R TYPE 9 TECHNICAL SPECIFICATION

CODE/R-9
CREW/1
WEIGHT/31.0t
LENGTH/16.2m
WIDTH/10.8m
HEIGHT/5.1m
MAX SPEED/208km/sec
MAX POWER/9.2E14cr
SOFTWARE/R.C.PAPE
GRAPHICS/"MAK"
HARDWARE/SPECTRUM

IT'S BEHIND YOU:

the making of a computer game

by Bob Pape

THIS PAGE INTENTIONALLY LEFT BLANK

IT'S BEHIND YOU:
the making of a computer game

by Bob Pape

THIS PAGE INTENTIONALLY LEFT BLANK

INTRODUCTION

A friend of mine who knows I used to write computer games a long time ago mentioned in passing that he'd seen one of my old games covered in a magazine along with a request for the author to get in touch. I think he was surprised when I said that I wasn't interested in the idea because there does seem to be a belief that all coders like a bit of recognition and enjoy talking about the old days. I can't disagree with that as I think deep down most of us who write computer games are show-offs at heart - we set aside a piece of our lives to create something that we hope will entertain people and then wait to bask in the glow of positive reviews from the critics and a complimentary nod of the head from players and peers. However, my experience in dealing with magazines has shown me that what you want to say and what gets printed can be two very different things indeed!

I ended our conversation by saying that if I had the time I'd write up what I could remember about that game, a game with an unusual name - *R-Type*. I thought there wouldn't be that much to say, after all it was twenty five years ago and I'd forgotten most of the specifics. Maybe I'd be able to stretch it out to a dozen or so pages by covering the basics and telling a few stories of what happened along the way as well as adding a bit of colour, including some gossip and trying to explain why some things turned out the way they did.

However, I found the more I thought about it the more it all started coming back to me and those few pages started getting longer and longer because now I wanted to put down everything I could while I was still able to remember it all. And it was also starting to get more and more personal. You see, the one thing I absolutely did want to be was honest - about what happened at the time and now looking back at it from a safe distance the way I felt about things and how they affected me. I also wanted to try and dispel the rosy tinted aura that seems to surround the writing of those old games - what a Gosh-Wow-Jolly time it was to be a coder back then - when the reality was often far far different.

So as a consequence of my obsessive attempt to try and cover everything I'm afraid you're going to have to wade through quite a bit of incidental material as well and I give you fair warning that later on you'll read such riveting details as how I managed to stay in clean clothes and what I liked to eat on a Friday night. But I'll try and balance that with the odd tale of threats, fraud, stupidity and chicken nuggets just to keep things interesting.

Bob Pape

THIS PAGE INTENTIONALLY LEFT BLANK

PREFACE

"It's 10:15 in the night - August 9th 1988 - R-Type schedule is due for completion today - still another week+ to go - just getting room to put homing missiles in."

I wish all of what you are about to read could be as accurate as that note I made in the back of my Rodney Zaks *Programming the Z80* book all those years ago and still have - but I'm afraid that's about all that remains on paper to remind me of a few months of my life during the Spring and Summer of 1988. That's not to say what you are about to read is made up in any way but twenty five years later memories have faded and those that remain have been tumble-polished in my head - unconsciously of course - so that they tend to put a shine on my version of events. I'm sure if you asked any of the other participants they would have their own quite different take on things, which of course is how it should be. When I talk about other people's thoughts and actions, why they did or said certain things or behaved in a certain way, obviously it's just my take on things. I certainly can't say why somebody thought something but I can say what I think they thought (does that make sense?) which of course may not be the actual truth of the matter, but it is mine. I don't have the right to tell their stories and maybe some time down the line they'll tell them themselves.

I hope I haven't confused you too much, if I have perhaps these words from a much missed comic genius will help.....

Andre Previn: *"You're not playing the right notes."*
Eric Morecambe: *"I am playing the right notes, but not necessarily in the right order."*

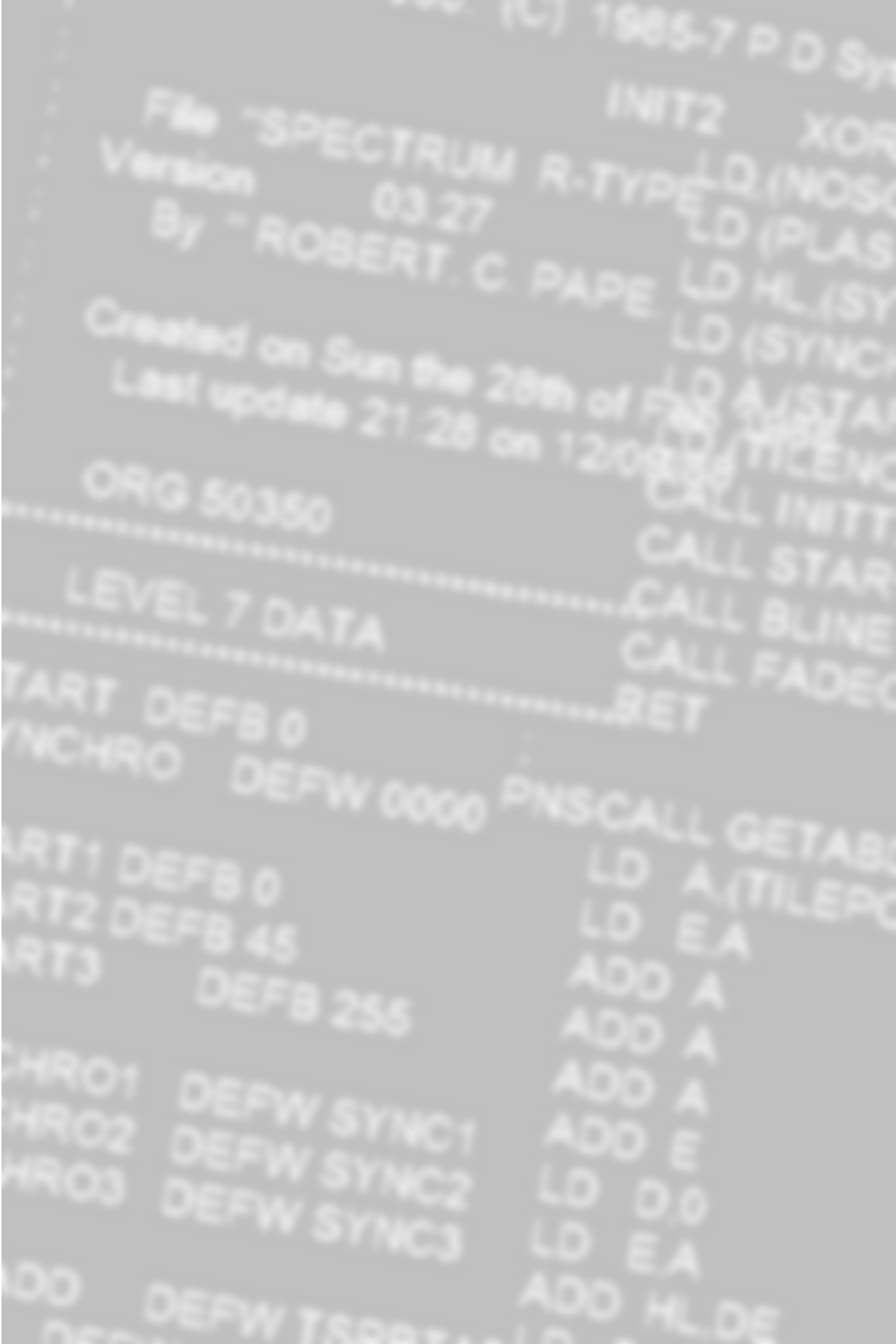
Before getting to the main course I'm afraid you're going to have to sit through the starter, a little bit about my computer background and some of the events leading up to my employment with Designmaker\Catalyst Coders as a games programmer. I'll follow that with a short piece on *Rampage*, my first real computer game, since some things happened during that that were to have an effect on how *R-Type* came to be and which would lead to some unwelcome surprises later on.

I hope that what I write won't be too nerdy or technical for people to understand but I have no wish to go down the road of those mass market books who assume their readers know nothing whatsoever about computers and laboriously explain in simple one word terms - probably at the prompting of a marketing manager afraid of losing sales - just what a 'computer chip' is and how RAM is not the same as ROM. So I'll make the assumption that you know what a 48K Sinclair ZX Spectrum is and that when I talk about the computer

games that were around at the time you'll know what I'm on about - or at least have a rough idea. I'll take a stab at explaining things if I can but forgive me if I don't seem too clear or go into a lot of detail, it was a long time ago and I have a hard time remembering it all now myself.

Lastly, a clarification. Although Spectrum *R-Type* was released under the Electric Dreams label I have always looked on it as an Activision game. Activision at the time were using a number of different names for a variety of reasons - Activision, Electric Dreams, Software Studios, Mediagenic - the distinction of which was lost on me then as it is now. At heart it was the same people in the same building doing the same jobs so I'll be going with Activision and sticking to it throughout this book.

BEGINNINGS


(C) 1985-7 P D Sy
INIT2 XOR
File "SPECTRUM R-TYPE
Version 03.27
By " ROBERT C PAPE
Created on Sun the 28th of
Last update 21:28 on 12/08
ORG 50350
LEVEL 7 DATA
TART DEFB 0
YNCHRO DEFW 0000
RT1 DEFB 0
RT2 DEFB 45
RT3 DEFB 255
HRO1 DEFW SYNC1
HRO2 DEFW SYNC2
HRO3 DEFW SYNC3
DD DEFW TSRB
LD (NOSC
LD (PLAS
LD HL,(SY
LD (SYNC
CALL INITT
CALL STAR
CALL BLINE
CALL FADEC
RET
PNSCALL GETABS
LD A,(TILEPO
LD E,A
ADD A
ADD A
ADD A
ADD E
LD D,0
LD E,A
ADD HL,DE

I left Olchfa Comprehensive School in Swansea at the age of sixteen with four GCSE O Levels to my name. This being 1977 and a time when jobs for school-leavers were readily available I went straight into employment with local toy company Mettoys, creator and manufacturer of metal die-cast toys most notably Corgi Cars and later somewhat incongruously the Dragon Computer. I was a sixteen year old Commercial and Production Management Trainee, a participant in a scheme the company had to take four young school leavers and move them around all the departments in the company, spending time in each grasping the fundamentals of how things worked.

The plan was for us to become the rising stars of management with a complete understanding of all parts of the company under our belt but of course we were the gone-next-week trainees who got all the crap jobs going. One of those places was the company Computer Department where there really were no crap jobs, just the Operators running the computers and the Programmers writing the software. While I would like to say that it was the programming side of things that got my attention I have to say I found it (commercial business programming that is) extremely boring and much preferred the exciting life of a Computer Operator with its all night hours, 43% shift allowance bonus payment and no dress-code whatsoever.

When my two years of being a management trainee were up I threw a spanner in the works by saying I didn't want to be any sort of manager and could I be a Computer Operator instead please? I think the whole thing was flawed anyway since Ian, another one of the four trainees, also chose to be a Computer Operator and that's where the both of us ended up.

I was 18 years old, working as a Computer Operator on an ICL 2960 mainframe (amongst others), and to be honest not a shred of computer nerdiness in my mind - it was a good job, it paid quite well and the hours were great. All that changed when I read a two-page spread in the Daily Mail about how those crazy Americans were now selling computers you could just plug in and use in your own house, an actual 'home' computer!

If you can imagine the words "*home computer*" said with the same quizzical incredulity as comedian Peter Kay's *"garlic bread!!"* you'll have some idea of how stupid it all seemed at the time but no, there were pictures of something called a PET and a TRS-80 that you could buy for around £1000 and do all sorts of things with like....errrr....balance your chequebook, keep track of recipes in the kitchen and write letters with. Unfortunately we'd have to wait for the arrival of the Sinclair ZX81 before we could do really important things like running a nuclear power station with them but the Daily Mail article did mention that you, yes you reading this, could make these 'Home Computers' do things by programming them yourself and that interested me. The fact that somehow, using something called BASIC, you could write your own programs and then see them working got me hooked.

My local library only had one computer book with BASIC in the title,

but thankfully it was the best there was - *Illustrating BASIC (A Simple Programming Language) by Donald G. Alcock. E*ven today I believe it is one of the best computer books for beginners ever written, easily outclassing all the 'For Dummies' and 'Idiot's Guides' by miles. After returning the book to the library I bought a copy for myself, spent a lot of time reading it and amazingly it all started to make sense! I went out and bought my first computer, a Sinclair Cambridge Programmable Calculator with a whopping eighteen bytes of memory and spent many a happy minute playing Lunar Lander on it.

Back then Swansea only had one computer store, a TV repair shop that also doubled as a Tandy franchise with a TRS-80 on display but it somehow also managed to stock Commodore PET 2001s as well. Though I couldn't actually afford to buy either of these dream machines the books and manuals on sale alongside were a lot cheaper and I ended up with quite a mixed library explaining all about PET Jiffys and Tandy block graphics.

But of course reading is never the same as doing and eventually I wound up on a Northern Line tube train heading for end-of-the-line High Barnet and a long walk to a small computer shop on the outskirts of London that was selling dodgy 16K TRS-80s - modified with a computer modulator that let you use any TV instead of the obligatory expensive monitor. So for a couple of hundred pound less than an official TRS-80 and a lot less money than a PET

My original Sinclair calculator with perhaps the most awful battery compartment known to man. It still works…..just!.

I now had a real home computer of my very own.

There was no stopping me, I bought even more books including both of David Ahl's *BASIC Computer Games* books and spent weeks perfecting the now long-lost art of typing in program listings. It wasn't games in particular that I was interested in just that there seemed to be more of them around than anything else in book form ready for typing in -

there was even a book that was nothing but a listing for a single game, *SARGON Chess*. Programs that came ready to run on cassette were another matter, usually having been shipped over from the USA they were expensive and hard to get hold of, and the only 'real' programs I had were some of the Scott Addams text Adventures that had been copied for me (illegally) by the shop where I bought my TRS-80.

I got to like playing text Adventures, maybe because it was all I had, but Addams' Adventures in particular have always had a warped streak to them, the Snakes and the Mongoose puzzle from *Pirate's Adventure* is particularly twisted and still makes me smile just thinking about it. Anyway, it was time to have a go myself, my very first program, a proper program, something that hadn't been seen before. I was going to write.....a Pinball game!

