 0C,48,84,0C,0C,48,84,0C,0C,48,84,0C,0C,48,84,0C,0C,48
,300
250 DATA 84,0C,0C,48,84,0C,0C,48,C0,C0,C0,C0,08,04
,4D4
260 DATA E8,F0,3C,CF,D0,B4,6D,8B,E8,F0,3C,CF,D0,B4
,9C6
270 DATA 6D,8B,E8,F0,3C,CF,D0,B4,6D,8B,E8,F0,3C,CF
,93A
280 DATA D0,B4,6D,8B,00,00,00,00,00,00,00,00,00,00,00
,27C
290 DEFINIT a-z:GOSUB 620:score=0:WHILE 1:DIM M(19,
24):OUT &BC00,65:FOR g=40 TO 0 STEP -1:OUT &BD00,g
:FOR h=1 TO 10:NEXT h,g
300 MODE 0:FOR G=0 TO 24:CALL 40000,&80EC,0,G:CALL
40000,&80EC,19,G:M(19,G)=1:M(0,G)=1:NEXT
310 ' DRAW MAIN SCREEN
320 ENV 1,1,15,1,15,-1,3:ENT 1,10,-1,2
330 FOR g=1 TO 18:m(g,0)=1:NEXT:LOCATE 2,1:PEN 5:P
RINT STRING$(18,"");PEN 1
340 ON ERROR GOTO 590:GOSUB 780
350 left=0:SCN=SCN+1:ON SCN GOSUB 730,750,770,790
' 4 DIFFERENT SHEETS
360 X=INT(RND*10)+4:CALL 40000,&8022,x,24:BX=x+1:B
Y=23:BDX=-1:BDY=-1:CALL 40000,&8000,bx,by
370 OUT &BC00,65:FOR g=0 TO 40:OUT &BD00,g:FOR h=1
TO 10:NEXT H,G
380 WHILE INKEY$<>"" :WEND:CALL &BB06
390 ' MAIN LOOP
400 EVERY 4,1 GOSUB 480:WHILE left>0
410 OX=X:WHILE X=OX AND LEFT>0:IF INKEY(8) THEN X=
X+1
420 IF INKEY(1) THEN X=X-1
430 WEND
440 DI:IF X<1 OR X>17 THEN X=OX
450 CALL 40000,&8022,OX,24:CALL 40000,&8022,X,24:M
(OX,24)=0:M(OX+1,24)=0:M(X,24)=1:M(X+1,24)=1
460 EI:WEND:ERASE m,t:WEND
470 ' BALL MOVE
480 OBX=BX:OBY=BY
490 BX=BX+BDX:IF m(bx,by) THEN 510
500 by=by+bdy
510 IF BX=18 THEN BDX=-1:SOUND t(bx),956,5,15,1,1:
IF BY>1 AND BY<24 THEN 570
520 IF BX=1 THEN BDX=1:SOUND t(bx),956,5,15,1,1:IF
BY>1 AND BY<24 THEN 570
530 IF by=1 THEN bdy=1:SOUND t(bx),956,5,15,1,1:GO
TO 570
540 IF m(bx,by)=1 THEN f=1:IF BY=24 THEN BDY=1
550 IF M(BX,BY)>1 THEN CALL 40000,a(M(BX,BY)),BX,B
Y:M(BX,BY)=0:F=1:score=score+10:left=left-1:IF LEF
T=0 THEN G=REMAIN(1)
560 IF F THEN BX=OBX:BY=OBY:BDY=-BDY:SOUND t(bx)+1
28,851,40,15,1,1:F=0:IF RND>0.5 THEN BDX=-BDX:GOTO
490 ELSE 490
570 CALL 40000,&8000,OBX,OBY:CALL 40000,&8000,BX,B
Y:RETURN
580 ' DEATH - TRIGGERED BY BALL'S SUBSCRIPT OUT/RA
NGE ERROR
590 CALL 40000,&8000,oBX,oBY:a$="Game Over":y=200:
GOSUB 600:a$="Score:"+STR$(score):y=160:GOSUB 600:
INK 15,26,15:INK 14,15,26:WHILE INKEY$<>"" :WEND:CA
LL &BB06:RUN 290
600 TAG:FOR g=1 TO LEN(a$):PLOT -10,-10,14+(g AND
1):MOVE (320-LEN(a$)*14)+g*28,y+RND*10:PRINT MID$(
a$,g,1):NEXT:TAGOFF:RETURN
610 ' TITLE SCREEN & INKS INIT
620 MODE 0:CALL &BC02:BORDER 3:RESTORE 620:FOR G=0
TO 12:READ I:INK G,I:NEXT

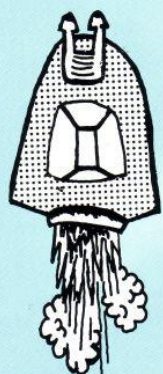
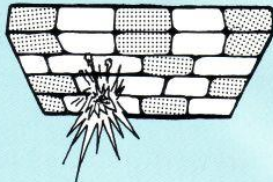
```



```

630 A$="WALLBUSTER":Y=350:GOSUB 600:INK 15,25,15:I
NK 14,15,25
640 PEN 3:PAPER 0:LOCATE 4,7:PRINT"By Sean McManus
650 PEN 7:LOCATE 9,10:PRINT"Keys:":LOCATE 1,11:PRI
NT"Cursor left & right"
660 PEN 1:LOCATE 5,16:PRINT"Press any key"
670 SYMBOL 255,48,254,22,60,116,210,254,16:LOCATE
20,25:PRINT":
680 BX=5:BY=3:CALL 40000,&8000,BX,BY:LOCATE 4,14:P
RINT"Feat. TRANSOUND"
690 DIM M(19,24):FOR G=0 TO 19:M(G,0)=1:M(G,24)=1:
M(0,G)=1:M(0,G+5)=1:M(19,G+5)=1:M(19,G)=1:NEXT:BDX
=-1:BDY=-1:EI:EVERY 4,1 GOSUB 480:DIM T(20)
700 WHILE INKEY$<>"":WEND:GOSUB 820:G=REMAIN(1):ER
ASE M,T
710 RETURN:DATA 3,26,6,15,10,20,5,25,1,19,21,4,8
720 ' SHEETS
730 s=4:FOR G=2 TO 17:FOR H=2 TO 6 STEP 1:S=S+1:IF
S=7 THEN S=3
740 left=left+1:CALL 40000,a(s),g,h:m(g,h)=s:NEXT
h,g:RETURN
750 s=4:FOR G=2 TO 17 STEP 1.5:FOR H=2 TO 10 STEP
2:S=S+1:IF S=7 THEN S=3
760 left=left+1:CALL 40000,a(s),g,h:m(g,h)=s:NEXT
h,g:RETURN
770 SCN=0:GOSUB 730:FOR G=4 TO 17 STEP 5:FOR H=10
TO 12 STEP 4:M(G,H)=1:LOCATE G+1,H+1:PEN 15:PRINT
CHR$(224):NEXT H,G:INK 15,15,24:RETURN
780 DIM t(19):FOR g=0 TO 6:t(g)=4:NEXT:FOR g=7 TO
12:t(g)=2:NEXT:FOR g=13 TO 19:t(g)=1:NEXT:RESTORE
780:FOR g=1 TO 7:READ a(g):NEXT:RETURN:DATA &8000,
&8022,&8064,&8086,&80a8,&80ca,&80ec
790 LOCATE 1,25:PEN 1:PRINT"*":FOR g=1 TO 16
800 FOR h=1 TO 16:IF TEST(g*2,16-h) THEN m(g+2,h-
1)=4:CALL 40000,a(4),g+2,h-1:left=left+1
810 NEXT h,g:LOCATE 1,25:PRINT" ":CALL 40000,a(7)
,0,24:RETURN
820 ' COSMOS 1 - (C) 1988 Sean McManus
830 A$="":ENV 1,7,-1,5:ENT -1,1,1,1:SOUND 135,0,1,
1
840 SOUND 1,239,40,14,1,1:SOUND 4,0,20,1:SOUND 4,2
39,40,14,1,1:SOUND 2,956,40,14,1,1:SOUND 2,956,40,
5
850 RESTORE 850:s=0:WHILE s<>2 AND A$="":READ s:A$
=INKEY$
860 IF s<>2 AND s<>0 THEN SOUND 1,s,40,14,1,1:SOUN
D 2,s*2,40,14,1,1:SOUND 4,s+5,40,14,1,1 ELSE IF s=
0 THEN SOUND 1,0,40,1:SOUND 2,0,40,1:SOUND 4,0,40,
1
870 WEND
880 SOUND 2,0,40,5:FOR g=14 TO 8 STEP -1:RESTORE 9
20:FOR h=1 TO 12:READ s:SOUND 1,s,40,g,1,1:SOUND 2
,s*2,40,g:SOUND 4,s+5,40,g,1,1:A$=INKEY$:IF A$=""
THEN NEXT h,g
890 WHILE A$="":A$=INKEY$:WEND
900 DATA 478,379,478,379,478,426,568,20,40,478,379
,478,379,478,379,179,239,478,379,478,379,478,379,1
59,478,379,478,379,478,379,142,239,478,379,478,379
,478,426,568,20,40
910 DATA 956,902,956,902,804,851,902,956,902,804,8
51,902,956,900,956,0,900,956,0,900,851,956,0,900,9
56,956,956,0,239,478,379,478,379,478,379,179,239,4
78,379,478,379,478,379,179,478,379,478,379,478,426
,568,1
920 SOUND 135,40,5,15:RETURN:DATA 239,20,253,70,50
6,50,239,20,253,70,506,50,2
930 ' Phew ! Time to save, and then for some seri
ous WALLBUSTIN' !

```



SCNSV

By Sean McManus

Here's a little something solely for you lucky 464 owners. What does it do? Well, judging from the weird title, something decidedly techie. In fact, this proggy will install two new RSXs, |SCNSV and |SCNLD, which save and load respectively, the current screen, with all its windows, pens, inks and what have you. Quite a nifty device, what? Try it for yourself and see what you think.



```

1 ' SCNSV and SCNLD RSXs [71]
2 ' (C) 1991 Sean McManus [72]
3 ' CPC464 only [73]
4 ' [74]
5 ' Do the |tman ? Oh. Maybe not. [75]
6 ' [76]
10 MODE 1:INK 1,26:INK 0,0:BORDER 0:PRINT"CPC464 Screen Saver":PRINT"(C) 1991 Sean
McManus":PRINT:PRINT"Installs two new commands - |SCNSV and |SCNLD which save and load the current
screen with all its windows, pens, inks etc.":PRINT [0E]
20 FOR g=40000 TO 40151:READ a$:a=VAL("&" +a$):POKE g+h,a:chk=chk+a:NEXT:IF CHK<>17239 THEN
PRINT"Error in data":STOP [81]
30 CALL 40000 [14]
40 DATA 01,4D,9C,21,49,9C,C3,D1,BC,5E,A5,4D,9C,56,9C,C3,61,9C,C3,
9F,9C,95,34,3E,53,D6,53,43,4E,4C,C4,00,21,00,C0,11,00,40,3E,16,CD,9E,BC,D2,9A,9C,21,28,B3,11,E,00,3E,16,CD,9E,BC,D2,9A,9C,21,00
[EA]
50 DATA B2,11,B9,00,3E,16,CD,9E,BC,D2,9A,9C,C9,3E,07,C3,5A,BB,21,00,C0,11,
00,40,3E,16,CD,A1,BC,D2,9A,9C,21,C8,B1,11,3F,00,3E,16,CD,A1,BC,D2,9A,9C,21,0C,B2,11,B7,00,3E,16,CD,A1,BC,D2,9A,9C,C9
[06]

```

TORTOISE

By Sean McManus

No, this has absolutely nothing to do with anatomical diagrams of shelled reptilians, before you ask. It does however share the Tortoise's most famous attribute, in that this superb proggy slows down Amstrad BASIC no end. This makes it much easier to spot where errors are occurring, as well as making gameplay painfully easy. Simply run the program, then type |TORT to slow BASIC down and then type |HARE to speed things up again, what else?