It was a total failure and I gave up halfway through. Funny thing though, I later realized where I'd gone wrong but by then it was too late as I'd chucked everything out. It was the collision detection, I'd totally messed it up by mixing spaces and blank characters which - though they both happen to look the same i.e. empty blocks of nothing - have a subtle difference when you're checking for ASCII codes. The ball movement was pretty awful as well, but then it was all written in Tandy BASIC so it was really doomed anyway, I was just too naive to realize it at the time.

My second attempt at programming went a lot better in that I actually finished it this time. *Garstly Grange* was a simple text adventure which had you exploring a deserted conjurer's house but it was hardly in the same league as the Scott Addams range being largely hard-coded for specific actions. By that I mean it wasn't written around the idea of an Adventure Interpreter such as Infocom's *Zork Implementation Language* and Gilsoft's *The Quill* but was mainly line after line of IF THENs. Again with hindsight you could see the major problem.....if you knew the game it played as smooth as silk but if you came to it cold the (very) limited responses would have you chucking it away in frustration in minutes. More naivety on my part, and I never showed it to anyone other than a few friends, but that experience probably meant more to me than anything ever since because I had finished it and it worked. A program. My program. I was a real Programmer!

The one thing *Garstly Grange* did have in common with the Scott Addams Adventures was a nice split-screen scroll, a type-in effect I got from a magazine somewhere. Of course it was in machine code but I hadn't got a clue what it was doing or even how, I just typed in the numbers and then, through the magic of the TRS-80 VARPTR command, half my screen stood still while the other half continued to scroll. I had bought a few books on machine code during my buy-anything pre-ownership days and even tried to read them but always gave up after just a few pages, it just didn't make any sense to me. It wasn't the syntax or logic behind the commands that I didn't get but rather the whole idea of Machine Code,

Assembly Language, Debuggers, Monitors, PEEKs and POKEs - what did it all mean!?

Perhaps I gave up too easily, or maybe having finished a program I didn't have anything else to prove to myself, but that was all I did on the TRS-80 for the next four or five years. After being made redundant at Mettoys I moved around the country as a Computer Operator taking my TRS-80 with me but never feeling the need to sit down in front of it and get creative. I was buying games and utilities and still using it but what little programming I was doing was via BASIC interpreters on the mainframes at work. I don't know what ICL were putting in their machines but on one 2980 mainframe I had to put a time check into a text adventure I was writing so that it could only be run after 6pm as it would grind the entire system down to a crawl. And that was just a simple BASIC loop!

I did get the TRS-80 out for one last project. I was into the *Traveller* RPG at the time so decided, naturally, that the 16K TRS-80 could easily handle a multi-player Play By Mail computer version of the trading part of the game. I may have been older but that naivety was still there and I got about as far as typing in the data for the planet names and goods descriptions before realizing that it just wasn't going to work - a problem that David Bell and Ian Braben managed to solve with *Elite* quite elegantly later on.

I first came across the FORTH programming language within the pages of computer magazines and there was just something about the whole idea of creating custom commands\keywords that would do exactly what you wanted them to that appealed to me. No more struggling with a limited BASIC Interpreter, if the commands didn't do what you wanted then just write a new one. And what better way to learn this language than with a dedicated FORTH machine, The Jupiter Ace, perhaps one of the daftest ideas ever launched on the early home computing scene. Yes, a home computer with a built in FORTH compiler\decompiler that resulted in very tight, compact programs might have taken the market by storm if it hadn't been for (amongst other things) a case that seemed to be made out of the same material as plastic yoghurt pots and a rubber keyboard that made the 'dead flesh' of the ZX Spectrum's seem warm and inviting by comparison.

I could go on, and will, by mentioning the tiny 1K of RAM the machine possessed, a black and white character based display, the alarming creaking and flexing of the machine when you pressed too hard on a key and the necessity to buy a RAM Pack before you could do anything even remotely interesting with it. But despite all the shortcomings it really was a great way to learn FORTH, well worth the £90 it cost from my local Debenhams and it wasn't long before I was coding some games for the machine. Actually, writing your own games was more of a necessity than anything else as the machine was a dead duck on the High Street so getting hold of ANY software for the machine was an exercise in futility.

Still, typing away on my bed at night after work I managed to come

up with four 1K games, one of which was a micro text adventure that stored all the data on-screen but hid it by making the screen and text the same colour and restricting play to a few lines at the bottom of the screen! Though the original plan was to try and make some money off these games nothing ever came of it, they didn't get sent out to anyone to look at and are now long gone, but I can see that learning FORTH was the spur I needed to make the change from BASIC to machine code as the next time I looked over a Z80 Assembly Language book things actually started to make a little sense.

I got into the ZX Spectrum quite late, sometime around 1984\5 and originally bought a 48K machine just to play games on, and one in particular, *Manic Miner*. I really had no intention at all of programming the machine as I was now working as a freelance mainframe operator and some-time BBC Micro developer, but working as a freelancer meant stretches when there was no work so I tended to play games on the Spectrum out of boredom more than anything else. The first time I attempted to code something for the Spectrum was when a magazine published an article about how you could use ROM routines instead of BASIC Commands to do things like clear the screen and make a single pixel appear on screen really quickly instead of using the BASIC PLOT Command and I thought hmmm..... if I could do this 64 times then I could make an 8x8 pixel graphic appear anywhere on the screen I liked.

So I duly made up an 8x8 grid, worked out which pixels needed to be set where to make it look like a ball and called the ROM PLOT routine 64 times via some simple loops all in machine code. That was about the limit of my knowledge, no Assembler to do all the hard work just working out what numbers I needed to POKE and how to set it all running. After a few failures I eventually managed to get my 8x8 ball appearing almost instantaneously anywhere on the screen I wanted it to. Of course calling a ROM routine 64 times, even with machine code loops, is (relatively) slow so I got hold of a copy of *The Complete Spectrum ROM Disassembly* book to see what exactly it was I was calling and a simple tape assembler to make things easier to code and bit by bit learnt to optimise the code I was writing. As soon as I realized I could replace eight calls to the ROM routine with a simple LD (HL),n command I knew I was on to something!

While all this was happening I'd bought a copy of Gilsoft's *Quill Adventure System* with visions of writing a text adventure game that I could sell - probably much like everybody else who bought a copy - and I had a few ideas for games but decided to try and learn the system first by creating a version of that old classic, *Colossal Adventure* by Crowther and Woods. I had a BASIC listing of *Colossal Adventure* which had been printed in a US computer magazine so the room descriptions, objects, messages, logic and commands were there in front of me and all I had to do was transfer them to *The Quill* which was pretty easy.

The Quill was a very nice piece of software but the more I used it the more I realized its limitations, especially in regard to something like *Colossal Adventure* with its long location descriptions and constantly having to describe things again and again as the text scrolled off the screen. If only *The Quill* had things like text compression, split screen scrolling and a bigger display area, getting *Colossal Adventure* to fit into the limited RAM wouldn't be a problem. But it didn't, so there was only one alternative and that was to implement them myself.

This is where I really learnt to program in Z80, spending weeks going through *The Quill* courtesy of a *Romantic Robot Multiface 1* and a disassembler program, making notes about where I thought various routines went and what they did and then trying to redirect them to my own blocks of code. Internally *The Quill* was a very neat piece of code which made tracking what routine did what pretty easy and it was also a very safe piece of code, calls to subroutines would be handled cleanly without messing the registers around too much which made slotting in my own custom routines all the simpler. What did surprise me was how easy I found it to write things like a real-time text compression\decompression system and 42 character per line routines and by that I mean once I'd worked out in my head what I wanted to do and how to do it then using Z80 Assembly Code to implement it just seemed the easiest way of doing things.

Eventually I ended up with a single block of code that added over fourty new commands to *The Quill* and implemented text compression, 42CPL output, auto text formatting, RAM Load\Save, Sound routines and the basis of a custom graphics system I was hoping to add later. I also gave it a name….*The Enhance*r.

The Enhancer was the first thing I wrote that I thought had a chance of making money, and who better to market it than Gilsoft themselves so I decided to pay a visit to their stand at one of the old PCW Computer Shows thinking that as soon as they saw it they'd realize what it could do and would be only too eager to take it off my hands! I did get as far as talking to someone about it but they didn't sound that enthusiastic, probably because they already had a product out called *The Patch* which gave *The Quill* split screen scrolling, RAM Save, sound routines and some flag manipulation but also because they were launching another add-on called *The Press* which allowed you to compress your text adventure as well as adding functions found in *The Patch*.

Of course I thought my version wiped the floor with *The Patch/Press* especially since magazine reviews mentioned it could take up to eight hours to compress a full-size text adventure and mine did it instantly and what I should have done is tried to get another company interested in it. But it can be hard to remain optimistic when you're just starting out and that old naivety was still in control so I kind of accepted that no one wanted my code and returned to Swansea totally disheartened.

```
SHUTCLS     0 0 - Change 'normal' CLS into a special 'shutter' type effect
FADECLS     0 0 - Change 'normal' CLS into a special 'fade' type effect
NORMCLS     0 0 - Restore CLS to normal ROM based one
NOCLS       0 0 - Whenever a CLS is encountered no action is taken
CRON        0 0 - Turn Carriage Return after printing messages ON
CROFF       0 0 - Turn Carriage Return after printing messages OFF
RAMSAVE     0 0 - Save Current game position to Memory
RAMLOAD     0 0 - Restore a previously RAMSAVE'd game position
PCONT       0 0 - Start outputting text at the end of the last output
AUTOSCR     0 0 - Set Scroll line to first line containing an asterisk\"*"\Code 42
DOALL       0 0 - Step through all words (1-252) FORCE'ing that action
REPEAT      0 0 - Repeat DOALL until all values accounted for
NOUNOBJ     0 0 - Return Object Number associated with current NOUN
NOUNLOC     0 0 - Return Object Status associated with current NOUN

* PFLAG     n 0 - Print Value in flag n
* PLOC      n 0 - Print Location n as a Message
* PSYS      n 0 - Print System Message n as a Message
* POBJ      n 0 - Print Object Description n as a Message
* PMESS     n 0 - Equivalent to MESSAGE (but can use Flag 11)
* SCROLL    n 0 - Set Scroll Line to line n
* WAIT      n 0 - Pause n Seconds (Break off if a key is pressed)
* RND       n 0 - Generate a Random Number between 1 and n inclusive
* OVER      0/1 0 - Toggle Global OVER on/off
* BRIGHT    0/1 0 - Toggle Global BRIGHT on/off
* FORMAT    0/1 0 - Toggle Automatic Text Formatting on/off

* Indicates that if 255 is used instead of the FIRST figure then the value held
  in Flag 11 will be used instead.

CHAR32      lsb msb - Toggle 32 CPL mode ON /  set SYSVAR CHARS to address
CHAR42      lsb msb - Toggle 42 CPL mode ON / set 42CHARS to address
UDG         lsb msb - Set SYSVAR UDG to address
CALL        lsb msb - Call Machine Code routine at address
PEEK        lsb msb - Return value of Byte at address
POKE        lsb msb - Place Byte in Flag 11 into address
WHITE       lsb msb - Output White Noise (value) number of 'clicks'
SWAPFL      n n1 - SWAP value of flag n with flag n1
COPYFL      n n1 - COPY value in flag n to flag n1
ADDFL       n n1 - ADD value in flag n to flag n1 and store in flag n1
SUBFL       n n1 - SUBTRACT value in flag n from flag n1 and store in n1
PAT         n n1 - Set Print Position to n,n1
FORCE       n n1 - Force the running of VERB n and NOUN n1
NEWSYS      n n1 - Change System Message n to Ordinary Message n1
* CARRY     n n1 - ADD to, SUBTRACT from or SET number of Objects carried
* SOUND1    n n1 - LOAD Sound(1) Register n with value n1 (or PLAY)
* S0UND2    n n1 - LOAD Sound(2) Register n with value n1 (or PLAY)
* MOVE      n n1 - Change Object Number n1 to status n

* Indicates that if 255 is used instead of the SECOND figure then the value held
  in Flag 11 will be used instead.
```

The Enhancer even came with documentation, though not as neatly laid out as this recently retyped list of extra Commands and sample instruction page overleaf.

As well as the previous Commands, the ENHANCER also contains a number of Special Features. You have already encountered Split Screen Scrolling and 42 Characters Per Line Output but also available are Automatic Text Formatting in both 32 and 42 CPL mode and a unique system of Text Compression. As you may have guessed, Text formatting is automatic and means that no matter what is printed out it will never be split in half when you come to the end of one line and the beginning of another, So when entering in a (say) Location description you do not have to worry about an untidy display as everything will be taken care of for you automatically.

The Text Compression system remains dormant until you activate it. Any part of the Quill may be compressed (except the Vocabulary and its attendant Options) and will result in a saving of anywhere between 30 & 50 percent. To use the Compressor all you have to do is enter a CODE 127 (© - the Copyright Symbol) at the start of the text and press ENTER. Everything AFTER the Symbol will be compressed and the Symbol itself removed. If you now ***PRINT*** *or* ***LIST*** *the text you will see it, apparently, unchanged, But it has been Compressed! An important point about the Text Compressor is that certain characters will not be accepted by the Compressor (UDG's and some obscure Punctuation) and if the Compressor comes across these characters it will convert them to a SPACE, but the majority of Characters (including Colour and Tab Control) are included and will be retained intact.*

NOTE You cannot EDIT or AMEND compressed text because that would result in the Compressor trying to compress already compressed text! It you find that you have accidentally tried to edit such a text DO NOT PRESS ENTER. To get out of the edit and keep the compressed text unchanged use CAPS SHIFT 6 to abandon the edit.

You may recall that when ***DOALL*** *and* ***REPEAT*** *were explained it was noted that Vocabulary Word Value 254 was used for something else. This is the final feature of The ENHANCER and allows the Adventure Player to use the Pronoun 'IT' during the game, Using Option A on The Quill, insert the word IT as Word No. 254 in the Vocabulary. Now whenever the Player enters a Command with IT as the Noun the program will 'remember' the last Noun entered and use that as the valid Noun.*

e.g. The player types in GET BOOK and the Database responds 'YOU HAVE THE BOOK', If the player now types in READ IT the database will interpret that as READ BOOK and carry out any Conditions and Actions entered under READ BOOK.

When the Adventure has been completed and saved to tape via the ***SAVE ADVENTURE*** *option of The Quill it is also necessary to save a 'run-time' copy of the ENHANCER as well. This is where the second program on the tape comes in as this will allow you to save off a shortened version of The ENHANCER and hence allow you to run the Adventure as a stand alone program.*

On loading the program you will be asked where you want the ENHANCER to be in memory, you should pay particular attention to the size of your completed Adventure (via option O on The Quill) and ideally the ENHANCER should be placed somewhere just above the top of your program, If you intend to use the ILLUSTRATOR from Gilsoft then you should be careful to place the code somewhere in between the two Databases. Remember, The Quill builds UP into memory, the Illustrator builds DOWN and the size of the run-time ENHANCER code is 2900 bytes.

To complete the system, a BASIC loader should be created to load all the necessary parts and a RANDOMIZE USR nnnn used where nnnn is the address at where you instructed The ENHANCER to reside in memory. Do not use the BASIC loader created by the Quill/Illustrator as this will not include instructions for loading The ENHANCER.

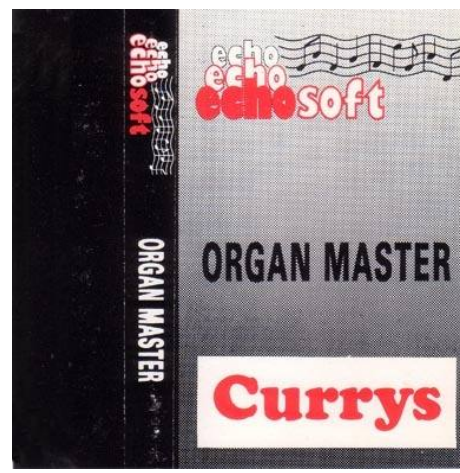
Note the throwaway line "a BASIC loader should be created to load all the necessary parts," which assumed a level of programming competence sorely lacking today.

While I was learning the ins and outs of Spectrum machine code I still had to make a living so I agreed to convert an integrated business package that a previous employer of mine was selling for the BBC Micro Computer to run on CPM machines, specifically the old Apricot computer. As a sideline Val, my ex-boss, had got hold of ten thousand piano-type keyboards which he was trying to sell to Spectrum 128, BBC and C64 users under the name *Echo Musical Synthesizer* along with the *Echosound Amplifier* - a small five Watt sound amplifier - so you could actually hear what you were playing, but they weren't selling that well. The fact that the ten thousand keyboards were in pieces and had to be put together one at a time didn't help nor did the heavy metal case and large keyboard size, which meant mail order postage costs were taking a large chunk out of any profits. In all the time he'd been selling them Val had shifted maybe eighty units so the chances of getting rid of the other 9,920 seemed pretty slim and it looked like he was going to have to dump the lot.

However, by some miracle, Val managed to talk Currys Electrical into taking the whole lot off his hands for a nationwide 1986 Christmas promotion of the Spectrum 128K - provided he could deliver all the units assembled and ready to go in a ludicrously short number of weeks that was.

An old garage outside the office was set up as a production line as staff and family members set to building almost ten thousand keyboards. A two-man production company were contracted to produce the interface boards and nearby Kempston Micro Electronics Ltd. supplied plastic casings for these which were actually old Joystick Interface boxes with the name scraped off and some of the holes widened (by hand, one at a time, all ten thousand) to take the slightly larger connectors. The software was a problem though, Currys were only interested in a Spectrum version and they wanted users of the 128K machine to be able to record and playback music, something the existing software didn't do.

Val contacted the original software developers for the keyboards but was quoted such a long development time and high price for the software that he decided to go elsewhere, and since I was the only person he knew who could program a Spectrum I ended up with the job. I can't remember if I was told I was doing it or I volunteered - probably the latter in a show of bravado boasting - but I found myself running around London's West End one afternoon trying to get hold of a copy of the *Laser Genius* assembler from Ocean



Cassette Cover of Organ Master - definitely no expense was spent!

Software that was supposed to specifically run on a 128K Spectrum (it didn't) and any books on machine code for the 128K (none whatsoever.)

It was a total hack job as I shoe-horned in a subroutine to make a note of what key on the keyboard had been pressed and for how long and then saved it off in a stupidly inefficient way. If I had any proper coding experience at this point then I'd have approached it differently and rewritten everything but for Val time was money so in just under two days I amended the mainly BASIC program, added a machine code routine to access the extra memory of the 128K machine and implemented my record\playback bodge.

Val was happy, calling up the original software developers and having me play back a simple tune while he held the phone up for them to listen to it and loudly told them what he thought of their original demands, but I knew I could have done better given more time. After that it was all hands to the pumps as we tried to get those ten thousand keyboards built and delivered on time and I spent most of those weeks delivering components from a warehouse to the production line twice a day and taking the finished units back to the warehouse for paletting and pick up in the evenings. We just made it and Currys were happy with everything, and so was Val for getting rid of what promised to be an expensive pile of unmarketable components but I was in two minds over my contribution.

My first commercial piece of software was now on sale (being given away if you want to be accurate) but all it was really was a

Loading screen of the Organ Master program - created by myself in a rare burst of graphical ability.

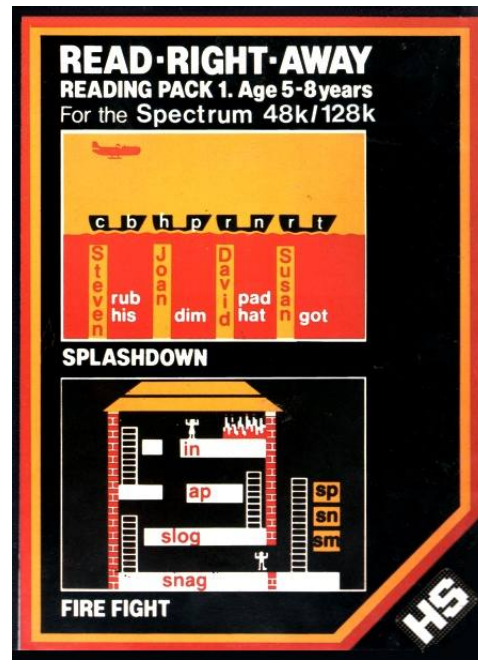
simple hack of someone else's code, and a poor one at that, not something I was too proud of. I finished off the BBC to Apricot conversion and returned home, now jobless and ready to pick up my Spectrum programming where I had left off.

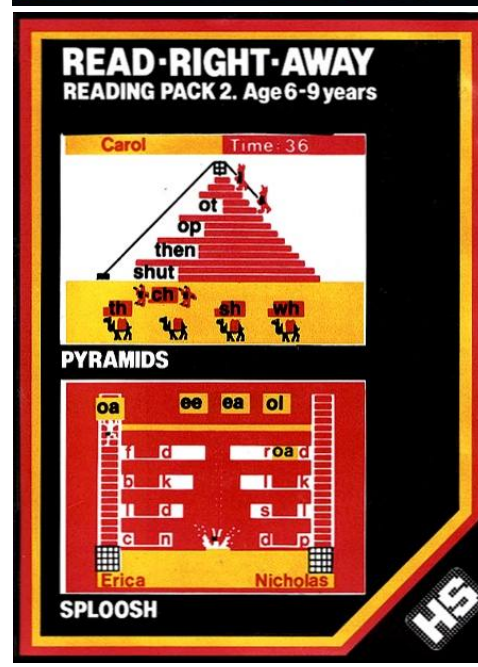
If no one was interested in an adventure editor utility then I thought I'd give the people what they wanted, and what they wanted was.....another *Manic Miner* clone! Yes, I can see the total futility of that now but back then it sounded quite a good idea, and I use the word clone deliberately because I decided to try and duplicate *Manic Miner* as closely as possible using the logic that since it had already been done all I had to do was work out how and I could do it too! So with Miner Willy graphics ripped from the original I soon had my(!) character smoothly walking and jumping around the screen, falling through collapsible blocks, being moved along conveyor belts and anything else I could imitate - all thanks to some sprite routines I had written and the magic of Z80 machine code.

Unfortunately I soon hit a major stumbling block in my plans, I couldn't draw, and what graphics I did produce were simple to say the least....one screen was a billiard table as seen from top down (but played as if it were a vertical surface) because I could just about manage to draw coloured balls, but anything else was totally beyond my graphic ability.

While I was trying to achieve fortune and glory with a small jumping man I still had to make a living and I stumbled across my next opportunity by accident. Reading through a computer magazine I noticed that a company producing educational software for the BBC Micro was based in Swansea, more than that the address was just a short distance away from where I lived so I got in touch with them. I wasn't looking for work, just the opportunity to talk to people about home computers and software without having to explain every single thing first, but my luck was in as they wanted to release Spectrum versions of their BBC programs and would I be interested in doing the conversion?

Of course this being the age of the bedroom coder H.S. Software turned out to be just that, one man working literally from his bedroom at home, but it was being professionally done with proper duplicating, printing and packaging and the two of us got on really well. I agreed to convert a couple of his BBC games, *Fire Fight* and *Splashdown* and put together a set of machine code routines that I'd be using for them and hopefully future games. The routines were simple,





They may not have set the home computer world alight but it was a good learning experience for me.

generic pieces of code: scroll a box X-Y in size up, down left, right, move a sprite around the screen, print

double-sized characters and some sound routines based on the ones in *The Enhancer*.

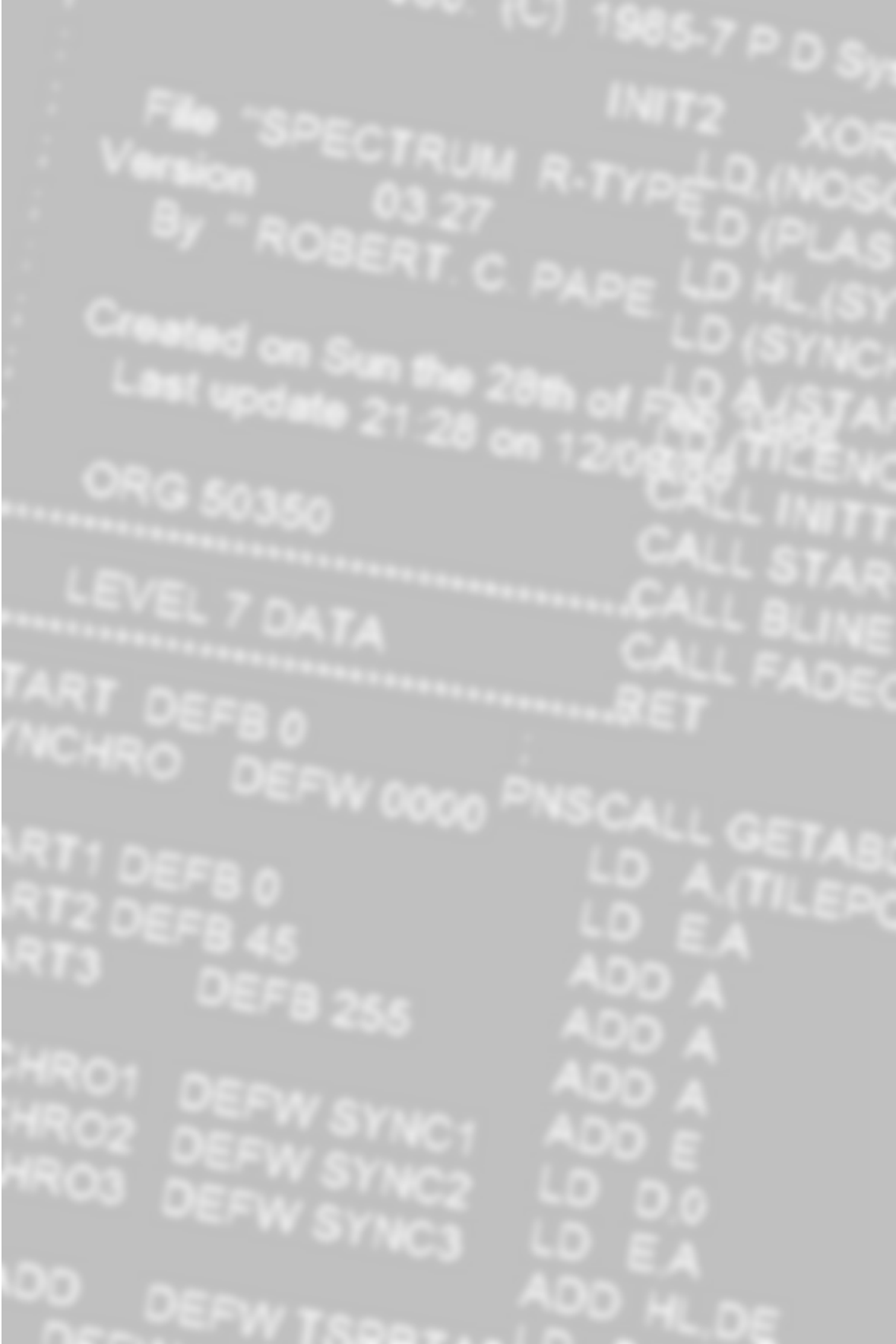
The games were a mix of BASIC and machine code with graphics based on the BBC originals, hardly on a par with the likes of Ultimate or Imagine with their simplistic approach and game play but then they were aimed at teaching five to eight year olds how to read and spell so hopefully that can be forgiven. The games garnered a review in the September 87 edition of Crash magazine where they were noted as being "*very nicely presented and fun to play"* while the graphics were "*very eye-catching and attractive."* I did another two games for H.S. Software along the same lines, *Pyramids* and *Sploosh*, and then we agreed to wait a while to see how they would sell before deciding whether to convert the rest of the games in the range or not.

I don't know what I'd have done next if - while popping into the local Employment Exchange - I hadn't seen an advert for COMPUTER GAME PROGRAMMERS and the promise I could earn up to £5000 a year writing games, which sounded a bit of alright to my innocent wide-eyed ears. Speaking on the phone to somebody in Southampton from a company called Designmaker I was asked to put together a demo tape of my work and send it off to them to have a look at, so I took what I'd written for my *Manic Miner* ripoff and created a single-screen of a house with three or four floors (even I could manage drawing brick and tile characters) and had about a dozen Miner Willys walking back and forth at different speeds moving in front of and behind various obstacles. I put together some of the screen scrolling and print routines from the code I used for the H.S. games, included a copy of *The Enhancer* and the documentation for it as well and posted it all off.

I was thrilled when I got a call saying that they liked what they saw and would I be interested in working for them? Months later, when I saw for myself how they dealt with demo tapes I have to say that I think I got the job just because what I'd written was in machine code and I could spell my name correctly! 'Reviewing' a demo tape I found out was as simple as someone in the office loading it up and if no one laughed at how crap or inept it was they were in. I freely admit to being one of those very same laughing 'reviewers' later on when I had the chance to look at some of the submissions Designmaker were sent, but then you would not believe the kind of rubbish that was being submitted in hope!