```

1 'TORT - A program to slow down Amstrad BASIC [71]
2 ' - Though the French may think it's wrong (-a linguistic jokette) [12]
3 ' (c) 1991 Sean McManus [73]
4 ' [74]
5
DATA
21a69c0681117b9ccdefbc21549c01589cc3d1bc00000000619cc3699
cc3759cc9544f52d4484152c521a09c110500010100c3e9bc21a09cc3ecbcc50
664:'Sorry about that line ! [1A]
6 READ a$:mem=&9C40:FOR g=1 TO LEN(a$) STEP 2:POKE
mem,VAL("&" +MID$(a$,g,2)):mem=mem+1:NEXT:FOR g=1 TO
14:POKE mem,&C5:POKE mem+15,&C1:mem=mem+1:NEXT:POKE
&9C9C,&10:POKE &9C9D,&E0:POKE &9C9E,&C1:POKE
&9C9F,&C9:CALL 40000 [B7]
7 PRINT"New commands":PRINT"|TORT - switches slow mode
on":PRINT"|HARE - switches it off again.":PRINT:PRINT"(c) 1991 Sean
McManus":PRINT [0D]

```

PACLONE

BY SEAN MCMANUS

If you don't know what this little gem of a game is about by now, you must

have had your head in a bush for the last few years. That's right, it's a clone of that old favourite *Pacman*. Run it, follow the on-screen instruc-

tions and you've got a real treat for the ears and eyes. Get tapping and start munching.

```

1 ' Pacclone - (C) 1990/1991 Sean McManus [71]
2 ' Original written July 1990 - Remix 29th September 1991 f [72]
  or ACU [72]
3 ' Listening 2 Anam (!) [73]
4 ' [74]
5 MODE 1:PEN 1:PAPER 0:INK 0.0:INK 1.26:BORDER 0:LOCATE 17.8 [3A]
  :PRINT"PACLONE":LOCATE 8.10:PRINT"By Sean McManus - Sept 199 [3A]
1":LOCATE 8.14:PRINT"Featuring TRANSOUND STEREO":SYMBOL 255. [3A]
48.254.22.60.116.210.254.16 [3A]
6 MEMORY 39999:FOR g=0 TO 44:READ a$:a=VAL("&"+a$):POKE 4000 [58]
0+g,a:chk=chk+a:NEXT:IF chk<>4077 THEN PRINT"Error in very f [58]
irst data line !":STOP [58]
7 DATA DD.6E.00,DD.66.02,CD.1A,BC.DD,5E.04,DD.56.05.01.10.04 [85]
.13.13.C5.E5.1A.AE.77.23.13.10.F9.E1.01.00.08.09.30.04.01.50 [85]
.CO.09.C1.0D.20.E8.C9 [85]
8 mem=40342:FOR g=1 TO 7:chk=0:FOR h=1 TO 77:READ a$:a=VAL(" [94]
&"+a$):POKE mem,a:mem=mem+1:chk=chk+a:NEXT:READ chk$:IF chk< [94]
>VAL("&"+chk$) THEN PRINT"Checksum" g "is wrong.":STOP [94]
9 NEXT [47]
10 DATA 10.04.00.00.00.00.00.00.00.00.00.00.00.00.00.00.FC.FC.0 [C6]
0.40.FC.FC.A8.54.E8.FC.A8.00.00.00.00.04.0C.0C.08.44.CC.CC.8 [C6]
8.44.CC.CC.88.41.C3.C3.82.00.00.00.54.D4.FC.A8.54.FC.D4.A [C6]
8.00.00.00.00.00.00.00.00.10.04.00.00.11.22.00.00.22.00.00.1 [C6]
502 [C6]
11 DATA C0.62.00.40.30.32.80.40.C0.C0.80.05.C0.C0.0A.41.0F.0 [9E]
F.82.54.C3.C3.A8.44.FC.FC.88.40.CC.CC.80.00.C0.C0.00.00.C0.C [9E]
0.00.00.C0.C0.00.00.C0.C0.00.00.C0.C0.00.00.40.80.00.10.04.0 [9E]
0.54.54.00.54.54.54.A8.00.A8.FC.A8.54.FC.A8.A8.54.FC.FC.A8.2 [9E]
112 [9E]
12 DATA 54.FC.FC.A8.00.00.00.00.40.C0.C0.80.40.C0.C0.80.05.0 [60]
F.0F.0A.41.F3.F3.A2.54.FC.FC.A8.44.CC.CC.88.40.C0.C0.80.40.C [60]
0.C0.80.00.00.00.00.10.04.00.00.00.00.00.00.00.00.00.00.00.0 [60]
0.00.00.00.00.00.00.00.00.00.00.10.00.00.00.98.20.00.54.CC.1 [60]
63E [60]
13 DATA 20.00.54.DC.00.00.00.A8.00.00.00.00.00.00.00.00.00.0 [43]
0.00.00.00.00.00.00.00.00.00.00.00.00.00.00.00.10.04.00.00.0 [43]
0.00.00.00.00.00.00.00.00.00.00.54.A8.00.00.FC.54.00.00.FC.7 [43]
C.A8.40.FC.FC.AA.40.FC.FD.00.40.FC.00.00.40.FC.FD.00.40.D4.1 [43]
01C [43]
14 DATA FC.AA.00.D4.FC.A8.00.C0.E8.00.00.40.80.00.00.00.00.0 [5C]
0.00.00.00.00.10.04.00.00.00.00.00.00.00.00.00.00.00.00.00.5 [5C]
4.A8.00.00.A8.FC.00.54.BC.FC.00.55.FC.FC.A8.00.FE.FC.A8.00.0 [5C]
0.FC.A8.00.FE.D4.A8.55.E8.E8.A8.40.D4.D4.00.00.C0.E8.00.00.1 [5C]
B58 [5C]
15 DATA 40.80.00.00.00.00.00.00.00.00.00.00.10.04.00.41.41.00.0 [71]
0.00.82.00.00.82.82.82.41.41.C3.41.41.2A.82.6B.00.C3.4B.82.0 [71]
0.00.82.00.41.41.C3.41.41.C3.0F.C3.00.0F.C3.0A.41.4B.87.C3.0 [71]
5.87.87.87.00.4B.4B.0A.41.C3.41.C3.41.00.00.41.00.00.00.00.1 [71]
320 [71]
16 DATA 10.04.30.20.30.20.30.20.30.20.00.00.00.00.00.20.30.30.1 [B7]
0.20.30.30.10.00.00.00.00.10.30.10.30.10.30.10.30.00.00.00.0 [B7]
0.20.30.20.30.20.30.20.30.00.00.00.00.30.20.30.20.30.20.30.2 [B7]
0.00.00.00.00.00.00.00.00.10.04.00.00.00.00.00.00.00.00.00.6 [B7]
08 [B7]

```

```

17 ON ERROR GOTO 18: !SPED,3: spon=1 [9F]
18 SCORE=0: INK 0,3: ON BREAK GOSUB 57: MODE 0: SPEED KEY 2,2: WH [8D]
ILE INKEY$<>"" : WEND [8D]
19 PAPER 0: PEN 1: LOCATE 8,2: PRINT "PACLONE": LOCATE 2,4: PRINT " [20]
Eat all the chicken nuggets, All the stodgy burgers, all [20]
the slurry pots and all the french straw": CALL 40000,40606,1 [20]
0,10: PEN 7: LOCATE 5,14: PRINT "Keys: A Z N M": PEN 4 [20]
20 LOCATE 5,25: PRINT "Press any Key": INK 0,3: LOCATE 1,17: PEN [18]
1: CALL 40000,40738,10,18: PRINT "Avoid the Tummybugs": LOCATE [18]
6,23: PRINT "HISCORE": "HI: LOCATE 20,25: PRINT CHR$(255): [18]
21 RESTORE 45: INK 15,24,0: INK 7,24: INK 9,19: INK 10,18: INK 2, [CE]
6: INK 3,15: INK 1,26: BORDER 3: INK 4,10: SPEED INK 10,10: INK 14 [CE]
,19: WHILE INKEY$<>"" : WEND: GOSUB 71 [CE]
22 INK 0,0: MODE 0: RESTORE 22: FOR G=1 TO 3: READ X(G),Y(G),D(G) [1F]
): CALL 40000,40738,X(G),Y(G)*2: NEXT: FOR G=1 TO 4: READ DP(G), [1F]
DPY(G): NEXT: DATA 4,3,4,12,6,1,3,8,1,0,1,0,-1,1,0,-1,0 [1F]
23 SPEED INK 10,10 [0C]
24 DIM T(20): FOR G=1 TO 7: T(G)=4: NEXT: FOR G=8 TO 13: T(G)=2: N [CA]
EXT: FOR G=14 TO 20: T(G)=1: NEXT [CA]
25 DIM M(19,11) [D6]
26 FOR y=0 TO 11: READ a$: FOR x=0 TO 19: CALL 40000,VAL(MID$(a [8A]
$,x+1,1))*66+40210,x,y*2: M(X,Y)=VAL(MID$(a$,x+1,1)): NEXT x,y [8A]
: EVERY 13,1 GOSUB 58: X(9)=19: Y(9)=1: M(19,1)=137: CALL 40000,4 [8A]
0738,19,2: EVERY 75,2 GOSUB 66: M(10,7)=0: CALL 40000,40540,10, [8A]
14 [8A]
27 FOR G=1 TO 3: M(X(G),Y(G))=M(X(G),Y(G))+128: NEXT [73]
28 F=217: X=10: Y=7: S=6: WINDOW 1,20,25,25: PAPER 11: INK 11,3: CL [1C]
S [1C]
29 CALL 40000,S*66+40210,X,Y*2: WHILE F>0: OX=X: OY=Y: OS=S: WHIL [B4]
E (OX=X AND OY=Y) AND M(X,Y)<128 [B4]
30 EI: A$="" : WHILE (INSTR("AZNM",A$)=0 OR A$="") AND M(X,Y)<1 [52]
28: A$=UPPER$(INKEY$): WEND: DI [52]
31 IF M(X,Y)=128 THEN 41 [E4]
32 DIR=INSTR("ZAMN",A$): X=X+DP(DIR): Y=Y+DPY(DIR): S=(INT(RND* [43]
255)+1)+6 [43]
33 IF Y=5 THEN IF X=-1 THEN X=19 ELSE IF X=20 AND Y=5 THEN X [0C]
=0 [0C]
34 IF M(X,Y)=9 THEN S=OS: X=OX: Y=OY [F1]
35 WEND [22]
36 CALL 40000,OS*66+40210,OX,OY*2: CALL 40000,S*66+40210,X,Y* [77]
2: IF M(X,Y)=5 THEN CALL 40000,40540,X,Y*2: SCORE=SCORE+1: F=F- [77]
1: M(X,Y)=0: SOUND T(X),25,5,15: SOUND T(X),75,5,15 [77]
37 IF M(X,Y) AND 128 THEN 41 [D3]
38 IF M(X,Y)<>0 THEN CALL 40000,M(X,Y)*66+40210,X,Y*2: FOR G= [A2]
50 TO 25 STEP -3: SOUND T(X),G,3,15: NEXT: F=F-25: SCORE=SCORE+2 [A2]
5: M(X,Y)=0: IF spon THEN !SAY,"Yum Yum inmy tum" [A2]
39 WEND [2A]
40 G=REMAIN(1): G=REMAIN(2): GOSUB 82: ERASE M,T: PAPER #7,0: FOR [3F]
G=1 TO 50: PRINT#7,"↓
":NEXT:GOTO 22 [3F]
41 G=REMAIN(1): G=REMAIN(2): PAPER #2,11: PAPER #1,11: WINDOW #1 [8D]
1,20,1,9: LOCATE #1,1,1: WINDOW #2,1,20,9,25: LOCATE #2,1,25:F [8D]
OR G=1 TO 13: PRINT#1,"↑
":PRINT#2,"↓
":NEXT [8D]
42 WINDOW 1,20,1,25: PAPER 11: INK 11,3: PEN 1: A$="Score:"+STR$ [89]
(score): FOR h=1 TO LEN(a$): FOR g=2 TO 12: LOCATE 10-0,5*LEN(a [89]
$)+h,g: PRINT MID$(a$,h,1):"
":NEXT g,h: FOR g=1 TO 1200: NEXT [89]
43 IF SCORE>HI THEN HI=SCORE [79]
44 ERASE T,M: GOTO 18 [9C]
45 DATA "99999999999999999999" [B6]
46 DATA "93555555559955555529" [6E]
47 DATA "95999599555599599959" [4A]
48 DATA "95955555955955555959" [78]
49 DATA "95959995955959995959" [4E]
50 DATA "55559555555555559555" [31]

```