I knew none of this at the time however so of course felt my talents had finally been noticed (ha!) and was looking forward to visiting Designmaker's offices in Neath, a small town next to Swansea. Strangely though I had never heard of or come across the fact that a major computer game software house was situated so close to me before. That I suppose should have been the first hint at the pain and problems that were to shortly come my way.

RAMPAGE


(C) 1985-7 P D Sy
INIT2 XOR
File "SPECTRUM R-TYPE
Version 03.27
By " ROBERT C PAPE
Created on Sun the 28th of
Last update 21:28 on 12/08
ORG 50350
LEVEL 7 DATA
TART DEFB 0
YNCHRO DEFW 0000
RT1 DEFB 0
RT2 DEFB 45
RT3 DEFB 255
HRO1 DEFW SYNC1
HRO2 DEFW SYNC2
HRO3 DEFW SYNC3
DD DEFW TSRB
LD (NOSC
LD (PLAS
LD HL,(SY
LD (SYNC
CALL INITT
CALL STAR
CALL BLINE
CALL FADEC
RET
PNSCALL GETABS
LD A,(TILEPO
LD E,A
ADD A
ADD A
ADD A
ADD E
LD D,0
LD E,A
ADD HL,DE

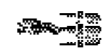
Designmaker's offices were above a tatty old carpet shop on the edge of Neath town centre.....and as I was soon to find out the place was a toilet. Along a corridor from us was a small taxi firm who did most of their business late at night ferrying drunks home. If I was lucky when I arrived in the morning and climbed the stairs I'd be greeted by the remains of kebabs and empty beer bottles, if I was unlucky it would be vomit and full beer bottles - only it wasn't beer that was in them any more. As I said, the place was a toilet - literally.

On my first day there I met up with a Commodore 64 programmer called Andrew Parton and two other coders and we were all introduced to a large slob-of-a-guy named Richard Kightley (imagine Johnny Vegas on a bad day) in a big empty room...that was 'the office' and Kightley the office manager. Kightley was using another room upstairs as his own that doubled as a storage room for his belongings and his bedroom, but that didn't last long as the taxi firm's night time clientele would turn out to be too much even for him.

It was on that first day I learnt I'd be coding the Spectrum version of *Rampage* for Activision which, to be honest, was a bit of a shock as I thought a first-timer like me would be put on smaller tasks rather than a major release for a big company. Andrew probably felt the same when he was assigned the C64 version of *Flying Shark* for Firebird and one of the other guys was given the Spectrum version of *Flying Shark* to code and the fourth walked out after just a few minutes never to return - either he lost his nerve or he had the most sense of the lot of us!

After about a week we had some tables and chairs to sit on and some computers to program with though nothing really in the way of proper development systems, but we didn't know that at the time so just dived in with what we had. We lost the Spectrum *Flying Shark* guy about a month later as he just wasn't up to it and the whole conversion got passed along to coders Dominic Robinson and John Cumming at Graftgold Ltd. who months later produced a hugely popular and skilfully coded version of the arcade original.

The loss of the Spectrum coder meant that Kightley, Andrew and myself were now the total workforce of Designmaker Ltd. but at least our 'office' wasn't so bare - one of Kightley's suggestions was to cut out full page ads from games magazines and stick them up on the walls. I think the idea was to show prospective customers and visitors how professional and cutting edge we were but all it did was make us look a real Mickey Mouse outfit!



When converting an arcade game to another system it helps to have the actual arcade machine there in front of you, something you think would be totally obvious to most people but unfortunately Designmaker couldn't stretch to getting hold of a proper *Rampage* arcade machine which made mapping out the game a little difficult. So in keeping with the Designmaker ethos of 'make do' Andrew and I met up at an amusement arcade in Swansea one evening with Kightley. The three of us

played on a *Rampage* machine there for an hour or so, shovelling in coins for as long as we could. Designmaker couldn't even afford a video camera to film the screen - and I doubt if we'd be allowed to use one in the arcade - so I came up with a cunning plan.

I had a portable cassette recorder in a backpack with a microphone draped over my shoulder so I wouldn't need to hold it and every time a new screen appeared I'd try and describe how many buildings there were, any special features that appeared, what the between-game newspaper headlines said etc. and hopefully try and transcribe what I'd said later on. It really was as low tech as that but luckily the game was quite repetitive in nature so I just about got away with it. Later on I received a video tape of someone playing *Rampage* which helped flesh things out and which probably came from Catalyst..... speaking of which.

It wasn't long before Andrew and I realised that Designmaker was really just an offshoot of another company, Catalyst Coders, a software house based in Southsea run by ex-coder David Wainwright, and that we were right in the middle of some fancy footwork. In the mid 80's the Welsh Development\Enterprise Agency were trying to attract new companies to some of the more run down areas of Wales. The town of Neath, due to the collapse of the neighbouring coal industry, was one of those. Businesses that set up in these areas could look forward to subsidised payrolls and help with start up costs or other bonuses in an attempt to kick -start the local economy but really it was just an excuse for cheap labour. I can't say outright that Wainwright was trying to cut his development costs but it certainly seemed that way to us, more so when we later on we found out that he had done the same in Glasgow through the Scottish Development\Enterprise Agency and set up a similar offshoot company there.

What Andrew and I couldn't get our heads around was why someone like Kightley was running things in Neath since the man had no understanding of games or the games industry at all and came across as a bit clueless about everything connected with running a business. But considering what happened later on perhaps that was the whole idea.

Catalyst were advertising for coders in several magazines but I never saw a PC, Nintendo or Arcade game from them. They also managed to get their address wrong, it was Albermarle not Albermark House.

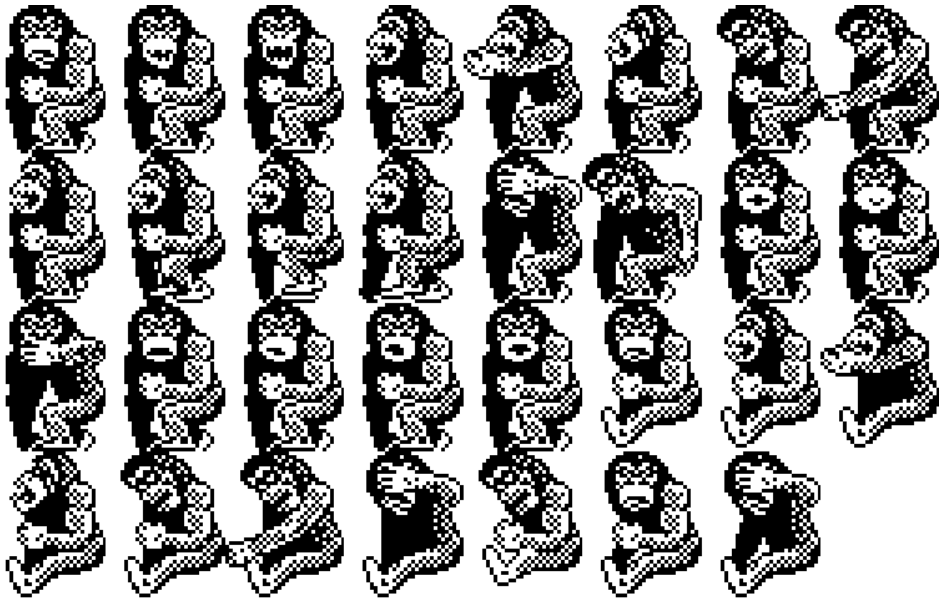
The *Rampage* arcade game was produced by Bally Midway in 1986 and immediately attracted attention due to it's cartoon-style graphics and unique multi-player arcade cabinet. Up to three players control the monsters George (Gorilla), Lizzy (Lizard) and Ralph (Werewolf) as they climb and jump their way around an American city trying to destroy the buildings on screen while avoiding attacks from tanks, helicopters, armed soldiers, jet planes, grenades and even the opposing players - contact with which slowly decrease a monster's Power. Various items appear that the monsters can eat, some of which increase the monster's Power while others drain it. When all the buildings have been reduced to rubble by punching or jumping on them (or through the use of too much explosives) the monsters are transported to a new city, with a different arrangement of buildings, and the destruction continues.

Original arcade version

When two or three people play *Rampage* it's a lot of fun, especially when you realize that not only can monsters attack buildings and defenders but they can fight amongst each other as well. The constant non-stop pace of the game coupled with many small clever touches means there's always something new to be discovered, amongst them - punching a trolley car and seeing it go shooting back off screen - watching a monster turn back into a small human and quickly sidestepping off screen, covering its nakedness with its hands while the other players try and eat him - seeing the look of surprise on a monster's face as it realizes that the building it's on is collapsing and it's going to fall.

Unfortunately the game does suffer when only one person plays, instead of sharing the attacks and damage everything is aimed at that player and there's no real time to explore the game - staying alive become the number one priority. What only became apparent much later on was just how intricate parts of the game are. Something as simple as reaching into a building and grabbing a bite to eat has a multitude of possibilities....a waving human means extra points, if they are holding a camera more points, if the camera flashes it dazzles you and you fall off, an electric toaster can be good or bad depending it it's switched on or not, a toilet can poison you, and so on. Further complications arise when you realize that George can only hold women, Lizzy men, and Ralph businessmen. Very little of this made it into the Spectrum version (or any other home computer version for that matter) but no one complained, players seldom came across these details in the arcade so really didn't know what was supposed to be there and what wasn't.

ZX Spectrum version



Original George the Gorilla Spectrum sprite graphics from Rampage

Being paid had never been a problem during this time, every month we got a cheque and a payslip but since everything had to be sent up from Southsea it usually took a few days, so to speed things up we opened accounts with The Halifax Building Society and money went straight into that instead, with payslips following in the post later. I was being paid about £400 a month for *Rampage* before tax and National Insurance which I assumed was the going rate for games programmers although stories in the magazines about coders earning enough to buy a Ferrari or a Porsche did get me wondering at times. It wasn't as if I could call up somebody and ask them what I should be getting paid, and as a first-timer I did rather take the approach that I should be grateful for actually getting paid to write a game and not worry too much about the amount. So when the penny finally did drop months later I had to accept some of the blame for not asking the right questions at the time. Mea Culpa.

Since I didn't drive, getting to and from Neath in the beginning was a bit of a pain, a half hour bus ride into Swansea City centre and then a train to Neath. But things got a bit easier later on as my sister started working in Neath town centre and would give me a lift home at night. The situation in the office however was deteriorating fast as we were now openly feuding with the adjacent taxi firm thanks to either some dodgy aerial\transmitters on their part or rubbish computer shielding from Mr. Sinclair and Mr. Sugar on ours. Every time we switched on our computers the taxi firm's reception would vanish in a storm of static interference. Of course they blamed us while we protested that everything was properly shielded and RF legal but whoever

was to blame the end result was that interference on their radio transmitters was starting to cost them in lost fares and they didn't like it. This they demonstrated by kicking the office door when they passed, banging on the walls, and playing U2 albums over and over again at full volume.

It was all starting to get a bit worrying as we were convinced that we'd turn up one morning to find the office broken into and stripped bare of equipment so Andrew decided he'd be better off at home and stopped coming in to the office. I would probably have done the same there and then but Kightley had found alternative rooms in an older, better building nearby so I packed up all my stuff and gave that a try for a while.

Shortly after the move to the new location Wainwright invited himself and about a dozen Catalyst programmers to stay for a few days and the whole lot of them were camped in a large empty room there in sleeping bags, which would have been OK but the landlord did an inspection right in the middle of all this and instead of finding a high-tech computer company hard at work he discovered a dozen scruffy coders dossing on the floor instead and promptly gave Kightley his marching orders! I came in to work on the game for a few more days but was totally on my own, no sign of Kightley, and it just wasn't worth spending time and money travelling to Neath any more - so without telling Kightley I packed up all my kit and started working from home as well.

No one seemed to care that I wasn't working in Neath any more, least of all Kightley. I was posting off demos directly to Catalyst in Southsea now and hearing nothing from him, in the meantime Neath was being slowly forgotten. I wasn't getting any kind of feedback about the demos or the game from anybody so I just kept on doing what I'd been doing before and the game slowly got written. But as the game was nearing completion I got a call from Wainwright one Wednesday afternoon asking me how I was doing, how was the game going, was it progressing and.....ummm.....that Activision wanted me in Southampton to work in -house there NOW! After being convinced that he was serious and that Activision were expecting the game to be finished by Friday (which was news to me) I packed my Spectrum kit, a sleeping bag and a few clothes and an hour later was on the way to the train station.

It was a six hour train journey from Swansea to Southampton and I didn't get there until around 11pm. The Activision offices took a bit of finding but eventually I buzzed the main door and got let into the building. Walking into a real software house for the first time was quite an experience, even at midnight the place was full of brightly lit screens, coders typing away, and a constant background of game music and sound effects. I did feel a little concerned that I'd soon be found out, that people would realise I wasn't a proper game coder and just who the hell did I think I was? But no one seemed to have spotted I was still a game-virgin so I spoke to Wainwright, dumped my stuff on an empty desk, set up my kit with a

borrowed monitor and started from where I'd left off that afternoon.

I worked straight through from 11pm Wednesday to 6pm Friday without any sleep, finishing the game off so Activision could ship it to be mastered over the weekend. I have little recollection of what happened during that period - even at the time it was just a hazy blur - so over twenty five years later all that remains are strange little memory fragments....how warm and quiet the toilets were; the theme from the TV show *Knightmare* being played over and over as coder Mev Dinc worked on the Spectrum conversion next to me; eating strange pea rissoles at a small chip shop; spending ages tracing a bug through my source code only to end up back where I started; the feeling of being wrapped in a cocoon of cotton wool, tired bodies in sleeping bags on the floor and a terrible ache in the pit of my stomach that I really didn't know what I was doing. But two things that I do remember in vivid detail are the sound and Artificial Intelligence routines for the Monsters.

As of lunchtime Friday there were still no sound routines in the game and I was starting to panic because this was one area I really knew nothing about and Activision were starting to notice. So I had a word with Mev Dinc and he pointed out that most of the sound FX in the games were thumps and explosions and collapsing buildings and really all that was needed for that was white noise. Luckily I had brought the source code of *The Enhancer* with me which contained a simple white noise generator and two basic BEEPy sound routines that I'd written for it so

I managed to grab one of these six foot long advertising posters from an Activision booth when Rampage went on display. The company were giving away some cool black satin-look jackets with the logo on as well but even though I coded the game I didn't rate highly enough to get one!