```

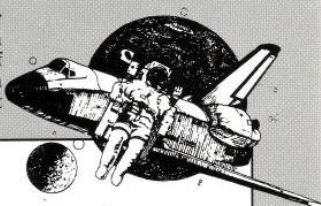
51 DATA "99955555999955555999" [A3]
52 DATA "9555999555959995559" [5D]
53 DATA "9595555595555555959" [1B]
54 DATA "95999599955999599959" [E1]
55 DATA "9255555559955555549" [9A]
56 DATA "9999999999999999999" [B9]
57 SPEED KEY 7,7:PAPER 0:PEN 1:MODE 2:LIST [81]
58 M=M+1:IF M=4 THEN M=1 [6A]
59 IX=X(M):IY=Y(M):M(X(M),Y(M))=M(X(M),Y(M))-128:CALL 40000, [5F]
40738,x(M),y(M)*2 [5F]
60 A=0:B=0:IF RND>0.5 THEN 62 [84]
61 IF X>X(M) THEN A=1 ELSE IF X<X(M) THEN A=-1 [A8]
62 IF A=0 AND Y>Y(M) THEN B=1 ELSE IF Y<Y(M) AND A=0 THEN B= [1F]
-1 [1F]
63 C=M(X(M)+A,Y(M)+B) [91]
64 IF C<>0 AND C<>3 AND C<>5 THEN A=0:B=0:IF RND>0.5 THEN 61 [50]
65 X(M)=X(M)+A:Y(M)=Y(M)+B:CALL 40000,40738,X(M),Y(M)*2:SOUN [B1]
D T(X(M)),478,5,10:M(X(M),Y(M))=M(X(M),Y(M))+128:g=FRE(0):RE [B1]
TURN [B1]
66 OXM=X(9):OYM=Y(9):CALL 40000,40738,X(9),Y(9)*2:DI:M(X(9), [2A]
Y(9))=M(X(9),Y(9))-128 [2A]
67 IF X(9)>x THEN A=-1 ELSE A=1 [6A]
68 IF Y(9)>y THEN B=1 ELSE B=-1 [7A]
69 X(9)=X(9)+A:Y(9)=Y(9)+B:IF x(9)=-1 OR x(9)=20 THEN x(9)=x [86]
(9)-a [86]
70 CALL 40000,40738,X(9),Y(9)*2:SOUND T(X(9)),956,5,15:SOUND [89]
T(X(9)),956,5,10:M(X(9),Y(9))=M(X(9),Y(9))+128:EI:RETURN [89]
71 ' 14/3/89 - acculz [F9]
72 ' [FB]
73 a$="":ENV 1,14,-1,4:RESTORE 75:WHILE a$="" :WHILE s<>2.77 [16]
AND a$="" :a$=INKEY$:READ s:IF s<>2.77 AND s<>1 AND s<>3 AND [16]
s<>2 THEN SOUND 1,s,15,13:SOUND 2,s/2,15,12:SOUND 4,s,5,13:S [16]
OUND 4,s+5,5,13:SOUND 4,s-5,5,12 ELSE IF s<>2.77 THEN ON s G [16]
OSUB 79,80,81 [16]
74 WEND:IF a$="" THEN RESTORE 75:GOSUB 82:s=0:RESTORE 75 [A4]
75 DATA 379,478,506,358,358,379,478,506,379,379,379,478,506, [FF]
358,358,319,319,379,379,379,478,506,358,358,379,478,506,506, [FF]
638,638,568,568,506,506,638,638,379,478,506,358,358,379,478, [FF]
506,379,379,379,478,506,358,358,319,319,379,379,379,478,506, [FF]
358,358,379,478 [FF]
76 DATA 506,506,638,638,568,568,506,506,638,638,379,478,506, [E6]
1,379,478,506,379,379,379,478,506,1,319,319,379,379,478, [E6]
506,1,379,478,506,506,638,638,568,568,506,506,638,638,379,47 [E6]
8,506,2,478,506,379,379,379,478,506,2,319,379,379,379,478,50 [E6]
6,2,478,506,506 [E6]
77 DATA 638,638,568,568,506,506,638,638,379,478,506,1,478,50 [12]
6,379,379,379,478,506,2,319,379,379,379,478,506,3,478,506,50 [12]
6,638,638,568,568,506,506,638,638,2,77 [12]
78 WEND:FOR g=1 TO 7:SOUND 4,153*RND*g,15,14,1:SOUND 1,158+( [33]
100*RND*g),15,14,1:SOUND 2,200,15,14,1:NEXT:FOR g=11 TO 4 ST [33]
EP -1:SOUND 4,153*RND*g,15,g,1:SOUND 1,158+(100*RND*g),15,g, [33]
1:SOUND 2,200,15,g,1:NEXT:RETURN [33]
79 SOUND 4,379,15,15,1:SOUND 1,358,15,15,1:SOUND 2,75,15,15, [8A]
1:RETURN [8A]
80 SOUND 4,379,15,15,1:SOUND 1,358,15,15,1:SOUND 2,75,15,15, [E6]
1:SOUND 4,478,15,15,1:SOUND 1,379,15,15,1:SOUND 2,100,15,15, [E6]
1:RETURN [E6]
81 SOUND 4,379,15,15,1:SOUND 1,358,15,15,1:SOUND 2,75,15,15, [89]
1:SOUND 4,478,15,15,1:SOUND 1,379,15,15,1:SOUND 2,100,15,15, [89]
1:SOUND 4,379,15,15,1:SOUND 1,358,15,15,1:SOUND 2,75,15,15,1 [89]
:RETURN [89]
82 RESTORE 75:FOR g=1 TO 10:READ s:SOUND 1,s,15,15:SOUND 2,s [4C]
/2,15,15:SOUND 4,s/3,15,15:NEXT:FOR g=15 TO 1 STEP -1:SOUND [4C]
1,s,15,g:SOUND 2,s/2,15,g:SOUND 4,s/3,15,g:NEXT:FOR s=400 TO [4C]
379 STEP -1:SOUND 1,s,6,15:SOUND 4,s,2,15:SOUND 4,s+5,2,14 [4C]
83 SOUND 2,0,6,5:NEXT:RETURN [B0]

```

DIGITAL

BY SEAN McMANUS

Watch out for this excellent colour swapping demo, featuring full screen size digits on a countdown to blast off from 99 seconds. Press any key when the program is up and ready and use it for what you will.



```

1 ' Demonstration of colour switching [71]
2 ' Digital Countdown [72]
3 ' (C) 1990 Sean McManus [73]
4 ' [74]
10 ON BREAK GOSUB 100:MODE 0:GOSUB 70:ORIGIN 0,0,0,640,0,400:GRAPHICS PEN 0:REST
ORE:FOR g=1 TO 7:READ x,y:MOVE x,y:FILL g:MOVE x+370,y:FILL g+7:NEXT:DIM digits(
9,7):DATA 90,350,70,250,200,250,90,200,200,100,70,100,90,50 [6F]
20 SYMBOL 255,48,254,22,60,116,210,254,16:LOCATE 20,25:PEN 15:PAPER 0:PRINT CHR$
(255)::FOR digs=0 TO 9:FOR onoff=1 TO 7:READ digits(digs,onoff):NEXT onoff,digs
[E2]
30 WHILE 1=1:CALL &BC02:CALL &BB06:INK 0,10:BOARDER 10:INK 15,26:FOR g=99 TO 0 ST
EP -1:FOR h=1 TO 7:IF digits(INT(g/10),h)=1 THEN INK h,0 ELSE INK h,13 [99]
40 IF digits(g MOD 10,h)=1 THEN INK h+7,0 ELSE INK h+7,13:DATA 1,1,1,0,1,1,0,0
,1,0,1,0,0,1,0,1,0,1,1,1,0,1,1,1,0,0,1,1,0,0,1,1,0,1,0,1,0,1,0,1,1,1,1
,1,0,1,0,1,0,0,1,1,1,1,1,1,1,1,1,1,0,1 [EA]
50 NEXT:FOR pause=1 TO 200:NEXT:SOUND 7,0,5,15:NEXT:FOR g=1 TO 500:NEXT:WEND [5C
]
60 MODE 2:INK 1,26:INK 0,0:BORDER 0:PEN 1:PAPER 0:LIST [0E]
70 RESTORE 70:FOR g=1 TO 4:READ x,y:ORIGIN x,y:GOSUB 80:NEXT:RETURN:DATA 130,290
,130,130,500,130,500,290 [47]
80 FOR x=-1 TO 1 STEP 2:FOR y=-1 TO 1 STEP 2:PLOT 60*x,1*y,1:DRAWR 0,50*y:DRAWR
20*x,20*y:DRAWR 20*x,0:DRAWR 0,-70*y:MOVE -120,50*y:MOVER 60,40*y:DRAWR 110,0:DR
AWR 0,-20*y:DRAWR -20,-20*y:DRAWR -70,0:DRAWR -20,20*y:DRAWR 0,20*y:NEXT y,x [75
]
90 MOVER -x*10,-y*10:FILL 1:MOVER x*25,-y*1:FILL 1:MOVER x*30,-y*10:FILL 1:MOVER
-x*15,-y*30:FILL 1:RETURN [8B]
100 MODE 2:INK 1,26:INK 0,0:BORDER 0:PEN 1:PAPER 0:LIST [C7]

```

Page 18

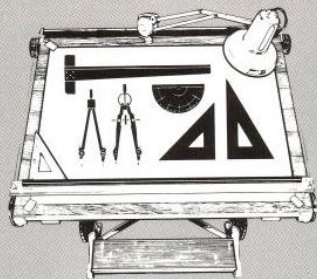
Amstrad User February 1991

3D MAPPER

By Sean McManus

Something for the designer here, with an excellent image creator, using a simple grid system to do your mapping. Run the program, then, us-

ing the keys indicated, simply map out your image on the grid, not forgetting to plot each point as you go. When, you've done that, press X to see a brilliant 3D version of your creation come to life on the screen. It couldn't be more simple.



```

1 ' 3d Mapper - 18th May 1991 [71]
2 ' (C) 1991 Sean McManus [72]
3 ' [73]
10 MODE 1:PAPER 0:INK 0,0:BORDER 0:INK 2,15:INK 3,6:INK 1,26:DIM shape(20,20):FOR x=1 TO 20:FOR
y=1 TO 20:LOCATE x,y:PEN 3:PRINT ".":NEXT y,x:PEN 1:LOCATE 27,2:PRINT"3D Mapper":LOCATE
22,4:PRINT CHR$(164)" 1991 Sean McManus":PEN 2:LOCATE 21,7 [6B]
20 PRINT"8=Up, 2=Down, 4=Left":LOCATE 23,8:PRINT"6=Right, 5=Plot":LOCATE 25,10:PRINT"X -
Process":GOSUB 100 [B2]
30 x=1:y=1:GOSUB 90:WHILE a$<>"X":a$="":WHILE INSTR("
8246X5",a$)<2:a$=UPPER$(INKEY$):WEND:LOCATE x,21:PRINT" ":LOCATE 21,y:PRINT" ":PEN 2:LOCATE
21,7:PRINT"8":IF a$="8" AND y>1 THEN y=y-1 ELSE IF a$="2" AND y<20 THEN y=y+1 [4D]
40 IF a$="6" AND x<20 THEN x=x+1 ELSE IF a$="4" AND x>1 THEN x=x-1 ELSE IF a$="5" THEN
shape(x,y)=shape(x,y)XOR 1:LOCATE x,y:IF shape(x,y) THEN PEN 2:PRINT CHR$(143):ELSE PEN 3:PRINT".";
[E1]
50 GOSUB 90:WEND [36]
60 MODE 1:GOSUB 100:FOR g=1 TO 20:FOR h=20 TO 1 STEP -1 [49]
70 IF shape(g,21-h) THEN FOR z=1 TO 10 STEP 2:PLOT 100+(g+h)*10+z,200+(h-g)*10-z-20:DRAWR
0,20,2:DRAWR 10,10,1:PLOT 100+(g+h)*10+10+z,200+(h-g)*10-(10-z)-20:DRAWR 0,20,3:DRAWR -10,10,1:NEXT
[53]
80 NEXT h,g:END [2B]
90 LOCATE x,21:PEN 1:PRINT"+":LOCATE 21,y:PRINT"+":RETURN [63]
100 SYMBOL 255,48,254,22,60,116,210,254,16:LOCATE 40,25:PEN 1:PRINT CHR$(255)::LOCATE 1,1:RE-
TURN [7D]

```

Page 14

Amstrad User April 1992

CHORDS

BY SEAN MCMANUS



Here's a clever little routine that's designed to test your knowledge of the piano keyboard. So, come on all you budding Dickie Claydermans, enter the skill level, watch for the chord name to appear, then try to beat the computer to the right keys. Not bad at all.

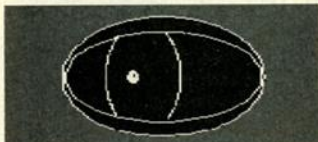
```
1 ' Keyboard Chords Tutor Mark II [71]
2 ' [72]
3 ' (C) 1991 Sean McManus - 8th February 1991 [73]
4 ' [74]
10 MODE 1:INK 1,26:INK 0,3:BORDER 3:PEN 1:PAPER 0:LOCATE 11, [46]
4:PRINT"Keyboard Chords Tutor":LOCATE 12,6:PRINT CHR$(164)"1 [46]
991 Sean McManus":LOCATE 10,15:PRINT"Enter Skill Level (1-5 [46]
)":WHILE INSTR(" 12345",a$)<2:a$=UPPER$(INKEY$):WEND:pause=6 [46]
00+VAL(a$)*200 [46]
20 MODE 0:INK 0,9:BORDER 9:INK 1,13:FOR g=1 TO 7:PLOT 85+g*6 [94]
5,50,1:DRAW 10,10:DRAW 0,250:DRAW -65,0:DRAW 0,-250:DRAW [94]
R 10,-10:DRAW 45,0:NEXT:DIM i(15):FOR g=3 TO 14:READ i:i(g) [94]
=26*i:NEXT:SYMBOL 255,48,254,22,60,116,210,254,16:LOCATE 20, [94]
25 [94]
30 INK 2,19:PEN 2:PAPER 0:PRINT CHR$(255);:FOR g=1 TO 6:IF g [27]
<>4 THEN PLOT 95+g*65,310,0:DRAW 0,-155,0:PLOT 70+g*65,310, [27]
1:DRAW 40,0:DRAW 0,-150:DRAW -6,-6:DRAW -20,0:DRAW -6,6 [27]
:DRAW 0,150:DATA 1,0,1,0,1,0,1,1,0,1,0,1 [27]
40 NEXT:DIM cname$(13),chord(13,3):FOR g=1 TO 12:MOVE 90+g*3 [1B]
5,190:FILL g+2:READ cname$(g):FOR h=1 TO 3:READ chord(g,h):N [1B]
EXT h,g:SPEED INK 4,4:PEN #1,15:PAPER #1,0:INK 15,20:WINDOW [1B]
#1,1,20,2,2:WINDOW #2,1,20,4,4:PEN #2,15:PAPER #2,0:WHILE 1 [1B]
50 FOR i=3 TO 14:INK i,i(i):NEXT:g=INT(RND*11)+1:a=3:b=3:c=3 [9D]
:d=chord(g,3):a$=cname$(g):b$=a$:ON (RND*3)+1 GOSUB 70,80,90 [9D]
:100:CLS #2:LOCATE #2,10-0.5*LEN(b$),1:PRINT#2,b$:CLS #1:LOC [9D]
ATE #1,10-0.5*LEN(a$),1 [9D]
60 PRINT#1,a$:FOR g=1 TO pause:NEXT:INK a,15,24:INK b,15,24: [94]
INK c,15,24:INK d,15,24:FOR g=1 TO pause:NEXT:WEND [94]
70 a=chord(g,1):b=chord(g,2):c=chord(g,3):RETURN:DATA C,10,1 [EA]
4,5,C#/Db,11,3,6,D,12,4,7,Eb,13,5,8,E,14,6,9,F,3,7,10,F#/Gb, [EA]
4,8,11,G,5,9,12,Ab,6,10,13,A,7,11,14,Bb,8,12,3,B,9,13,4 [EA]
80 a=chord(g,1):b=chord(g,2)-((chord(g,2)=3)*12):c=chord(g [2A]
,3):a$=a$+" Minor":b$=b$+"m":RETURN [2A]
90 GOSUB 70:d=chord(g,1)-((chord(g,1)<5)*12):a$=a$+" Seven [0E]
th":b$=b$+"7":RETURN [0E]
100 GOSUB 80:d=chord(g,1)-((chord(g,1)<5)*12):a$=a$+" Seve [E9]
nth":b$=b$+"7":RETURN [E9]
```



```

(CiBj) 10 ' EYE - COLOUR SWAPPING DEMONSTRATION
(CHAs) 20 ' (C) 6/4/92 SEAN MCMANUS
(AtBj) 30 '
(ktAj) 40 '
(PtBw) 50 ON BREAK GOSUB 350:DEFINT a-z:CALL
      ABC02:MODE 0:INK 0,0:BORDER 10
(CWfI) 60 DEG:PAPER 14:PEN 15:FOR g=1 TO
      500:PRINT CHR$(134);:NEXT:INK 14,13:INK
      15,10:FOR g=1 TO 100:PLOT
      320+200*SIN(G),200+100*COS(G):DRAW
      320+200*SIN(360-G),200+100*COS(360-
      G),0:NEXT
(AoDs) 70 PEN 1:PAPER 0:SYMBOL
      255,40,254,22,60,116,210,254,16:LOCATE
      19,24:PRINT
      CHR$(22);CHR$(1);CHR$(255);CHR$(22);CHR$(0
      );
(LuBt) 80 PLOT 1,1,1:DRAW 639,1:DRAW
      639,399:DRAW 1,399:DRAW 1,1
(ISBs) 90 FOR g=225 TO 320:PLOT
      200+75*SIN(g),200+90*COS(g),6:NEXT
(OoBs) 100 FOR g=35 TO 140:PLOT
      200+75*SIN(g),200+90*COS(g):NEXT
(AxBs) 110 FOR g=220 TO 325:PLOT
      360+75*SIN(g),200+90*COS(g),8:NEXT
(CpBt) 120 FOR g=40 TO 135:PLOT
      360+75*SIN(g),200+90*COS(g):NEXT
(JpCq) 130 READ t:FOR z=1 TO t:READ
      i,r1,r2:PLOT 320-r1,200,i:FOR g=-90 TO
      270 STEP 5
(FuCj) 140 IF g=90 AND i>1 THEN IF i=2 THEN i=5
      ELSE IF i=3 THEN i=4
(KjBn) 150 DRAW
      320+r1*SIN(g),200+r2*COS(g),i:NEXT
(MpBl) 160 NEXT:DATA
      3,1,200,100,2,195,75,3,190,50
(HnBt) 170 i=10:FOR g=220 TO 325 STEP 2:IF
      g=240 OR g=304 THEN i=i+1
(OpBl) 180 PLOT
      320+75*SIN(g),200+90*COS(g),i:NEXT
(WuBn) 190 FOR g=35 TO 140:IF g=50 OR g=120
      THEN i=i-1
(IqBl) 200 PLOT
      320+75*SIN(g),200+90*COS(g),i:NEXT
(FrFi) 210 FOR h=1 TO 3:i=5+h:FOR g=1 TO 180
      STEP 14:PLOT
      200+60*h+10*SIN(g),200+10*COS(g),i:DRAW
      200+60*h+10*SIN(360-g),200+10*COS(360-
      g):NEXT:PLOT 200+60*h,205,0:DRAW

```



■ It takes a little while to draw itself on-screen, but Sean McManus's Eye demo is distinctly spooky. Every get the feeling you're being watched...?

```

      2,-2:DRAW 0,-2:NEXT
(JmCk) 220 FOR g=0 TO 13:INK g,0:NEXT:INK
      1,26:INK 10,26:INK 11,26:INK 12,26
(OvBl) 230 INK 2,26:INK 5,26:GOSUB 270:WHILE 1
(GwBs) 240 FOR g=1 TO RND*2000+1000:NEXT:IF
      RND>0.5 THEN GOSUB 310
(BuBi) 250 a=INT(RND*2)+1:ON a GOSUB
      290,300,270
(PxBt) 260 FOR g=1 TO RND*1500+500:NEXT:INK
      6,0:INK 8,0:GOSUB 270:NEND
(AvBn) 270 INK 10,26:INK 11,26:INK 12,26:INK
      7,26:RETURN
(NuBn) 280 INK 10,0:INK 11,0:INK 12,0:INK
      7,0:RETURN
(IAv) 290 GOSUB 280:INK 6,26:RETURN
(MIAu) 300 GOSUB 280:INK 8,26:RETURN
(IICk) 310 FOR g=2 TO 5:INK g,0:INK 14-g,0:INK
      g+1,26:INK 6,0:IF G<3 THEN INK 7,0
(LIBs) 320 FOR H=1 TO 350:NEXT H,G:INK 5,0:FOR
      h=1 TO 350:NEXT
(ArCr) 330 FOR g=5 TO 2 STEP -1:INK g,26:INK
      14-g,26:INK g+1,0:INK 5,26:IF G<3 THEN
      INK 7,26
(LvWx) 340 FOR H=1 TO 350:NEXT H,G:RETURN
(PnBp) 350 MODE 2:PEN 1:PAPER 0:INK 1,26:INK
      0,0:LIST

```

TAKE NOTE!

- Want to know how long a type-in to send in? Well, THE SHORTER THE BETTER! We want to cram as many type-ins into these four pages as possible.
- OK, so you might have a 6128, but OVER HALF OF OUR READERS DON'T. When you write a program for type-ins, you stand a much better chance of getting it accepted if it runs on both the 6128 AND 644.

EYE



Not overly keen on demos, I'm afraid they appear on these pages with an infrequency matched only by ... well, something which isn't very frequent. But when I saw Eye, all that changed.

Written by Sean McManus of Stevenage, Herts, Eye is a colour-swapping and animated graphics demo which is both unusual, technically interesting and well, yes, eye-catching!

What the demo does, is to draw an enormous eye on your CPC's monitor. The eye blinks, swivels from side to side, and generally makes its presence felt.

Beware! Big Brother Amstrad is watching you...

Sprite Driver

Sean McManus of Stevenage, Herts., has come up with a program in response to recent pleas within the pages of AA. His *Sprite Driver* program is just the thing to liven up your BASIC games.

The sprites are full colour XORed sprites (print the same again to delete the original), and are limited to using text co-ordinates. The sprites are referred to as numbers from zero to 126. They must first be created. Use any art package, programming language or sprite definer as long as they can be put up onto the screen from BASIC.

The GRAB command can be used to lift the graphic from the screen and store it for use with the sprites. The information it needs is X co-ordinate, Y co-ordinate, Width in bytes, Height in pixels and address in memory to store it.

```
|GRAB,x,y,w,h,a
```

The sprites are defined with SDEF. It links each Sprite number with the Address it's been grabbed to.

```
|SDEF,s,a
```

To print a sprite onto the screen the SPRITE command is used. Just tell it the Sprite number and the X and Y co-ordinates to position it to.

```
|SPRITE,s,x,y
```

X and Y co-ordinates start at the top left of the screen (0,0) and extend to the bottom right (79,24).

The program launches straight into a demonstration. You can save the RSXs as a block of code with SAVE "code".b.40000.272. Reload it with:

```
MEMORY 39999:LOAD"code":call 40000.