I pulled them out of that and shoved them into *Rampage*. There wasn't enough time to finesse anything, no interrupt driven sound, so I came up with a few different short-length bursts of white noise and just slotted them in where I thought they'd sound good. Luckily there's no music and just a few other beeps in the game (or at least in my version!) and I think it just about works, but if I'd known the finish date was coming up so quickly I would have put a bit more work into that side of things.

The monster AI is something that even today I find a bit frightening, and was one of the last routines to be written. I had a byte for each Monster that showed if it was under player or computer control and sometime that Friday morning at Southampton, even though I hadn't written the AI routines yet, I set the byte to computer control just to see if the rest of my code would recognize it. Not only did it recognize it but the monsters started moving around the screen, climbing buildings and punching holes in them! I couldn't believe how this was possible so I stopped everything and checked the code and there, right in the middle of things, were some basic AI routines. I must have written the AI code sometime Thursday, but was so stressed and tired that I'd just blanked the whole thing out of my mind and forgotten I'd done it. Again lack of time meant that the finished routines weren't as good as I'd have liked them to be but I didn't have a choice, so after a bit of tweaking I had to go with what I had.

There was one last major bug to fix and then....it was done! I'd finished my first commercial computer game with maybe an hour to spare and gave it to Activision to be sent off for mastering, and if you're wondering where the game testing, quality control, bug reports etc. were during all this then there's a simple answer, there weren't any. Activision's procedure for releasing a game at the time was basically if Dave Cummins (the head tester) could play it through without it crashing or throwing his joystick at the wall in frustration then you were done and off it went.

Another loading screen I created. If the graphics look pretty good it's because I used a Video Digitiser to grab images from the video tape of the game I had been sent. I was so green at the time that I didn't realize that it was somebody else's responsibility to do this.

There is absolutely no feeling in the world to that of finishing a game you've spent a large part of a year writing and realising that there was nothing more to worry about - a strange mix of euphoria, relief, calmness and pride. I would have left Activision there and then and caught a train home but Wainwright wanted me to stay around the office until he got back from somewhere or other a few hours later so I had no choice. Wainwright was at Activision because

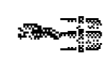
he was finishing off the Commodore 64 version of *Rampage* along with a couple of other Catalyst programmers, Mike Archer and Dave Jolliff, both of whom I'd met earlier when they came and slept on the office floor in Neath - and they had a problem with monsters climbing up through the sky and off the top of the screen so the game still had a bit more work to do on it.

I was just relieved to be finished and slumped in a chair as all of Activision's staff left for the weekend leaving just a few die-hard coders behind to carry on their work, which does sound incredibly trusting but as I found out later not that unusual in the industry. The phone rang in an office somewhere so I thought I may as well answer it, not doing much else at the time, and found myself on the line to Activision USA and talking to someone high up there asking where everyone was and was *Rampage* ready to ship yet? I did my best to cover everyone's ass - "*they're not here at the moment...the Spectrum version has gone for mastering...the Commodore 64 version will be finished in a few hours*" - that sort of thing and he seemed happy. Not only was I now a proper games programmer I was learning the art of bullshit as well!

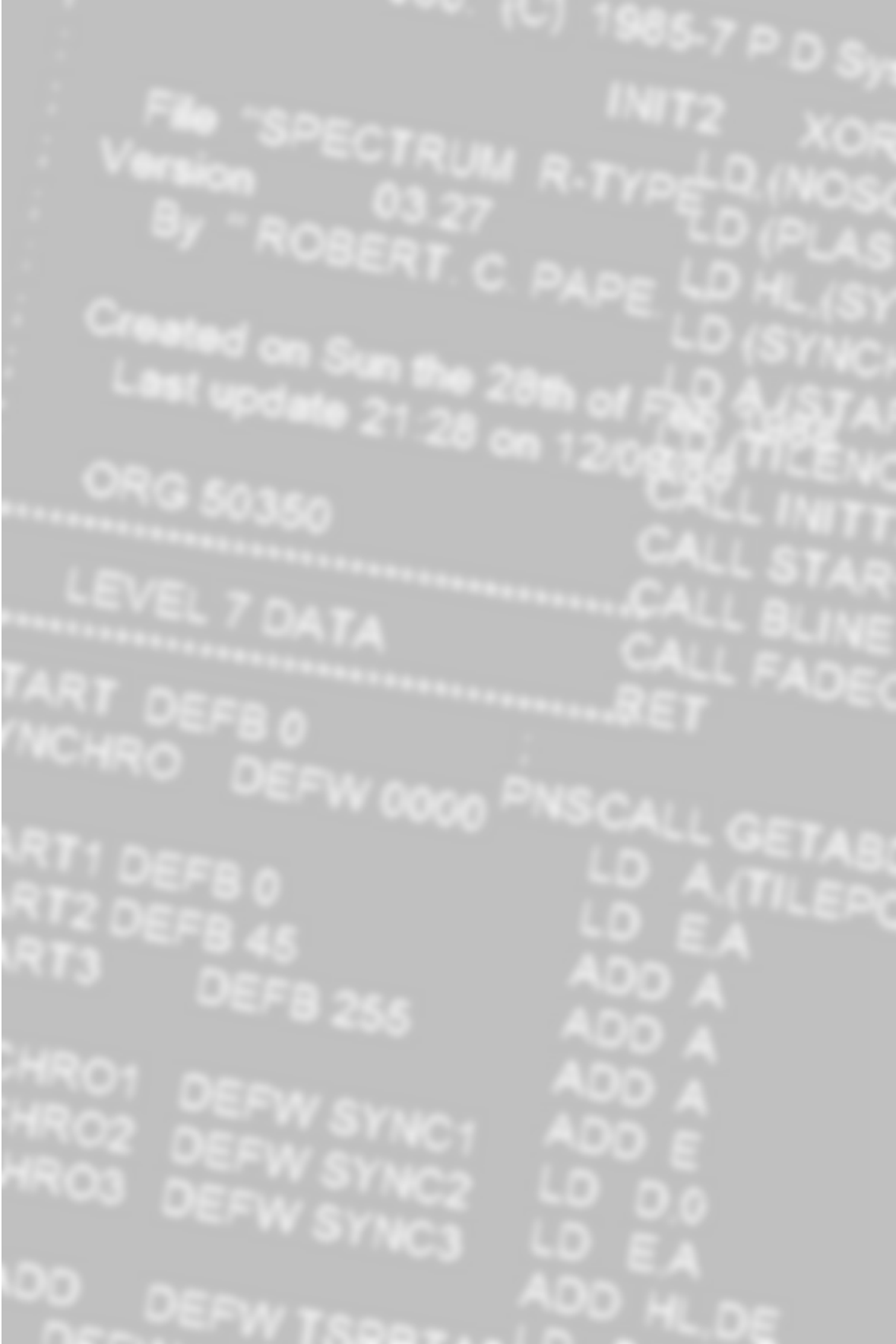
Wainwright eventually turned up leaving it too late to get a train back to Swansea and giving me no alternative but to spend the night in Southampton. I didn't fancy sleeping on the floor at Activision so Wainwright offered me a bed at his place instead, and not having a better option I accepted. I was also really hungry, not having eaten since the night before, so we tried to track down some food but by now it was 3am and all we could find open was an Indian restaurant that could just about manage a steak and chips if we didn't mind it not being that fresh - I didn't and soon polished it off.

Wainwright's house was a little.....messy, from the room that had no floorboards (they were stacked up in a pile in the corner) to the sofa under which was supposed to be the actual video tape of 'Page Three Stunna' Samantha Fox used to prepare the video grabs for her *Strip Poker* game for the Martech label. What was under there however seemed to be several years worth of rubbish and pizza boxes and not even the appeal of seeing Miss Fox in all her naked glory was enough to get me to put my hands in amongst that lot!

I headed for bed, sleeping in my sleeping bag as I didn't fancy the state of the sheets on the spare bed, and woke up ten hours later to find a young lad typing away on a Commodore 64 in the corner of the room, another one of Wainwright's cheap programming discoveries. I got out of there quickly and caught the train back to Swansea looking forward to a well earned rest and a quiet Christmas at home.



R-TYPE


(C) 1985-7 P D Sy
INIT2 XOR
File "SPECTRUM R-TYPE
Version 03.27
By " ROBERT C PAPE
Created on Sun the 28th of
Last update 21:28 on 12/08
ORG 50350
LEVEL 7 DATA
TART DEFB 0
YNCHRO DEFW 0000
RT1 DEFB 0
RT2 DEFB 45
RT3 DEFB 255
HRO1 DEFW SYNC1
HRO2 DEFW SYNC2
HRO3 DEFW SYNC3
DD DEFW TSRB
LD (NOSC
LD (PLAS
LD HL,(SY
LD (SYNC
CALL INITT
CALL STAR
CALL BLINE
CALL FADEC
RET
PNSCALL GETABS
LD A,(TILEPO
LD E,A
ADD A
ADD A
ADD A
ADD E
LD D,0
LD E,A
ADD HL,DE

Following the Christmas wind-down it was time to start coding again, or rather earning money, and early January 1988 Kightley was back in touch. Having given up all pretence that Designmaker were nothing to do with Catalyst Coders he announced that Catalyst had a three game arcade-conversion deal with Activision and they'd like me to do the Spectrum version of one them. All I had to go on at the time were the names of those games...a video pinball-game called *Time Scanner*, a scrolling shooter by the name of *R-Type* and here my memory fails me but the third could very well have been the top-down racing game *Championship Sprint*. Since Swansea was hardly on the cutting edge of video game entertainment a day trip to London was arranged for those of us left from the Neath office to go and see these games in action in the arcades there.

A few days later Kightley, Andrew and myself boarded a National Express coach and headed off to the Electrocoin test-site arcade in Tottenham Court Road and a much swankier arcade in Oxford Street - a round trip of 450 miles just to look at a couple of arcade machines! We met up with some people from Catalyst who'd also made the trip and had a short play of the games. Initial impressions were mixed: *R-Type* looked like a total no-hoper on ANY of the home computers of the time, *Time Scanner* looked OK once you'd worked out how to do the smooth ball movements and as for the third - if it was *Championship Sprint* - then that looked a shoe-in for Tony Mack since he'd converted its predecessor *Super Sprint* for Activision the year before. Because of my unsuccessful past attempt at writing a pinball game I knew I wanted *Time Scanner*, I thought I was competent enough to pull it off this time and maybe it would lay an old ghost to rest. As for *R-Type* I remember saying to one of the other coders "*I pity the poor sod who has to convert that*" when I saw it, and that wasn't just my thinking for a Spectrum conversion. I really believed there was just too much there to fit onto a home computer and not just the colour - the amount of sprites on screen, the size of the play area, movement patterns, all of it was just too much. To say that I was disappointed when told I wouldn't be doing *Time Scanner* and had been given *R-Type* instead would be a bit of an understatement.

After my visit to London I received a call from Wainwright asking if I could come to Southsea to talk about the game. Catalyst had the top floor of a small office building near the sea front called Albermarle House which they shared with another computer company, Data Design. The Catalyst offices were a bit of a tip, full of computers and assorted junk and later on I would find myself having to spend the night there. Unfortunately there was no chance of getting any peace or quiet as someone would be in the office coding or playing games no matter what hour of the day or night, so I sneaked into Data Design's offices and got a few hours sleep on a hard concrete floor in front of a gas fire to keep warm with my jumper as a pillow. Ah…..the glamour of being a games programmer.

Electrocoin's sales announcement of R-Type - almost buried in the small print.

When I arrived at Catalyst that first time I found out that Dave Jolliff would be doing the C64 version of *R-Type* and I knew that the two of us would have to have a little chat about money first. For writing *Rampage* I received the magnificent sum of £2500 plus a bonus of £100 for actually finishing it and Dave had got pretty much the same for the C64 version which, by talking to the other coders at Catalyst, we discovered was a really low figure. I learnt from Dave that Wainwright was again

offering £2500 per format for *R-Type* but between the two of us we decided that we weren't going to be ripped off for two and a half grand this time, in fact what we were going to ask for - no.....demand....was £3000!

Dave and I cornered Wainwright in an empty room and spent an hour arguing with him that since we'd both produced a finished game that we were deserving of a raise - and any and all other reasons we could come up with to justify the extra £500. Wainwright countered that he was only getting £4500 from Activision per format so that any extra money coming our way was literally going to have to come out of his pocket. But we stood firm, and in the end managed to get him to agree to paying us £3000 each for our versions of *R-Type* and left the room congratulating ourselves on our business acumen.

I'm guessing once we were gone Wainwright laughed himself silly at the two mugs he'd just managed to con into thinking that £3000 for a game for a major software house was BIG MONEY but that's the rite of passage everyone seemed to go through in those early years. We found out later that Wainwright was getting near £7000 per format from Activision which was when the penny dropped, and if you can remember those *Tom and Jerry* cartoons when Tom looks at the camera and turns into a jackass because of a stunt Jerry has pulled on him you'll get some idea of how we felt. But things have a way of balancing out and a few months later it would be our turn to pull a fast one on Wainwright in similar circumstances.

With hindsight I can sort of see the reasoning behind the choice to assign *R-Type* to me, *Time Scanner* was definitely (at the time) the sexier title and the one with the potential to make a bigger impact while *R-Type* seemed to be yet another horizontal scrolling shoot-em-up. Maybe *Time Scanner* being a SEGA title needed more of a guarantee that it would be completed to a high standard while *R-Type* coming from a little known Japanese company called IREM could be allowed to slip a little, time and quality-wise. Whatever the truth the incentive was there to keep the prestige title in-house at Catalyst and feed the scraps to me in Wales.

Nowadays it's called everything from Innovative to Classic but at the time there really wasn't anything that special about *R-Type* that was making an impact in the arcades. To most people it was just another minor horizontal scrolling shootemup, albeit with a few extra bells and whistles. If it had been that popular then I wouldn't have had to travel to London to play it, Southsea itself boasted a number of excellent video arcades and it wasn't to be found in any of them.

Even Electrocoin, the company who distributed it in the UK, weren't pushing it too hard in their promotional material and there was no buzz or anticipation in seeing it on a home computer. In the end I gave up telling people the name of the game I was working on because it invariably raised the follow-up question "*R-Type*, *what's that?"*

The IREM Corporation started life in 1974 as The IPM Corporation, producing and renting electronic-based leisure items to the arcade entertainment business. When Taito's *Space Invaders* became a worldwide hit in 1978 IPM decided to develop and manufacture video arcade machines themselves and created their own cruder version of *Space Invaders*. A year later they changed their name to IREM - which initially stood for International Rental Electronics Machines but would later become Innovations in Recreational Electronic Media.