(FjAw) 10:ESD--EASI-SPRITE DRIVER
(KpAs) 20:(C) 1989 SEAN MCMANUS
(PLAu) 30:WRITTEN FOR AMSTRAD ACTION
(MpAp) 40:PUBLIC DOMAIN
(HuAi) 50:
(JLjB) 60:MODE 2:PEN:1:PAPER:0:PRINT>Loading machine code into 40000."
(MxAv) 70:INK:0:0:INK:1,26: BORDER:0
(KjBj) 80:PRINT:PRINT"Easi-Sprite Driver"
(OsAv) 90:PRINT:PRINT"New commands:"
(DwCj) 100:PRINT"(SDEF, sprite number, memory address --Define a sprite address
(LmBj) 110:PRINT"|SPRITE, sprite, x, y --Print sprite
(OxMj) 120:PRINT
(PwAo) 130:PRINT:"Notes:
(RvKx) 140:PRINT:"Co-ordinates used are text co-ordinates minus one."
(DxJx) 150:PRINT:"I.e.: (0-19, 0-24) in mode 0
(GoAx) 160:PRINT:"..... (0-39, 0-24) in mode 1
(MxBi) 170:PRINT:"..... (0-79, 0-24) in mode 2"
(JaAl) 180:PRINT
(HtBi) 190:PRINT:"Do not exceed 126 sprites."
(IwCj) 200:PRINT:"Code is not very error trapped to keep code length to a"
(EjBx) 210:PRINT:"minimum. Use caution. Do not print undefined sprites, or"
(AjBm) 220:PRINT:"sprites with a dimension of zero."

(JmBj) 230:DATA:01,4F,9C,21,6A,9C,CD,D1,3B1
(JkMj) 240:DATA:BC,21,86,9C,C3,D2,9C,5B,4E8
(KrAv) 250:DATA:9C,C3,70,9C,C3,B2,9C,C3,53F
(MxBj) 260:DATA:17,9D,C9,53,50,52,49,54,30F
(MgBl) 270:DATA:C5,53,44,45,C6,47,52,41,341
(FjBj) 280:DATA:C2,00,FC,A6,4F,9C,00,00,34F
(JgKx) 290:DATA:FE,03,C2,CC,9C,DD,6E,00,47E
(OwBj) 300:DATA:DD,66,02,DD,7E,04,F5,CD,466
(FqBl) 310:DATA:1A,BC,F1,E5,6F,26,00,29,36A
(BwKx) 320:DATA:01,4F,90,09,4E,23,46,50,1FD
(KsBj) 330:DATA:59,69,60,4E,23,13,13,46,1FF
(IrBk) 340:DATA:E1,C5,E5,1A,AE,77,23,13,400
(MgBl) 350:DATA:10,F9,E1,01,00,08,09,30,22C
(HjBi) 360:DATA:04,01,50,CD,09,C1,0D,20,20C
(DoAx) 370:DATA:E8,C9,FE,02,C2,CC,9C,DD,588
(AqBk) 380:DATA:7E,02,87,21,4F,9D,4F,06,269
(GsBi) 390:DATA:00,09,DD,5E,00,DD,56,01,278
(BoBk) 400:DATA:73,23,72,C9,21,DD,9C,C3,42E
(EsAx) 410:DATA:D2,9C,7E,FE,00,C8,CD,5A,4D9
(AoBj) 420:DATA:BB,23,C3,D2,9C,18,45,52,3BE
(BoAv) 430:DATA:52,4F,52,18,07,00,45,41,198
(AwBl) 440:DATA:53,59,20,53,50,52,49,54,25E
(FuBm) 450:DATA:45,20,44,52,49,56,45,52,231
(NtBi) 460:DATA:2E,0A,0A,DD,28,43,29,20,103
(ErBj) 470:DATA:31,39,38,39,20,53,45,41,1D4
(MtBk) 480:DATA:4E,20,4D,43,4D,41,4E,55,22F
(KvBk) 490:DATA:53,2E,07,0A,0A,DD,00,FE,1A7
(KjBj) 500:DATA:05,C2,CC,9C,DD,6E,06,DD,45D
(JgKx) 510:DATA:66,08,CD,1A,BC,DD,56,01,345
(KjAx) 520:DATA:DD,5E,00,DD,7E,02,12,4F,2F9
(BoBk) 530:DATA:13,DD,7E,04,12,13,47,C5,2A3
(DoBj) 540:DATA:85,7E,12,23,13,10,FA,E1,396
(AxBi) 550:DATA:01,00,08,09,30,04,01,50,097
(AsBi) 560:DATA:C0,09,C1,DD,20,E9,C9,00,369
(EjAx) 570:a=49C40:L=230:FOR z=1 TO 34
(JLjB) 580:cs=0:FOR x=1 TO 8:READ v$:v=VAL("s"+v$):POKE a,v
(OuBl) 590:cs=cs+v:a=a+1:NEXT:READ c$:c=VAL("s"+c$)
(GlBr) 600:IF c=Ccs THEN PRINT"DATA ERROR IN LINE",L:END

(GiAj) 610:L=L+10:NEXT
(BpAp) 620:CALL:40000
(HkAj) 630:
(LwAp) 640:"Demonstration
(IiAj) 650:
(ALBj) 660:DATA:10,08,00,54,FC,FC,FC,45C
(PpAw) 670:DATA:40,00,00,AB,00,00,00,00,0EB
(HuBi) 680:DATA:E8,00,54,10,30,30,30,30C
(NiAw) 690:DATA:C0,80,AB,74,54,54,54,3AC
(LrBi) 700:DATA:60,00,B8,AB,74,10,54,20,2B8
(MxBj) 710:DATA:E8,AB,FC,20,FC,00,54,AB,4A4
(KrBj) 720:DATA:54,AB,FC,20,30,00,10,20,278
(IjAx) 730:DATA:54,AB,FC,20,00,00,00,00,218
(OtAx) 740:DATA:54,AB,FC,00,00,00,00,00,1F8
(KLBi) 750:DATA:54,AB,FC,00,00,00,00,00,1F8
(CrBj) 760:DATA:54,AB,FC,20,20,20,20,20,298
(JrAx) 770:DATA:74,AB,AB,AB,AB,AB,AB,AB,52C
(OuAx) 780:DATA:AB,AB,B8,B8,54,AB,FC,10,4C8
(EgAx) 790:DATA:AB,AB,54,74,74,B8,FC,74,4B4
(AxXj) 800:DATA:74,00,00,B8,10,10,20,00,16C
(FuBj) 810:DATA:AB,00,00,54,FC,FC,FC,4EC
(EpAv) 820:a=48000:L=660:FOR Q=1 TO 16

(OwBs) 830:cs=0:FOR x=1 TO 8:READ v$:v=VAL("s"+v$):POKE a,v
(BvBn) 840:cs=cs+v:a=a+1:NEXT:READ c$:c=VAL("s"+c$)
(FkBs) 850:IF c=Ccs THEN PRINT"DATA ERROR IN LINE",L:END

(IjAp) 860:L=L+10:NEXT
(HwAo) 870:INK:7,25:INK:4,13
(IjAr) 880:MODE 0:|SDEF:1,48000
(KrAj) 890:PEN:1:PAPER:0
(JxBm) 900:LOCATE 3,1:PRINT"ESD-Demonstration"
(MxAu) 910:LOCATE 7,2:PRINT"(Batman)
(NrAr) 920:FOR n=23 TO 5:STEP -2
(JLjB) 930:FOR x=0 TO 19:STEP 2:|SPRITE:1,x,0
(BxAc) 940:FOR y=1 TO n
(AkAj) 950:|SPRITE,1,x,y:|SPRITE,1,x,y-1
(NvAm) 960:NEXT y,x,n
(EmAj) 970:WHILE INKEY$=""
(PsBr) 980:|SPRITE,1,(INT(RND*10)*2),(INT(RND*10)*2)+3
(HuAk) 990:WEND
(OmAc) 1000:MODE 2:LIST
```

ENVLIST

BY SEAN McMANUS

This neat little proggy installs two RSX's ENVLIST and ENTLIST, which prints out a list of the sound envelopes set up, to see which are free and which are set up without having to scour through screenfulls of program code.



```
1 ' ENVELOPE LISTER [71]
2 ' (C) 1989 SEAN MCMANUS [72]
3 ' [73]
10 MODE 2:INK 1,26:INK 0,0:BORDER 0:PRINT"ENVELOPE LISTER UTILITY":PRINT:PRINT:P
RINT"By Sean McManus":PRINT:PRINT:PRINT"New commands":PRINT"|ENTLIST; |ENVLIST.
":PRINT:PRINT"Machine code loading into 40000-40277":PRINT"Himem:39999":PRINT:PR
INT [A3]
20 MEMORY 39999:ADDR=40000:FOR G=1 TO 7:CHK=0:FOR H=1 TO 40:READ AS:D=VAL("&"+AS
):POKE ADDR,D:CHK=CHK+D:ADDR=ADDR+1:NEXT:READ CHK2$:IF VAL("&"+CHK2$)<>CHK THEN
PRINT"ERROR IN LINE"G*10+180:END [57]
30 NEXT:CALL 40000 [A1]
40 DATA 01,4D,9C,21,49,9C,C3,D1,BC,FC,A6,4D,9C,56,9C,C3,CE,9C,C3,65,9C,C9,45,4E,
54,4C,49,53,D4,45,4E,56,4C,49,53,D4,00,CD,1D,9D,1346 [73]
50 DATA 21,90,9C,36,C2,21,DC,9C,CD,4D,9D,3E,01,CD,6F,BB,06,01,C5,78,C6,03,CD,72,
BB,3E,01,CD,6F,BB,C1,C5,78,CD,FF,9C,C1,78,C5,CD,1539 [32]
60 DATA C5,BC,DC,A1,9C,C1,78,3C,FE,10,CA,42,9D,47,C3,7A,9C,E5,C5,3E,20,CD,5A,BB,
C1,E1,7E,FE,00,C8,46,3E,00,C6,03,10,FC,47,23,E5,155E [8C]
70 DATA C5,7E,CD,FF,9C,C1,E1,05,C8,23,E5,C5,3E,2C,CD,5A,BB,C1,E1,C3,B7,9C,CD,1D,
9D,21,90,9C,36,C5,21,F0,9C,C3,70,9C,41,4D,50,4C,1660 [DE]
80 DATA 49,54,55,44,45,20,45,4E,56,45,4C,4F,50,45,53,2E,54,4F,4E,45,20,45,4E,56,
45,4C,4F,50,45,53,2E,16,64,CD,0B,9D,16,0A,CD,0B,B96 [10]
90 DATA 9D,16,01,0E,00,92,38,03,0C,18,FA,82,F5,79,C6,30,CD,5A,BB,F1,C9,3E,02,CD,
0E,BC,3E,01,CD,90,BB,3E,00,CD,96,BB,3E,01,01,1A,1013 [15]
100 DATA 1A,CD,32,BC,3E,00,01,00,00,CD,32,BC,01,00,00,C3,38,BC,3E,0A,CD,5A,BB,3E
,0D,CD,5A,BB,C9,7E,FE,2E,C8,CD,5A,BB,23,C3,4D,9D,10CB [23]
```

Tutorials