For most of the 80's IREM produced some rather lacklustre titles, only *Moon Patrol* in 1982 and *Spartan X* in 1984 (renamed *Kung Fu Master* for the western market) standing out from the rest but all that changed however in 1987 when side-scrolling shoot-em-ups *Mr. Heli* and *R-Type* were released. *R-Type* comprises eight different levels, vaguely 'bio-mechanoid' in nature which in the original release meant very little but gained an elaborate back-story as more and more sequels were released. Game play is pretty standard, clichéd even, as the player flies a small snub-nosed spaceship around a slowly scrolling right-to-left screen filled with obstacles and enemy aliens in set patterns before facing an end of level challenge that has to be defeated before being able to proceed.

R-Type built on the central idea of the earlier *Gradius\Nemesis* games of Konami where the player started off with a relatively weak main ship who's firepower could be increased through the use of Power Ups but added a few unique elements. The choice of which weapon a player selected was a vital part of the game, selecting the wrong weapon could see you getting through to the next level with a bit of luck and good reflexes but there was often a reason to pick a particular weapon at a particular stage of the game. This reached a head on the fourth level of the game where two of the three major weapons you could select would be of absolutely no use in getting you through safely.

Power Ups were gained by shooting a particular alien enemy which would explode leaving behind a coloured jewel icon to be flown into by the player. The colour of the jewel deciding the nature of the Power Up: Extra Speed, The Force, Bounce Laser, Ring Laser, Ground Laser, Extra Ship, Homing Missiles and Power Orbs (two small balls positioned above and below the player's ship.) These are the names that all of us involved in the game at the time called the weapons but today they bear longer, more official titles - the Power Orbs are 'Bits', the Circle Laser 'Counter-air Laser" etc.

The second innovation was The Force, a detachable, invulnerable add-on that the player could fire off and call back to the ship through the use of a second fire button. The Force could be attached to the front or rear of the player's ship where it acted as a weapons modifier, firing the main weapon forwards or backwards depending on position. *R-Type* added to the difficulty factor by having some of the enemies attack the player from behind, defeating these could only be achieved by skilful manoeuvering of the ship or having The Force attached at the rear allowing the weapons to be fired backwards.

A third major innovation was the ability to hold down the Fire Button and build up the Wave Cannon (we called it the Plasma Weapon) which could be released to devastating effect, destroying whole waves of aliens with a single shot. A bar at the bottom of the screen charted the build up of the weapon, which could be released at any time with varying degrees of destructiveness. Unfortunately you could not fire your normal weapons while charging up and it only fired forwards so using it properly became a strategic element of the game.

The eight levels that made up the game were exotic and colourful, often including parallax scrolling of the background along with unique graphics and effects. Highlights of each level included:

Level 1 - A full screen circle of rotating aliens that the player has to fly into, destroy, and fly out of and an End of Level monster with a deadly whipping tail that became the iconic representation of the game in this and later versions.

Level 2 - Giant multi-segmented snakes that circle around the screen before flying into and out of a large anemone-type creature at the end of the level.

Level 3 - A giant spaceship well over two screen lengths in size that the player has to fly around and destroy piece by piece. The ship IS the level in fact and some precise flying is needed to avoid being hit by the ship or its weapons.

Level 4 - Screens filled with small green globes, laid like eggs by some of the aliens that limits player movement and a large End of Level alien ship that separates into three components which move around the screen independently.

Level 5 - Multi-segmented snakes again that this time explode when hit sending pieces of themselves all around the screen and a large End of Level alien ship covered in clumps of grey cells that fly off and try and hit the player's ship.

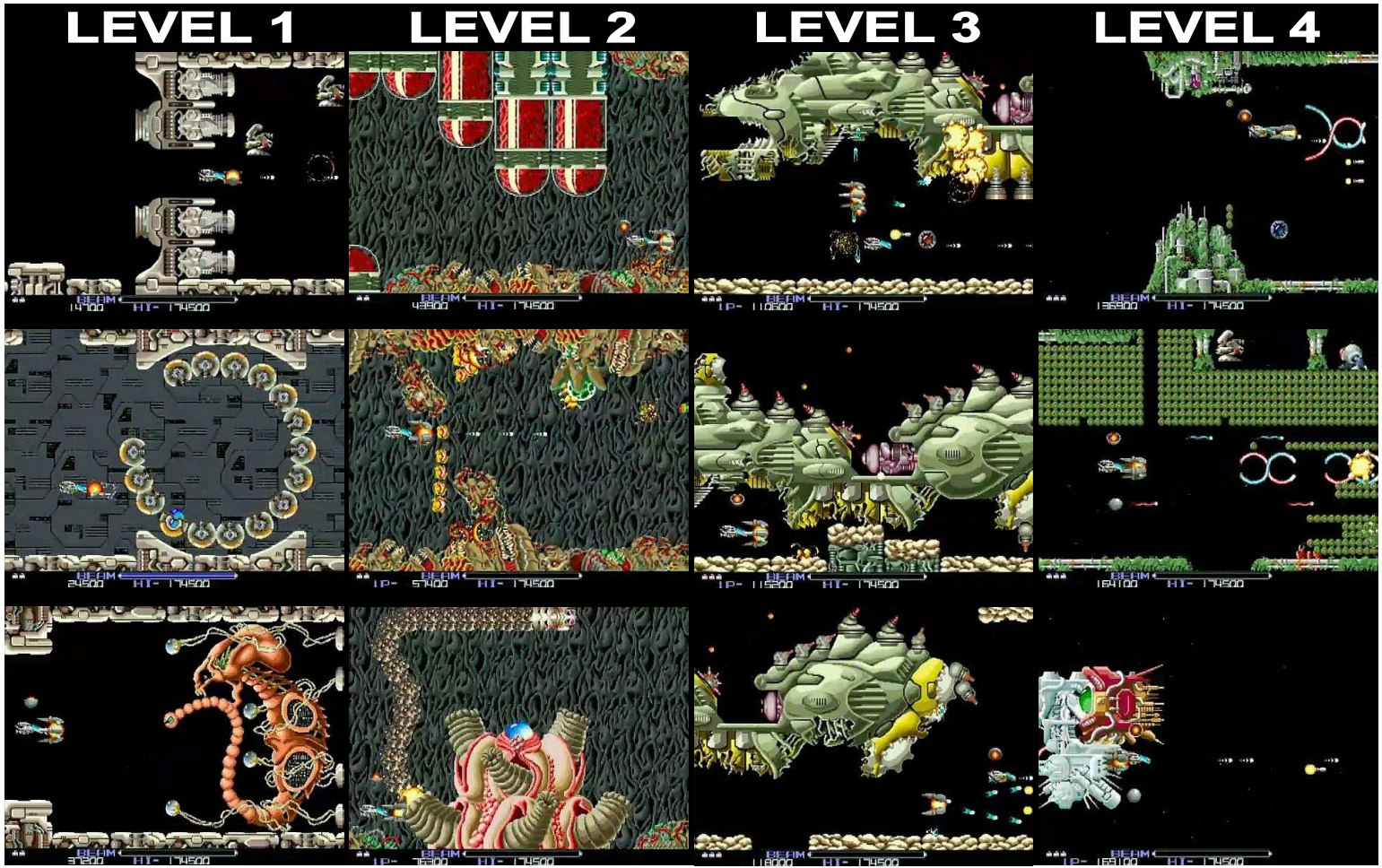
Level 6 - The entire level is a maze with dead ends and large piston-like aliens who move around it trying to crush the player.

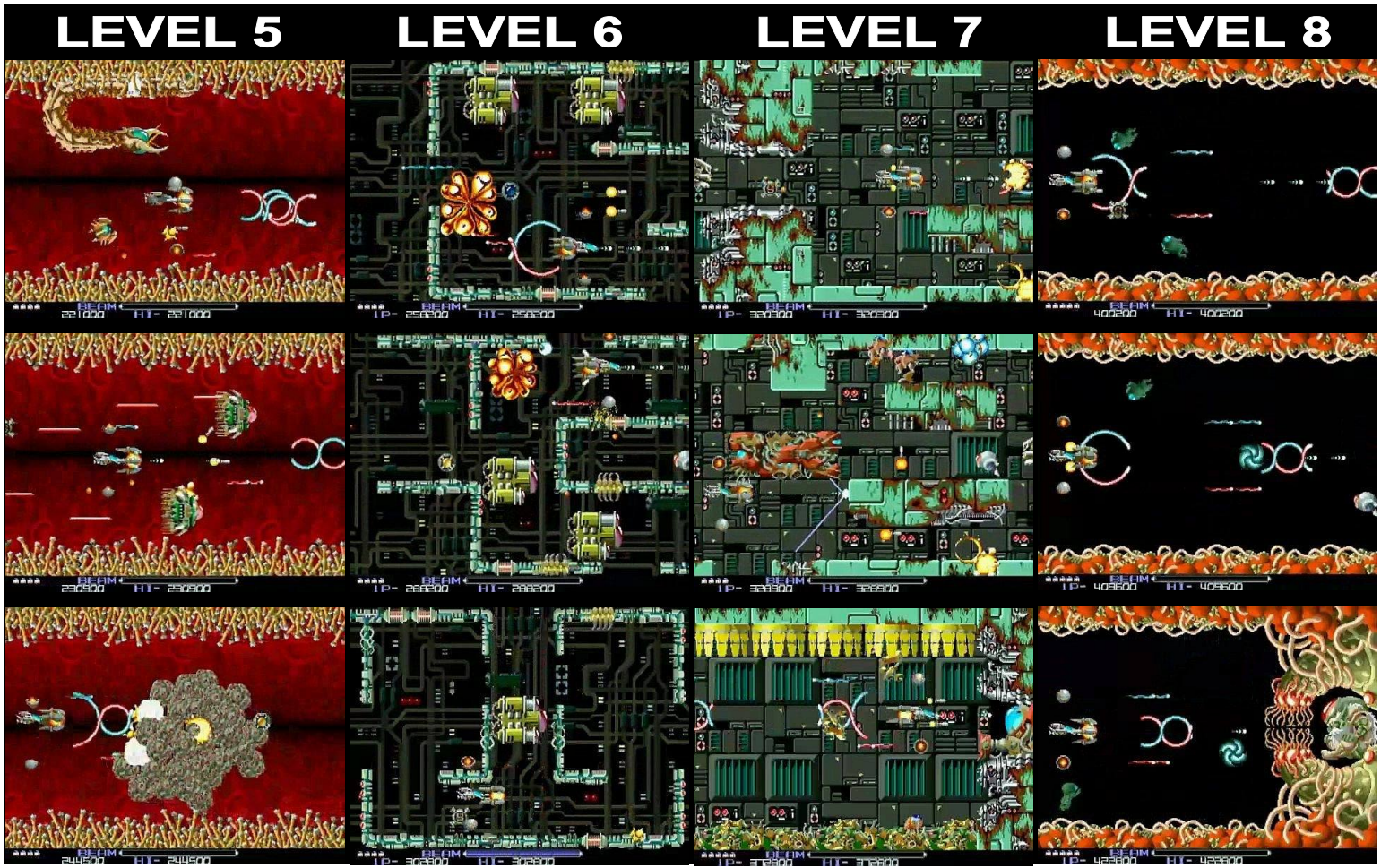
Level 7 - Near continuous streams of aliens firing off a hail of bullets culminating in an End of Level garbage dump where the player has to dodge rubbish being dropped from the top of the screen.

Level 8 - Aliens that can't be destroyed leading up to a final End of Level monster hiding behind a tentacle barrier.

Finishing the game isn't easy, the player has to learn the attack patterns of the aliens and use The Force accordingly and it can be difficult to restart a level after the player ship is destroyed as any Power Ups collected up to that point are removed leaving the player with just the basic weapons. Finding out the correct way to defeat some of the larger aliens adds to the difficulty while the levels get steadily harder and harder, the screen filling up with enemies and alien bullets and the player having to constantly make use of The Force, the Plasma Weapon, sharp reflexes, quick thinking and constant alertness.

The following pages give a pictorial sample of each of the levels (read downwards)


LEVEL 1
LEVEL 2
LEVEL 3
LEVEL 4


LEVEL 5
LEVEL 6
LEVEL 7
LEVEL 8

According to the notes in my source code I started development proper of *R-Type* on February 28th 1988 which probably meant I had just raked through the mess that was the *Rampage* source code and pulled out the routines I could reuse, which to be honest wasn't much. While I had been testing out some different ways to implement the game I'd received a video tape in the post from Catalyst of all eight levels of the arcade version of the game, which in this case meant a video camera had been pointed at a monitor screen and the entire game recorded as somebody played it. The game had been set so that the Player was invulnerable, and luckily it was the ship-is-like-a-ghost-and-things-pass-through-it-unhurt kind of invulnerability rather than it being an unstoppable brick that destroyed everything it touches which meant you got to see the complete walk\fly\attack patterns of every alien in the game without them being interrupted mid-flight by crashing into your ship and exploding.

When we originally visited the arcades to see what *R-Type* was like none of us could get past the first level so this was my first real look at the complete game and I have to say that on seeing the whole thing I was relieved more than anything. Far from being eight unique levels there seemed to be quite a bit of repetition in how thing were put together, Level One was definitely the longest and most varied of them but the reason for this was purely commercial - in the arcade this is the attract mode, the level that would be played the most by gamers so it made sense to make it interesting and put in as many different aliens as possible in order to get the player wanting to see more.

Level Two was big snakes and giant scorpions and that was really it. Level Three was the Big Ship and nothing much else. Level Four was Level One sprites with a changing background. Level Five was more big snakes. Level Six was a maze with giant pistons. Level Seven was Level One again but with more of everything and Level Eight seemed to be where the designers gave up and went home early. I could see that once I'd written all the alien routines for Levels One and Two I'd be reusing them for the bulk of the other levels and all I really had to worry about after that was the big ship from Level Three, the background of Level Four and the odd unique alien that appeared once and never showed up again.

I hadn't forgotten about the End of Level Bosses but looking at the game as a whole there are only really four of those, the other end of levels are only scenery at heart so that made things a lot easier. It was still a daunting task though, even with being able to recycle a lot of the alien patterns, but at this early stage when you're trying to get a handle on just how long everything is going to take and how much work it's going to be I could see it was going to be a lot less than it first appeared and psychologically that gives you a massive boost because you shift your thinking away from "*how the hell can I do all this*" to "*I know I can do all this*" and really start to get to grips with the game.

At home in Swansea I started work on the game, or rather the tools I'd need to write the game, as it was obvious that some sort of level editor would be needed to create and map the levels and also a sprite editor for the sprites and graphics. There were a few of these kind of programs around at the time but they tended to be written by a coder for a specific game or were totally generic in nature. In the former case you usually had to deal with some non-standard way the output data was created or you'd find one that looked promising but forced you to work with tiles or characters in a way that just wouldn't work for your game and in the latter, generic case, you usually ended up with way too much data.

This meant I had no choice but to write my own editors, which would take some time but would produce the graphics in exactly the right layout I wanted rather than trying to convert somebody else's format. When I say "I" of course I really mean Mark Jones, who would be handling the level mapping, sprite design and all other graphics for the game.

Mark had done most of the graphics for all the formats of *Rampage* and everyone knew he was more than capable of doing the same job for *R-Type* but how faithful he was to the original graphics and the quality of the work he produced was to be a major factor in how successful the game later turned out. Mark asked me what I wanted and what he could do and I basically told him to give me everything he could manage and I'd try and put it all in the game, This worked out really well as he would first create the graphics for the Atari ST version in all their sixteen colour pixel glory and then take those same graphics and tweak them a little so they came out looking pretty damn good on the colour-challenged Spectrum. I think if he'd had to create the Spectrum graphics only then things would have come out a lot different but by having to create the ST graphics first, graphics that were almost identical to the arcade version in detail, and then reworking the colours, he got a lot more accuracy into the final result. This approach would pay dividends for Rare in the future with the SNES version of *Donkey Kong Country* so Mark was definitely onto something.

I do have to put my hands up to cheating slightly on the tool coding though, TileD and ED209 (the names I gave to the Spectrum based *R-Type* level and sprite editors) had a copy of the commercial *Melbourne Draw* art package embedded in them which handled all the character creation and drawing functions. Not having to create basic graphic functions from scratch saved both of us a lot of important set-up time.

I have to say that there was no initial blueprint of how the game was going to be written, I mean I didn't sit down and think any of it through. It wasn't a case of "*oh yes, there's going to be minimal colour clash and smooth scrolling and the way I'm going to do it is.....*" all I wanted was to get a reasonably sized scrolling screen pulling data reasonably quickly from a level map and stick a sprite engine over that, and I didn't want to cop out with the colours and do a

nice, safe, monochrome version that everybody else seemed to go with. I knew that approach would work, having played a Spectrum game from FTL called *Light Force* I could see that it was definitely possible, all I had to do was come up with something that worked along the same lines i.e. a background that scrolled on pixels and player and enemy sprites that moved on characters. I had an advantage over *Light Force* in that my sprites never had to move over any part of the background so didn't have to do any masking, either through big chunky black outlines (like the later Electric Dreams game *Karnov*) or by changing the colours of the sprites so they didn't stick out like a sore thumb (again like *Karnov*, and to some extent parts of *Light Force*).

I stumbled across the answer by accident, all because I can't draw to save my life. I had decided to map and store the game levels as 4x4 character tiles rather than individual characters, so for example instead of a 32x20 character section of a map taking up 640 bytes it would only take up 40 bytes - 32/4 x 20/4 - each one of those bytes referring to a tile made up of sixteen characters - which of course saves a huge amount of memory.

With the in-game graphics not yet available from Mark I was writing my tile and scroll routines with dummy tiles, specifically big coloured triangles, circles, diamonds etc. simple graphics that even a non-artist like me could knock up and funnily enough it was this amateur approach of mine that gave me the first inkling that maybe it was going to work. In my cack-handed drawing attempt I didn't quite fill up the tiles right to the edges, in fact I had big empty character-sized spaces all over the tiles because the less characters I had to use to make up a tile the quicker I could get them in to test. So after I had got the tiles being decoded properly onto the screen and had written a really basic test scroll routine to get things moving I was fully expecting to see a big colour-clash mess as the level scrolled across the screen. But…...all these different coloured shapes were sailing slowly across the screen and not a colour clash to be seen anywhere!

Of course it was those big spaces between the shapes in the tiles that was the ‘secret’ to the lack of colour clash as what happens is if there's always a one character empty space buffer between pixel scrolling characters of a different colour then there's never any chance of the data from one character picking up the colour Attribute from another. Oh, and it only really works if all your characters share the same PAPER Attribute (and that includes the empty buffer character) which, once I'd ditched the idea of duplicating the parallax scrolling of the original worked perfectly against an all black background.

The irony in all this was that if I'd put in the proper game graphics instead of all my dummy stuff there'd have been no excess spaces, the colours would have bled all over the place and I'd have been left with having to go down the monochrome road. But luckily I'd spotted this right at the start so got on to Mark and told him he could use whatever colours he wanted for the backgrounds provided

IREM Japan's publicity material for R-Type - Front page of a handout

Back page of the handout explaining the basics of the control system

he left a one character black space between characters that had a different horizontal colour, and since the game didn't move up\down he could put whatever he liked in vertically.

That's all there really is at the heart of *R-Type*, the background scrolls one pixel at a time and nearly all the sprites - the ship, aliens, bullets, weapons etc. move eight pixels at a time, on character (or rather Attribute) boundaries with most of the sprite being a coloured INK with a Black PAPER Attribute so as to blend in with the usually black game background. If you want to be pedantic about it the background is really just the black empty space part of things while the graphics that make up the levels - the scenery and solid stuff scrolling on pixels that your ship can crash into - is the foreground, with all the game sprites being sandwiched between the two. Since nothing is supposed to go over the foreground there's no need for any masking or colour changing of the sprites. Just take it as read that when I'm talking about 'the background' I actually mean a combination of both the scenery and black space.

All I needed to do next was prevent the sprites from interfering with the background graphics and find a way of speeding things up because, even at this early stage, everything was running just too damn slow!

The equipment I was using to write *R-Type* with at this time was the same as for *Rampage*, a standard 48K Spectrum with Interface 1 and microdrives and everybody's favourite hardware copying/backup device Romantic Robot's Multiface 1. I had a copy of *OCP Editor\Assembler* on the microdrive along with my source code and with a push of the button on the Multiface I could go from an empty Spectrum to one ready to assemble in just a few seconds, however the drawback to this method was that the assembler was resident in the Spectrum's memory taking up valuable RAM that I couldn't use. With a bit more Z80 experience under my belt I suppose I could have worked something out with the 8K of RAM that came with the Multiface or come up with some binary file-joining method but that RAM restriction definitely played a part in restricting some of the features I wanted to put in to *Rampage* and I could see the same thing happening with *R-Type*.

When I was finishing *Rampage* at Activision programmers Zari (aka ZZKJ) and Chris Wood were there working on the Spectrum version of *Super Hang On* and using an Amstrad PCW 8256 with a cross assembler to code the game so I had a word with them about how it all worked. I remember reading a magazine article where Matthew Smith said he'd used a TRS-80 to develop *Manic Miner* and that some of the Tandy code was still in the finished Spectrum version but I just couldn't get my head round the concept, I mean, the TRS-80 screen was nothing like the Spectrum so how could it even display properly? Talking to Chris soon sorted it all out and I knew that I'd need something similar for *R-Type* so I spoke to Kightley and he got hold of an Amstrad 6128 for me, but it was pretty much of a brick for a while as I had to wait for an

add-on serial interface to arrive before I could do anything with it.

Eventually things were sorted out and I could now assemble on the 6128 and send the compiled code directly to the Spectrum via the RS232C connector on the Interface 1, all it needed was a small loader on the Spectrum end to download the code. But in the meantime, while I was waiting for all this hardware to come together, I'd already made a start on the basic graphic routines for the game.

None of us had any idea what the aliens in the game were supposed to be called so I just made up my own names for them. This made them easier to identify when I came across them again in later levels and usually the name pretty much described what they were e.g. Red Twisters, Butterflies, Walkers, Blue Gobot, Nodding Snailiens, Podders, Missile Walkers, Spermies, Brains in Cases. It also made it easier when I was speaking to Mark on the phone to refer to something as Crawling Babies rather than "*those big headed red things that appear on Level Eight*" and before long he and I were talking and thinking pretty much alike.

Mark was starting to produce the proper graphics now but of course this was in the days before the Internet and we had to rely on cassette tapes in jiffy bags through the post to get them to me which really wasn't too much of a problem at this early stage but we could see it would be as we got further into the game. Mark was always spot on when it came to Spectrum graphics, especially when it involved repeating patterns and getting the most out of a character as he knew how tight things were on the memory front for me. As all the sprites were going to be character based even one single pixel in a character would mean having to store all 8 bytes of that character data and would make the collision detection appear really ropey since the entire character would then be classed as part of the sprite. But Mark was well aware of this and made sure that what I got was optimised for memory as much as possible.

I had a simple compression system that I was using for the sprite data which picked up on empty bytes and discarded them then used a bit flag to represent the missing data, it wasn't great but it was really fast when it came to decompressing the data and it did free up some badly needed space. Of course with eight individual levels the game had to be a multi-load since there was no way everything would fit into 48K of memory (which is really more like 40K when you discard the screen RAM) and with Level One being the largest of all the eight levels it was that that set the limits on how much memory I could use for the game itself and how much I had to set aside for the Levels.

Each game level was comprised of the tile map for that level, the 4x4 tiles, the character data that made up the tiles, the alien sprites, the data that made up the sprites, the alien movement patterns, control data for each alien, special code that was unique to some aliens and finally an attack table. The attack table was what triggered the aliens to appear, as the background scrolled it

kept track of how many characters into the level it was and how many pixels into a character and when an entry in the attack table matched those values the alien was switched on and off it went.

I thought that having to record all the attack patterns for the aliens was going to be hard work but to the contrary I found myself actually enjoying the procedure and used the same approach on all later arcade conversions I worked on. Originally I had planned to tape a sheet of clear acetate over the TV, play back the video of the game I had received and use a felt pen to trace the patterns of the aliens onto the acetate as they moved across the screen. I actually tried this idea out for a whole five minutes before I realised what a stupid and impractical way of doing things it was - too much was happening too quickly - and all I ended up with was a mass of hastily scribbled lines. Time for Plan B.

I drew a load of rectangles about three inches by two inches onto a piece of paper and made use of the local library photocopy machine to duplicate a few dozen copies. I'd start the video rolling and wait for an alien to appear and as soon as that happened I'd pause the tape, make a rough sketch of what the background looked like in one of the rectangles, make a mark on the rectangle corresponding to where the alien had entered the screen and restart the video. Then I'd ignore everything else and just watch the pattern the alien made and try and duplicate that by drawing it in the rectangle until it exited the screen. Most of the time an alien would enter from the right, fly across the screen and exit off the left but occasionally there'd be some interaction with the background so I'd have to try and put that down as accurately as I could and in those cases I'd usually use another rectangle to show what position the background was in when the alien finally exited the screen so I'd have the timings right.

Basically I was storyboarding the game one alien at a time but there were all sorts of time saving methods to cut down on the sheer numbers because a lot of the aliens would come in waves or in identical patterns, and in that case all I needed to do was record the pattern the first alien in the wave made and then make a note as to how many were following behind it. You can see just this at the start of Level One and it's exactly how all the big snakes were done, just draw the pattern the head makes and every segment behind it uses the same pattern but offset a few frames - it saves a lot of memory as well. I'd leave all the pattern recording for last thing at night, spending about half an hour in bed just drawing these patterns. I found it quite restful and a good way to unwind after coding during the day and the constant viewing and re-viewing of the game a few seconds at a time meant I was soaking up a lot of information and small points about the game I would have otherwise missed.

The next few pages go into some technical detail about the game itself so if you've got no interest in how and why the game does what it does then feel free to skip them. You won't miss out on anything.

A UK television set displays pictures at 50 frames per second (in truth it interlaces two slightly different fields to give a display resolution of 25fps) so if you want to see anything on the TV the Spectrum has to output a video signal at 50fps as well. The actual mechanics, timing, ULA control etc. are extremely complicated, I never understood them back then and I don't now so I'm not going to try to explain electron beam flybacks and the slow down of contended memory, but just go with the idea that every 50th of a second the Spectrum squirts out a screen that the TV can display.

The Spectrum has a byte in memory that gets incremented every time it displays a screen, 50 times a second, come hell or high water. Thanks to that counter byte we have a handy little tool that lets us do two things, we can wait for the screen to display, set the counter byte to zero, carry out some code, then read the counter byte again and find out in 50ths of a second just how long it all took. Or we can wait for the screen to display, set the counter byte to zero, carry out some code, read the counter byte and wait...and...wait...and...wait until it reaches a number we want and then we know that no matter how quickly or slowly the code runs each and every GAME frame is going to take exactly the same amount of time.

If you want a home computer version to run and play and feel the same speed as an arcade one does then surely you need to know such things as what speed does the arcade version run at, what's the screen size in pixels and how fast is the processor? No, they're irrelevant. The only thing you need to know is how long it takes one complete screen of the game to scroll across the display area and from that you can work out how many frames per second your version should run at. I say "should" because at this stage it's just a target to aim for but at least it gives you some idea of just how far away you are from the bulls eye.

The arcade version of *R-Type* runs at slightly over 12 seconds per screen (the time it takes one screen of the game to scroll from left to right) and since my screen was 240 pixels wide with two blank 8 pixel strips on the left and right to hide the scroll edges a bit of maths give us (12.5 seconds * 50fps) / 240 pixels or 2.6, rounding up to 3. So as long as the Spectrum version managed to do everything it was supposed to in 3/50ths of a second the game would scroll and play at approximately the same speed as the arcade version.

Most graphic intensive games utilise a back-screen, a block of RAM to which all the graphic data is written prior to being moved to the main display, which helps prevent flickering and also means you can set up your own addressing methods rather than use the unique Spectrum screen layout. A back screen is really the only way to go which, in a heavy arcade game like *R-Type*, means a game cycle of four steps something like this:

1. Blank out the back screen.
2. Decode the tile data and write it to the back screen, appropriately pixel shifted so you have your movement.
3. Write all your sprite data to the back screen.
4. Move the contents of the back screen to the main display as quickly as possible, preferably during the vertical fly-back phase of the CRT gun.

That's an awful lot of data to be moving around every few 50ths of a second and unfortunately I didn't have the luxury of working with an original concept where I could change things like the maximum number of sprites displayed, the dimensions of the screen or the complexity of the background just to shave off some processing time. You can bypass a few of those steps completely by not using a back-screen and writing directly to the main display screen instead but you'll find that if you don't sync everything up to the Vertical Blanking (fly-back) of the display then your graphics are going to tear like mad. You'll also see a nasty flickering all over the screen as the graphic data is being erased and rewritten in front of your eyes.