```
10 INPUT "Are you interested in colour  
swapping? ", answer$  
20 IF LEFT$(UPPER$(answer$),1)="Y"  
THEN PRINT "Goodie!"  
30 IF LEFT$(UPPER$(answer$),1)="N"  
THEN PRINT "Oh, that's a shame!"  
40 PRINT "Here's my article about it."
```

London by night – this is all you need to have seen to have the tools at your disposal for fast BASIC animation. But let's start with something simpler: junior school education.

Most of us have fond memories of making cartoons at school, by drawing two slightly different pictures, and then flicking rapidly between them. This is, in effect, two-frame animation—a cartoon made from two static pictures. To simulate this on the CPC we have to be able to 'flick' between two images on screen instantaneously. At this point, machine code programmers generally grin smugly, fold their arms and watch their BASIC counterparts wallow in failure.

The problem is that BASIC takes too long to draw a frame. Certainly any design worth animating would take at least several seconds to draw, shattering the illusion of motion. Perhaps we're approaching the problem the wrong way...

In mode 0 (let's use this mode – the reason will become obvious later), you can have 16 different colours, including the 'background', from a possible 26, on screen at any given time. Pens are used to draw, and the pens are filled with INKS, using the INK command. If you're not familiar with these commands, now would be a very good time to reach for the manual. Pens can be made to flash by assigning two inks to them. The computer will automatically alternate the inks – changing all the occurrences of the pen wherever it is on the screen. Instantaneously. Problem solved!

So, we can make the computer do our 'flicking' for us, by drawing our two images in different pens, and assigning the inks to flash. For example, if we have two frames, one drawn in pen one and the other drawn in pen two, we might assign the inks as follows:

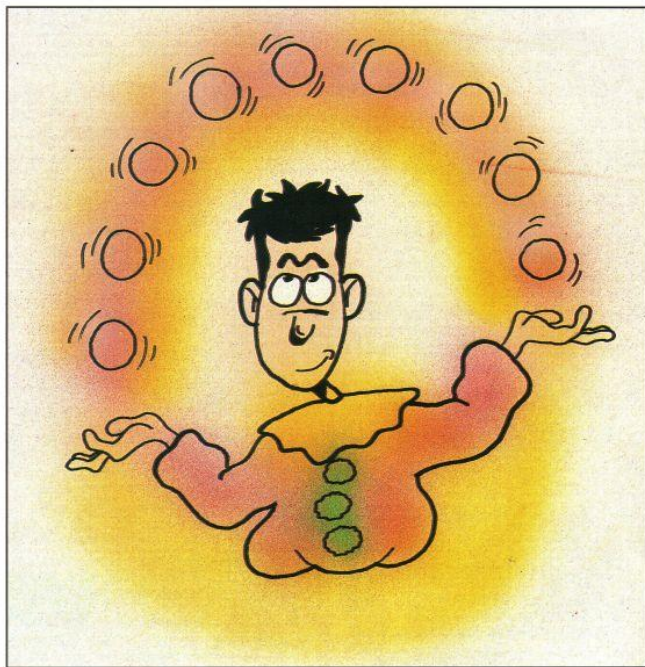
```
INK 1,26,0:ink 2,0,26
```

This assumes that 0 is the background colour. Pen 1 will be 26 (bright white), while pen 2 is the background colour (hidden). Then, pen 2 will become visible, as pen 1 is hidden. This 'flicking' between the images will continue until stopped, in the background of whatever else is happening.

Try DEMO I. It shows several animated jugglers. Upon running, you will be able to see the design as it appears in the default inks. Press any key to start. Parts of the design that are common to both frames are drawn in a pen which does not flash – it remains

A VERY MOVING EXPERIENCE

Basic Animation is the game; Sean McManus is the name. Read on and pick up some important programming tips.



the same colour in both frames. As the scrolling message says, once the design is drawn, the computer can get on with other business, without interrupting the animation. The speed of the animation can be altered using the SPEED INK command. Can you imagine how painful it would have been to write this fairly simple demo without colour swapping?

Whilst 'Roland on the Ropes' was deemed trendy in its day because the

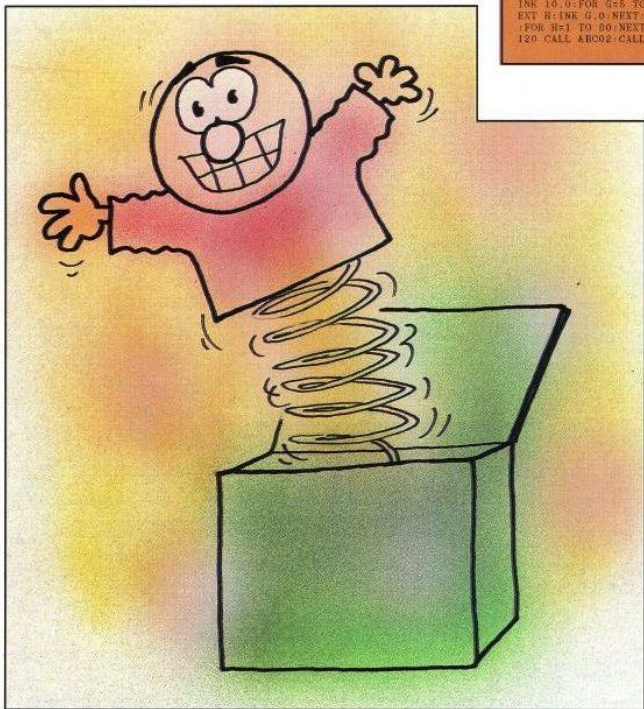
words flashed different colours between loading blocks, we now know how to do some very interesting interblock animations, without too much effort.

But all this only allows for two-frame animation. Let's return to London (yes, wondering about that, weren't you?): Piccadilly Circus by night and its towering facades of neon advertising. Very effective. Yet, each individual panel is quite simple. Shaped tubes are

DEMO I - The juggler

```

10 ' Juggler - colour switching DEMO I [F1]
20 ' (C) 1991 Sean McManus [F2].
30 ' [F3]
40 ON BREAK GOSUB 120 [9B]
50 DATA #EEEEEEEEEEEEFFFO,0EEFFEEFFEEFFFO,0000EE4444FF0000 [D3]
60 0,0000044440000000,00000666600000,000A888889000000,99AA [D3]
70 008880099AA0,39A0088660099A0,00660066B8006000,0006666666 [D3]
80 660001,0000097777000001,0000017777000001,00009777700000 [D3]
1 [D3]
60 DATA 0000322222220001,000003322000091,000600333000000 [09]
70 0,000000000000000 [09]
80 TO CALL ARBOE:Y(0)=100:X(1)=500:Y(1)=300:Y(0)=100:MODE 0:RES [F2]
90 TORS 50:FOR #=1 TO 16:READ #S:FOR #=1 TO 16:FOR #=0 TO 1:FOR [F2]
100 #=0 TO 1:PLOT #+4*(#),Y#+4*(#),VAL("E"*MID$(#S,1)):(PLOT [F2]
110 #S:NEXT #.A:TAG MOVE 190*#+16,60*#+16:PRINT CHR$(143):TAG [F2]
120 OFF:NEXT #.Y [F2]
80 SYMBOL 255,48,254,22,60,116,210,254,16:PEN 1:LOCATE 20,25 [F2]
90 PAPER 0:PRINT CHR$(157): [F2]
90 PRINT CHR$(23):CHR$(1):DEG:TAG:#S:"Juggle":FOR #=1 TO LEN [0A]
100 (#S):FOR #=1 TO 13:PLOT -10,-10,1:MOVE 310+200*#IN(#S+15):15 [0A]
110 #RND#20,160+200*COS(#S+15+315)+RND#20:PRINT MID$(#S,#,1):NE [0A]
120 XT 1,E:TAGOFF [0A]
100 CALL ARBOE:INK 15,10,3:INK 14,3,10:INK 3,15:INK 2,6:INK [0A]
110 7,24:INK 6,26,2:INK 11,2,26:INK 9,2,0:INK 10,3,2:INK 0,3:INK [0A]
120 4,10:INK 15,24,3:INK 12,3,24: BORDER 3 [0A]
130 #S="This juggler is now animating himself in the juggler [49]
140 and - the only processing to this scrolling message. "WHIL [49]
150 #=1:FOR #=1 TO LEN(#S):LOCATE 2,23:PRINT MID$(#S,#,1):N [49]
160 EXT WEND [49]
170 MODE 2:PEN 1:PAPER 0:CALL #C02:LIST [F2]
    
```



lit and unlit, in sequence. This does not sound dissimilar to an approach we can adopt for animation on the CPC. Our tubes become lines and shapes, and our lighting is achieved by colour swapping. In Mode 0 we have 15 different colours (potential frames) at our disposal.

Enter Demo II. An animated Jack in the Box - a potential nightmare in BASIC - becomes quite simple by applying our 'neon logic'. Upon running, inks (=

frames) one to five are set to the background colours, to hide them while they are being drawn. Pen 8 is used to display a 'Please Wait' message, and our two-frame animation comes into its own with the little metronome that ticks away in front of us while the computer busies itself with the jack drawing. Each jack takes about 20 seconds to draw - well beyond the limit of normal animation. As it now stands, FOR-NEXT loops have been used to SLOW down

DEMO II - Jack in the Box

```

10 ' Jack In the Box - DEMO II [F1]
20 ' (C) 1991 Sean McManus - 24th September [F2]
30 ' [F3]
40 ON BREAK GOSUB 120:MODE 0:SYMBOL 255,48,254,22,60,116,210 [66]
50 254,16:PEN 13:LOCATE 20,25:PAPER 0:PRINT CHR$(157):INK 13, [66]
60 23 [66]
50 #=320:Y=300: BORDER 0:INK 0,0:INK 0,26:LOCATE 1,8:PEN 8: [C0]
100 #=0:PRINT #+55:"NEXT Wait":FOR #=1 TO 5:INK #,0:NEXT # [C0]
110 #=0:INK 0,6:DRAW 50,0:DRAW 0,10:DRAW -20,70:DRAW -10,80 [C0]
120 #=0:INK 0,6:DRAW 50,0:DRAW 0,10:FOR #=1 TO 0:PLOT 74,150*#+6:DRAW [C0]
130 #=0:INK 0,6:DRAW 50,0:DRAW 0,10:FOR #=1 TO 0:PLOT 74,150*#+6:DRAW [C0]
140 4,0 [C0]
80 NEXT PLOT 105,162:DRAW -30,-42,14:DRAW -30,42,175:INK 15 [22]
100 #=0:INK 14,0:20:STEP:INK 15,16:FOR LOOP#5 TO 1:STEP -1:#= [22]
110 320:Y=140: LOOP#5:#=1:PLOT 270,100,loop:DRAW 270*50*#1 [22]
120 N(280*#),100+50*COS(280*#):PLOT 380,100,loop [22]
130 TO DRAW 300+50*#IN(90-#),100+50*COS(90-#):PLOT #,Y:DRWA [5B]
140 #=0:DRAW -20,-50:DRAW 5,-5:DRAW 51,0:DRAW 6,6:DRAW -20,5 [5B]
150 #=0:DRAW -5,5:-DEG:FOR #=0 TO 2:FOR #=1 TO 360 STEP 40:PLOT # [5B]
160 #SIN(#)+6,Y#COS(#):#20:DRAW #+#SIN(#)+5,Y#COS(#)+#20 [5B]
170 #EXT #,# [5B]
80 PLOT #+30,Y-60:FOR #=0 TO 270 STEP 5:DRAW #+35*#IN(4)+5, [06]
100 #+25*COS(#):NEXT DRAW #,22,50: PLOT #+15,Y-120:DRAW 40, [06]
110 #=0:FOR #=1 TO 2:PLOT #+45*#20,Y-140:DRAW 10,20:DRAW 10, [06]
120 #EXT DRAW 10,20:DRAW 10,-20:DRAW -50,0 [06]
80 PLOT #+30,Y-140:DRAW 10,10:DRAW -50,30:DRAW -20,17,0 [06]
100 #=0:DRAW 10,10:DRAW 10,-50:DRAW -10,-50:DRAW 10,-10:DRAW 10, [06]
110 #EXT DRAW 10,10:DRAW 10,-50:DRAW -10,-50:DRAW 10,30:DRAW -2 [06]
120 0,15:DRAW 10,-30 [06]
100 #=0:FOR #=1 TO 360 STEP 16:PLOT #+10*#IN(6)+120,Y-100+10*CO [0E]
110 #SIN(#)+10,110,Y-100+10*#COS(#):NEXT #,loop:FOR #=1 [0E]
120 TO 100 STEP 2:PLOT 0,0:DRAW 270,0,0:DRAW 110,0,0:DRAW 110, [0E]
130 0,0:INK 1,2,4:WINDOW 61,1,6,1,25:PAPER 0:CLS 61:LOCATE [0E]
140 1,22:PEN 10 [0E]
110 PRINT #+55:"PRINT" #+55:"WHILE 1:INK 10,6:CALL ARBOE: [52]
120 INK 10,0:FOR #=0 TO 1:STEP 1:INK 6,26:FOR #=1 TO 120:#=15: [52]
130 EXT B:INK 6,0:NEXT INK 1,26:CALL ARBOE:FOR #=1 TO 5:INK 6,26 [52]
140 #=0:FOR #=1 TO LEN(#S):INK 6,0:NEXT WEND [54]
150 CALL ARBOE:CALL ARBOE:MODE 2:PEN 1:LIST [CA]
    
```

its bounce! Note the technique of hiding a design in progress, to preserve its impact with spontaneity. Press any key to launch Jack, and another key to recall him. Repeat until bored, and then break. At the end, all the inks will be reset, revealing all frames.

The Jacks are drawn from bottom to top, since they overlap and the top one must be the best. It is possible to plot the crossover points in a pen that is lit in both frames, and allow special programming for this. (See DEMO I's 'Juggle' text.) For an animation of this speed, it's hardly worth it.

These two programs demonstrate the power at the Basic programmer's fingers when he 'Thinks Neon' and uses colour swapping to bring screens alive.

Now, it's up to you. Work on your own designs, or pick and change on the two demos we've looked at here. If you're happy with the result, why not send it in to ACU for the crew to have a look at. The best piece of simple animation will be rewarded with a special ACU certificate, to boot.

So, get programming and let's see those demos soon.

This is my super-dooah ACU animation entry:

Demo name.....

My name.....

My address.....

.....

.....

Please send this coupon to:
 ACU, Sunnyside Cottage, Carluddon, St. Austell,
 Cornwall PL26 8TY

PROGRAMMING

Listings that don't work? 10-Liners that crash? Usually, if everything has been typed in correctly, it is a problem of compatibility. The vast majority of programs nowadays are written on a CPC6128, causing occasional problems for 464 users. This article will show you how to convert most 128 programs for its predecessor, opening up even more public domain software and rescuing half-working listings you have slaved for hours over. Stand back, nothing up my sleeves, as I prove that (most of the time) 6128 into 464 does go!

The 464 and 128: what are the differences? A disc drive, 64k, 0.1 of Basic and 5664.

The 6128 has a disc drive built in. The main problem is the CAT command. On the 6128, this will directory the disc, while on the 464, this will catalogue the tape indefinitely. CAT commands should be removed, as should all other disc commands, for example: |DISC, |TAPE, |USER, |REN, |ERA, |A, |B.

Filenames on disc also have a three character extension, which is optional on saving. If

omitted, the computer will add .BAS or .BIN as appropriate. Upon reloading, specifying the extension (if it is one of the above) is also optional.

This causes a problem for tape users if the extension is specified on reloading but not on saving (or vice versa), but this is easily solved - simply remove the extension (the full stop and the three characters after it) in all filenames.

The system variable DERR is used for identifying disc errors. Such a routine ought not be required by a tape system but the easiest safety catch is to simply replace DERR with ERR (a valid system variable on both machines for Basic errors).

Our second difference was 64k. There are no two ways about it, I'm afraid. If your program is trying to use the extra 64k of RAM, it will not work

on your 464 without a RAM expansion. The following commands are from the Bankmanager utility supplied with the 6128 and, if you should see them, the program is trying to use the extra memory: |SCREENCOPY, |SCREENSWAP, |BANKOPEN, |BANKREAD, |BANKWRITE, |BANKFIND.

Our next difference was 0.1 of a Basic. The original 464 Amstrad had version 1.0 of Basic, while the 6128 features an upgraded version 1.1. This has some of the original's features altered slightly and fea-

tures several new commands.

Let's begin with the modifications. Basic 1.0 had a bug in it (or at least we are going to concentrate on this one), whereby placing bars (!) in REM statements often had unpredictable side effects. For this reason, you should never place a bar in a REM statement on the 464.

The 464 Basic was also rather particular about where you placed your DATA statements. They had to be at the end of a line. Not so on the 6128, where they may be found at the start or the middle of the line. The correct conversion is to place the data statement as the last statement on the line,

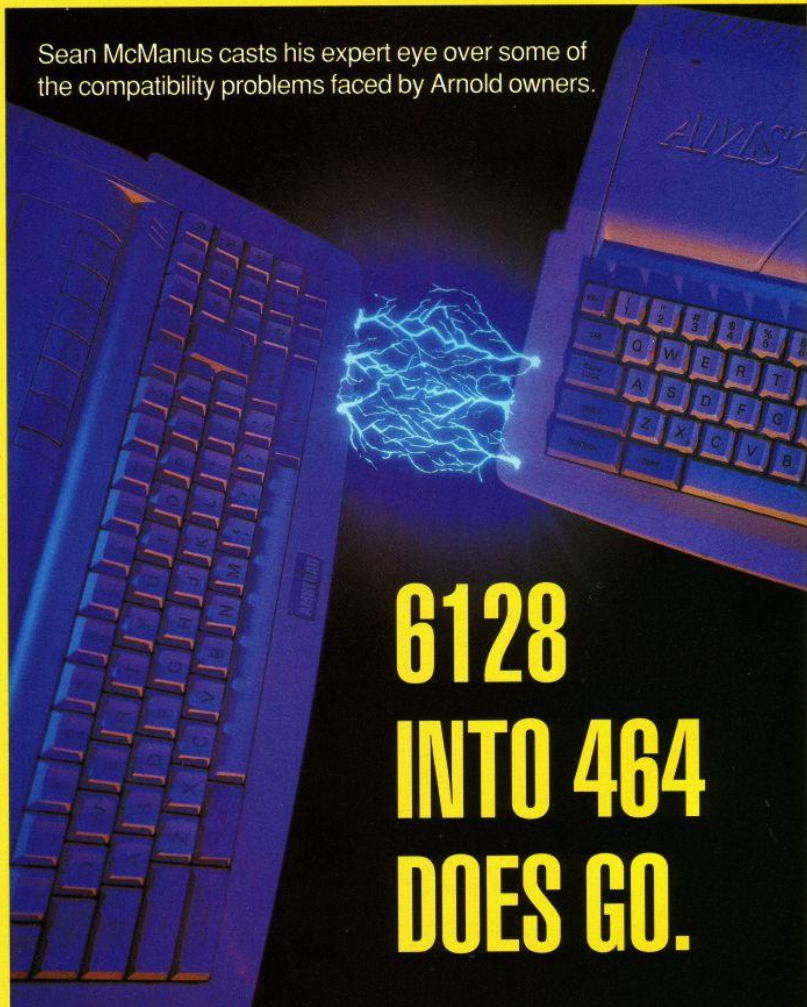
moving everything else along, but keeping all the other commands in the same order. For example:

```
10 DATA
1989,1990,1991,1992:
CLS:PRINT "Deputy
Dog" [6128]

10 CLS:PRINT "Deputy
Dog":DATA
1989,1990,1991,1992 [464]
```

Passing data to RSXs became much more user friendly with the introduction of the 6128, though it may prove quite user irritating to convert them all back. On the 464, string parameters had to be passed via vari-

Sean McManus casts his expert eye over some of the compatibility problems faced by Arnold owners.



6128 INTO 464 DOES GO.

ables, using the @ symbol, as in this example:

```
| PRINTOUT,@a$
```

The @ feeds the memory address of the variable to the RSX, but don't worry if I am losing you - the actual adaptation is easy. 6128 owners do not have to use variables, neither do they have to use the @ symbol. If an RSX is being used with variables in a 6128 program, insert a @ before each variable name (that is, after each comma). If no variable is being used, create a dummy one, making sure that you don't use a name which is already in use - I suggest you use an obscure name to be on the safe side, such as DUMMY\$. Look at these examples:

```
| PRINTOUT , a $
[6128]
| PRINTOUT , @ a $
[464]
```

```
| ADD,a,b,c [6128]
| ADD , @ a , @ b , @ c
[464]
```

```
| PRINTOUT,"HelloSid"
[6128]
| dummy$="Hello
Sid";| PRINTOUT,dummy$
[464]
```

We now come to the extra commands of the 6128. Some of these are very easy to change and can be replaced by POKE commands or CALLs. The more difficult commands we will need programs to emulate.

Our first easy command is FRAME. This has the effect of waiting for frame flyback and makes animation appear smoother. This can be replaced by CALL &BD19 for exactly the same effect. The command never takes any parameters.

Another command we can deal with in a similar way is CLEAR INPUT, which clears the input buffer of all typed characters. This can be replaced by CALL &BB03.

The 6128 has a command DEC\$, which will format a number in a string to the format given. This can be replaced by STR\$, and output using

PRINT USING. See the following example:

```
P R I N T
DEC$(11^5,"###",##")
[6128]
```

```
PRINT USING
"###",##";STR$(11^5)
[464]
```

The 6128 has, in addition to ON BREAK GOSUB and ON BREAK STOP, the command

ON BREAK CONT. This is emulated by creating a dummy line with the command RETURN and ON BREAK GOSUB-ing it. Upon breaking, the program will GOSUB our line, and RETURN straight away, having done nothing. This is the same as ON BREAK CONT. The example shows how:

```
10 ON BREAK CONT
20 REM the rest of the program
is in here [6128]
9090 REM the last line of
the main program
```

```
10 ON BREAK GOSUB
9091
20 REM the rest of the program
is in here [464]
9090 REM the last line of
the main program
9091 RETURN
```

All of the plotting commands have an optional ink mode parameter, which follows the obligatory x and y coordinates/offsets and the optional ink. For example, all of the following are legal in 6128 Basic:

```
DRAW 320,200,1,4:PLOT
7,4,2,0:PLOT 200,201
```

The extra parameter is accommodated by printing CHR\$(23) before it and then printing its own CHR\$ before plotting. See the following examples:

```
DRAW 320,200,1,4
[6128]
P R I N T
CHR$(23)CHR$(4):DRAW
320,200,1 [464]
```

```
PLOT 200,100,2,0
[6128]
P R I N T
CHR$(23)CHR$(0):PLOT
200,100,2 [464]
```

Perhaps the most interesting of these is MOVE. It too may have an ink and an ink mode parameter - funny when you think of it in terms of logic. The command means "Go to coordinate x,y and don't plot a point there in this colour in this mode". The 464 MOVE permits neither an ink mode parameter (deal with as above), nor a graphics ink parameter - solution to follow.

The 6128 has two new commands to set the graphics ink and the graphics paper without plotting anything, namely GRAPHICS PEN and GRAPHICS PAPER. The following short program will emulate the former:

```
10 rem GRAPHICS PEN
emulator - 1991 Sean
McManus
20 data dd,7e,00,c3,de,bb
30 addr=40000:for g=0 to
5:read a$:poke
addr+g,val("&" + a$):next
```

This routine, once run, will make CALL 40000,i set the graphics pen to i. For example:

GRAPHICS PEN 5 on the 6128 becomes, after the above program is run, CALL 40000,5

This routine is completely relocatable - you can put it anywhere you choose by changing the value of addr in line 30 and changing the address you subsequently call. I suggest that the routine is added to the program at the beginning, so it need only be run once, even though the CALLs to it may be multiple.

GRAPHICS PAPER is emulated by changing the "de" in the data to "e4". If you need both these commands, remember to poke the routines into two separate places for calling.

The 6128 command COPYCHR\$ calls for similar treatment. It looks at the screen

and produces the ascii value of the text character at the cursor's location in the given stream. The following program will emulate it:

```
10 REM 6128 Copychr$ emulator
20 data dd,7e,00,cd,b4,bb,ff,
cd,60,bb,32,00,00,f1,c3,b4,bb
30 addr=40000:for g=0 to
16:read a$:poke
addr+g,val("&" + a$):next
```

The 6128 version takes the format COPYCHR\$(#s), where s is the stream. The call takes the format CALL 40000,s (another relocatable routine). It returns its result in location 0, which must be peeked. See the following adaptations (which assume that the routine above has already been run):

```
a=COPYCHR$(#5)
[6128]
C A L L
40000,5:a=PEEK(0) [464]
```

```
PRINT COPYCHR$(2);
[6128]
CALL
40000,2:PRINT
PEEK(0); [464]
```

The 6128 also has a FILL command, which requires quite a complex program to emulate it. There are thousands of fill routines in the Public Domain (and doubtless a few in back issues of ACU), so there hardly seems any point in reproducing one here, especially since they are rarely relocatable, and we are aiming for maximum flexibility here. This is why I could not use RSXs - they might have clashed with machine code in the program and would not be relocatable. If you know your way around the firmware, you might like to add RSXs as appropriate.

Our final difference? 5664. Oh, that was supposed to be a joke. No, not very funny was it?

Armed with this knowledge, 464 users should be able to throw open the doors to even more PD and even more listings. All we need now is someone to tell us how to run console games on our machines. Any takers?

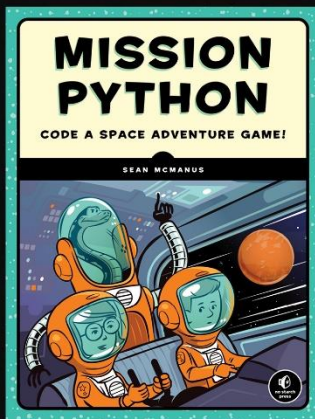
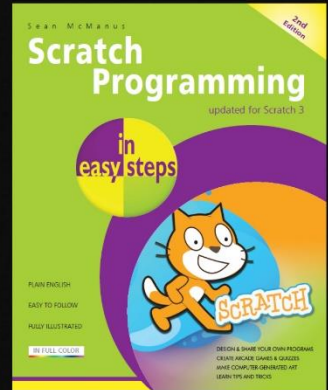
Learn to code

Great books for your PC or Raspberry Pi!

Scratch Programming in Easy Steps

Start programming today!

Discover the building blocks of coding as you create games, digital art, music projects, and more!



Mission Python

Build a space adventure game!

Mission Python shows you how to make an arcade adventure using the popular (and free!) Python programming language.

Find free chapters and more details at www.sean.co.uk/books

Hacking The Further Adventures of Fred

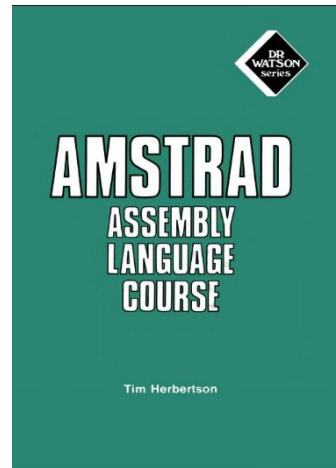
I hacked my own game to tweak the difficulty, get rid of an annoying sequence, and change the default keys.

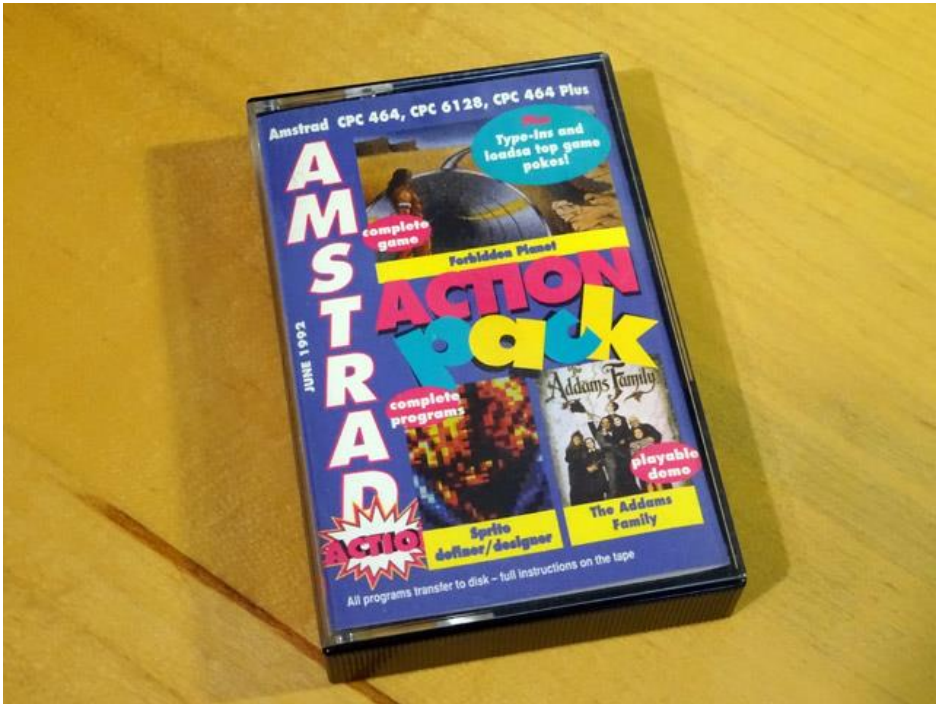
I was surprised to stumble upon Dr Watson's Amstrad Assembly Language Course in the indoor market where I had a Saturday job. It was in the glass display case, along with an oddball selection of tape games for popular machines of the day. I think it was the only serious software they had, presumably ordered in error. I put it on my Christmas list, and was taking my first tentative steps in assembly language by New Year's Day 1989.

By this time, I'd already done a fair bit of BASIC programming, and was ready to go further. Using the course and some old tutorials from Computing with the Amstrad magazine, I started creating sprite routines, and began to build a game.

That game was called The Further Adventures of Fred. The first game I'd written when I got my Amstrad in 1984 was a maths quiz hosted by the smiley face from the character set, which I'd named Fred. In Further Adventures, I'd created a multicoloured version of the character and put him in the kind of surreal and abstract settings that were a normal part of 80s gaming. The gameplay was simple: collect tokens, dodge baddies, no gravity. Looking back today, I can see the similarities with Mutant Monty, a game I'd enjoyed a few years earlier.

The Further Adventures of Fred was due to appear on an Amstrad Action cover tape, but the magazine was axed before it could be published.





Bits of the game project were published, though: the sprite code became Easi-Sprite Driver which appeared in AA as a type-in, making it easy to add sprites to BASIC games. The Sprite Definer I wrote appeared on an AA cover tape (June 1992). I think my first appearance in AA was when I sent in some tweaks to the assembler in the Dr Watson course.

Remastering Fred

There were several things that bothered me about The Further Adventures of Fred:

- The difficulty was too hard. The first level had a monster hurtling straight towards you, so you can die almost instantly. My reasoning was that the stakes are low at the start of the game and so it could start with a bit of drama. Now, I think it's just annoying. Creating games in a vacuum, as many of us did pre-internet, made it hard to get the difficulty right.

- There was a practical joke at the end of Level 1 where the game pretended to crash and showed the Amstrad start-up screen. It seemed funny at the time, but again, it's just annoying.
- The default keys were set to joystick, which means nearly everyone today needs to redefine the keys before they can play, especially on an emulator.

So, in 2020, I decided to fix the game. The question was: could I do it, thirty years on, and without much documentation to go on?

Understanding the code

At one point, I had the source code for the game running from ceiling to floor in my room, printed on continuous printer paper. I'd stare at the code for ages trying to find bugs in it. When I came back to the game in 2020, I did find a printout that I could tell had been stuck to the ceiling, but it looked like quite an early version. I couldn't remember enough about Z80 assembly language to make sense of it, either.

Going through my old development discs, I found a BASIC program I wrote that compiled the game from separate files. This was a big help, because it revealed the memory addresses of the sprites, game code, text, and levels data. If you're interested in tinkering with the code, I've [documented these on my website](#).



Decoding the levels

With no real documentation to go on, I was in the strange position of "hacking" my own game to try to work out how the data was structured. For some reason, I'd never really tried hacking anyone else's games to find pokes, so this was new territory for me.

It was all about finding patterns. I began by looking at the levels data starting at memory address 10089 and looking for numbers in the ASCII range (for the name of the level) and for repeating patterns that might represent the walls and the tokens you have to collect. Level 1 (in the original game) only had bricks, tokens, and monsters, so it wasn't too hard to spot the pattern.

```
run
-----
LEVEL: 1
MEM:      10089
The Starting Gun. 0
05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05
05 .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## .. 05
05 .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## .. 05
05 .. ## .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## .. 05
05 .. ## .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## .. 05
05 .. ## .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## .. 05
05 .. ## .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## .. 05
05 .. ## .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## .. 05
05 .. ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## ## 05 05
05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05 05

MEM:      10310

 14 5 14 5 2 4 1 2 3 4 4 1 1 3 4 3 3 3 4 5 5 3 4 18 7
 1 0 0 0 0 0 0 67
-----
```

I wrote a program to display the levels data (shown above). It showed the level's name as text, and laid out the level map as it would appear on screen, but using the numbers in the memory locations instead of sprites. My program also showed the rest of the levels data as numbers. It took some puzzling to work out what this data represented. Some of it was colour data, some of it governed the enemy movements and locations. Most of it I didn't need for my game updates, luckily.

Adjusting the game difficulty

That first screen was really letting the game down. I know it's not the greatest game ever, but it seemed a shame that a lot of people would be put off before seeing much of it, because level 1 was too hard.

Now that I'd decoded the levels data, I was able to change some of the data to make the game easier.

I placed a token between the player and the enemy that charges towards you on the first level (shown below). Enemies can't walk through tokens. This edit stopped you dying immediately if you didn't move fast enough when the level began. I moved a token rather than adding one to avoid having to check for or fix any side effects of adding a token to the game. I don't know, for example, whether there are checks at any point on how many tokens you have. I doubt it, but it was safer not to raise the question.



I swapped the data for two levels around, so the first level is the old level 4, and the old level 4 is now the old start level. The opening level is now the one called Toot-And-Come-In (that's how you operate an Egyptian doorbell, incidentally). I think that provides a softer start to the game.

To actually move the data around, I was loading the game code into memory and then writing simple BASIC programs to PEEK at what's in the memory, and POKE new values there.

Funny peculiar, not funny ha ha

I was really keen to get rid of the sequence that fakes a machine crash at the end of the first level. It was a relic from a time when the game had little joke sequences between all the levels, but it was the only one left, and it was simply annoying.

Fixing this was harder than fixing the levels data, and required some sleuthing. I started by finding the message that looks like the Amstrad had just been turned on. When I'd found that, I searched for where it was referenced elsewhere in the code. Pressing F7 in the fabulous Winape emulator opens the debugger which has a tool for searching for bytes in memory. The text was stored at &22BF. Memory addresses are stored low byte first, so I actually had to search for "BF 22" to find references to memory address &22BF.





My Z80 knowledge is rusty, and this code probably isn't the cleanest either. There were a few op codes I could remember, though. &C9 was RET, used to return from a CALL instruction. 0 was NOP, representing no instruction.

When I found the address that used the machine reset text, I tried zeroing out some of the instructions there and tried poking RET (&C9) to force a return from that part of the code. None of it worked: it just corrupted all the text display and mode changes. Most likely, I'd found the text display code, and not code specific to this joke.

Next, I searched for "joking" because this routine says "Ha Ha! Only joking!" after a short delay. This uses the double-height text rather than the normal text display routine, so I was happy this would lead me somewhere different.

When I found where that text was stored, I searched for where it was referenced elsewhere. I used a disassembler to find my way back to the

start of that routine, which I could identify by looking for the &C9 that returned from the routine before it.

I zeroed the whole “joking” routine, and poked a &C9 at the end. It worked! I might actually have shouted out “yes!” and clapped my hands.

The pace of the game is much better flowing smoothly from level 1 to level 2, without the distraction of the fake reset in between.

Changing the default keys

The default keys in the game are joystick, which means pretty much everyone nowadays has to start by redefining them. It's friction nobody needs. I wanted to make the default keys something more sensible.

This wasn't too hard to fix. I searched for references to CALL &BB09. This is the firmware routine for reading a keyboard character from the buffer if available. I had a hunch I'd probably used this rather than something more sophisticated.

That code was only referenced in one place. When I studied the code, I found a sequence of five CP and JP Z instructions after it. These





instructions could be used to see whether a specific key was pressed, and jump to a particular part of the game code if so. The CP instruction compares what's in the accumulator with a particular number. The JP Z instruction jumps to an address if the comparison matches. There was one pair of CP/JP Z instructions for each of the gameplay keys (including the quit key).

I took a note of the values in the CP instructions. Then started the game, redefined the keys, and took another look. The values had changed! I tried poking the values I'd redefined into the game code and running it to see if it worked. It did!

I was now able to save the game with the new levels data, new key combination, and missing reset routine.

Finalising the game

These small changes made a big difference to the game. It's now easier for people to start playing, and smoother all around. I took the opportunity to tell players about the cheat mode on the loading screen:

Press SHIFT + 1 to skip to the next level. If anyone's willing to spend some time playing the game, I'm happy for them to see as much as they want.

I also took the opportunity to update my other programs, many of them previously published in AA and ACU. Revisiting them 30 years on (yikes!), I could easily see simple improvements I could make. I added more intuitive key combinations and clearer, more prominent instructions. The games would originally have appeared alongside their instructions in print, but now had a life of their own floating around on the internet.

It was fun revisiting these games. I was surprised at how much I remembered about Amstrad programming, and about these game projects.

You can download my updated Amstrad games disc at <https://www.sean.co.uk/books/amstrad/>. It includes the new remix of The Further Adventures of Fred, my sprite utilities, and several games I wrote for AA and ACU. My website also includes the full text of my Amstrad programming book, which has also been updated.

[Here's a more detailed version of this article, including code.](#)





www.sean.co.uk