You can chase the signal being put out by the electron gun of the CRT (or it can chase you) and you end up in mid-write, putting down a graphic slightly before or after the screen has been written to causing it to literally split - you may want to put a straight vertical line on the screen but what you get looks like a broken stick!

Every coder tries to find ways around those four steps or to minimize the amount of time they take and the methods and techniques I initially tried out during testing to speed things up were far from unique but still it was all taking too long. With plenty of memory you can merge Steps 1 and 2 by having a second back-screen which holds just the background scrollable graphics and as you are processing the scroll you move them to the first back-screen, automatically overwriting and clearing out what's already there, but an abundance of RAM was a luxury I didn't have.

I did know that there were going to be a LOT of sprites moving around so made sure to write quite a bit of dummy data down when I was testing various things, and found the time it took just got longer and longer. Usually you can get away with some slow down in a game when the screen starts to fill up (even the arcade version of *R-Type* did this) but this was almost the opposite, the game would run slow most of the time due to the abundance of sprites and then speed up when nothing was happening!

Scrolling the screen one pixel at a time at a rate of 3/50ths a second per Game frame means that it takes 14 seconds to scroll an entire screen, while 4/50ths takes 19 seconds, 5/50ths 24 seconds, 6/50ths 29 seconds etc. You have to stick with

these times as it's the only way to get a regular, paced game, anything else exposes you to speed-up and slow-down depending on what's on the screen and there's not a lot you can do about it. 29 seconds to scroll a screen is just ridiculous, 14 seconds was impossible with all that I had to do which meant I had no alternative, things HAD to run at 4 or 5/50ths of a second per Game frame.

To sum up - you can write everything to a back screen in RAM and then try and copy it to the main screen display as quickly as possible, you get a lovely smooth display (provided you beat the TV's electron gun) but doing everything twice eats the processor time up and uses memory you can't afford. Or, you can write directly to the main screen and cross your fingers that you don't get a glitchy\flickering display, but boy is it fast and that chunk of RAM you were going to use as a back screen can be used for something far more important like more game code.

I will qualify that last paragraph by saying that for the vast majority of cases a back screen is the ideal way to go, think of *Pac-Man* with a static screen and only four sprites or a game that uses masked sprites like *Rampage*, in one case you have plenty of processor time to do what you want and in the other it's the only way can get the desired effect. With a game like *R-Type* where the screen is continually scrolling and lots of large sprites and bullets and weapons are flying around the screen almost non-stop it's not a case of which approach to choose, there is no alternative, it has to be a direct write to the main screen, but even using this method was working out too slow, I was aiming for 4/50ths of a second per Game frame and unless I could come up with something radical I was never going to be able to achieve that.

What I needed was a way to merge the smoothness of a back screen with the speed of writing directly to the main screen, to dramatically cut the time everything was taking and minimise the flickering on screen. It took a bit of trial and error but eventually I came up with a solution that did everything I wanted.

At the heart of everything are two blocks of RAM, each 640 bytes long, that are in effect collision detection maps for the game and 640 bytes because if you looked at the 256x160 pixel main screen display (and I mean just the scrolling play area not the score\Beam\Ship counter lines underneath it) then you have an actual Attribute or character resolution of 32x20 characters. For simplicity sake I'll call them Collision Map 1 and Collision Map 2. There's also a back screen of 6912 bytes, duplicating the main screen completely and including the Colour Attribute area which even though it was over 1K too big for what I needed made screen addressing a lot simpler and came in handy when I needed some scratch RAM to play with. During my testing with dummy data I had come up with three ideas that seemed obvious and they were:

What was the point of scrolling empty blank characters?

Why erase a sprite character if in the next frame another one was in its position?

If parts of two sprites shared the same character block then why waste time drawing the one you wouldn't see?

Before a level started the 4x4 character tiles that made up the background for one complete screen was decoded and the characters were written to Collision Map 1. To draw a complete screen all I had to do then was step through the Map one byte at a time and get the eight bytes of data that corresponded to the character number and write it to the correct position on the main display screen

Again, before the level started, I would see what sprites needed to appear on the very first frame of the game (usually just the player ship, The Force, ship exhaust etc.), get the data required and write it to the 6912 byte back screen, using Collision Map 2 to flag the identity of the particular sprite that filled that particular character position, ignoring empty characters.

When it came time to display the first frame of the level I would, using Collision Map 2, search for non-zero values and copy the 8 bytes of data that corresponded to the flagged character position from the back screen to the main screen as quickly as possible.

At this stage the state of everything prior to a normal game frame is:

- The background graphics exist only on the main Spectrum screen display and nowhere else
- Collision Map 1 has the background mapped out as a set of bytes corresponding to the character positions on the main screen. If it's a zero then it means no background in that character.
- Collision Map 2 has all the sprites mapped out in a similar way to Map 1, but instead of the main display screen they point to data that exists on the back screen.
- The back screen contains all the sprite data to be put down in the next game frame.

Once the sprite data has been copied across to the main display then the game starts properly, and (generally) for every game frame the following happens.

1) The background is scrolled to the left one pixel. If one complete character of data has been scrolled then; the Attributes are scrolled one character, Collision Map 1 is scrolled horizontally left one character, new characters are pulled from the tiles and written to the rightmost edge of the screen ready to be introduced the next frame.

Just to explain the scroll process, Collision Map 1 is scanned a line at a time for horizontal strips of

continuous non-zero characters and if any are found then only that particular strip of characters is scrolled. As you can imagine this happens quite a few times over the entire screen but since you are only scrolling background that has characters in it speeds thing up considerably since there's no need to scroll empty characters, A normal full screen scroll would just waste a lot of time moving empty space about for no reason.

2) Collision Map 2 is filled with zeros, erasing the previous frame's sprite information. There's no need to clear the back screen every frame since it's just a holding area for the sprite character graphics before being drawn to the main screen .

3) Alien sprites are moved and their graphic data written to the back screen a character at a time and a corresponding entry made in Collision Map 2 for every non-zero character in that sprite. But prior to doing any of that a check is made against Map 1 to see if a character has moved over part of the background and against Map 2 to see if a character has already been drawn in that position. If either of the checks returns a non-zero character then nothing is written, this prevents the sprite from drawing over a background character or wasting time drawing data that has already been drawn into that back screen character.

4) The player moves his ship and fires his weapons. Using Collision Map 1 we can check if the player has collided with the background while using Collision Map 2 we can check if the player or his weapon has hit a sprite and deal with it accordingly.

5) Collision Map 2 is scanned through a byte at a time, if the value is zero then the corresponding character on the main screen is filled with zeros, overwriting any graphics from the previous frame. A non-zero value indicates that data from the back screen should be used, again overwriting any previous graphics.

And that's all there was to it.

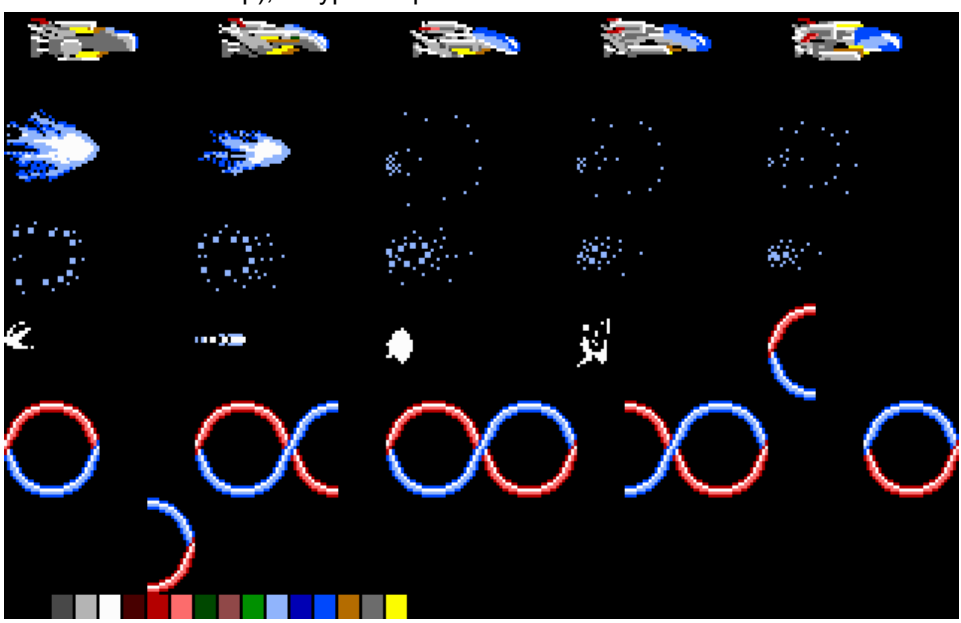
Start screen of the Spectrum version, all strapped in and ready to go?

If you decided to skip the preceding pages then all you need to know is that I now had a workable scrolling screen and a sprite engine that was running reasonably quickly with still a bit of leeway left for all the extra coding that needed to be done.

Getting each alien to appear in the right place at the right time and follow the right movement pattern was a hard slog, especially at the beginning where I had no previous instances of that alien to build on. As I mentioned before I had a rudimentary idea of when and where it appeared and what it then did thanks to my storyboard rectangles but I now had to get it off the paper and onto the screen and there was only one way to do that - keep trying and trying until it looked right. Take for example the Podders (those aliens you shoot to release a Power Up), a typical sprite sequence would see one enter from the right side of the screen, fly down and land on a piece of the background, take a few steps forward, wait, then launch itself and fly off out of screen to the left. Once I'd identified the correct time to bring on that particular Podder (by comparing the background against my storyboard of rectangles) I'd break down the flight pattern into smaller sub-sequences - fly right, walk left, wait, launch, fly right - and then put them back together so that:

a) It looked as smooth as I could make it given the limited screen resolution of the Spectrum and
b) It looked as close as I could get it to the arcade original.

The idea of the sub-sequences was that I then had building blocks I could use for other instances of that alien, so when the next one appeared with a different sequence of



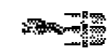
Mark's original ST graphics of the main ship and Circle Weapon

movements I could take the parts I'd already written, arrange them in a different order and not have to start from scratch each time.

The other advantage of using sub-sequences like this is that it saves enormously on memory. If for example I had an alien that moved across the screen flapping it's wings and moving up and down every few frames then I'd only really need to write a small sub-sequence lasting those few frames and put it in a loop, thus saving me a lot of memory (and work.) Of course you have to take a different approach if an alien exhibits any kind of 'intelligence' such as homing in on the Player or changing it's sprite sequence due to avoiding the background. This idea of not writing a fixed sprite sequence but rather having some sort of A.I. that does all the movement for you was something we later came to suspect that the arcade original was implementing in a few places. The aliens we called Walkers and Missile Launchers and, to some extent the Podders themselves, all exhibit this simple logic at times but none of us ever thought of actually trying to write it ourselves, it was simpler to just hard code as much as we could.

What at first appears to be a disadvantage - the one at a time, glue it all together piece by piece until it looks right approach - actually turns out to be quite a good idea the further you get into the game because you find yourself grabbing sub-sequences and whole sprite sequences to reuse in later levels. You can even get away with using pre-existing sub-sequences for other aliens, keeping the movement patterns and changing the sprite numbers to create brand new alien sequences without having to work out complicated movement patterns again and again.

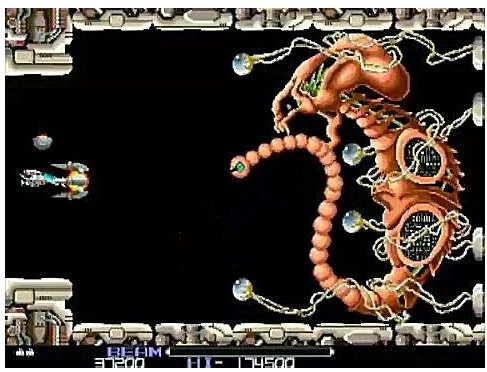
But starting out it is slow and that first level took quite a few weeks to come together properly since I had decided that I wanted a near-complete and playable Level One up and running before starting on the others. It's a bit of a risky thing to do since all you get during this time from the people in charge is "*haven't you finished that first level yet*" and "*I hope the rest of the game doesn't take this long*" as well as making a lot of people feel nervous about your abilities, but it did mean I'd have all the major core routines in and working plus a good library of ready-coded aliens for use in later levels under my belt.



At the end of Level One comes the Level One Boss, an H.R. Giger inspired bio-mechanical creation with a large whipping tail - and if there's one thing that sums up *R-Type* more than anything else it's this. Appearing as box art, on advertising posters, in full page magazine ads and as a sample screen in nearly all reviews of the game this iconic alien has become the standard bearer for the game. Dobkeratops (to give it its proper name) is usually the one thing most players look for when playing any of the *R-Type* games or conversions so getting it onto the Spectrum would be a challenge. But to me as I was writing the game it was just another funny looking monster.

The sheet of clear acetate reappeared again only this time with a neat grid of squares printed over it,

which is how I managed to get the whipping tail into the game. Looking at it now I can see it's a pretty poor copy of the arcade version which is much smoother and more sinusoidal in nature whereas my version seems quite stilted and juddery, but then I had to move everything eight pixels at a time so I suppose it was inevitable. This time taping the acetate to the TV screen, single stepping through the video and making a note of which square contained which section of tail actually worked out quite well and with a bit of tweaking I managed to get the whole tail movement into just sixteen frames of animation.



End of Level 1 Boss - Arcade.

End of Level 1 Boss - Spectrum. Everything looks a lot bigger but the pixel resolution is actually much less compared to the arcade version.

It was now around mid May of 1988 and Level One was pretty much complete along with the player ship and weapons and the collision detection and some basic explosion effects so I decided to make a start on Level Two. I'd been working on the game for about two and a half months and I knew I could handle everything to come, I didn't know how I would but it was that psychological boost at work again saying "*oh you'll work that out when you come to it, don't worry*" - so I didn't. If you're not careful you can spend more time worrying if you're capable of doing something rather than trying it and finding out for sure, something I've unfortunately seen happen to other game coders and the half completed programs they've left behind as self-doubt overrode everything else.

I thought things would carry on pretty much as they had been but a few days after I started Level Two I got a phone call from Karl Jeffrey and everything changed. I'm sure Karl had done some of the graphics on *Rampage* for me when Mark wasn't available toward the end - but I could be mistaken - because to be honest I'm not sure how I first met him, you'll have to chalk this one down to memory loss on my part I'm afraid. Karl had an offer for me, he was coding the Atari ST version of *R-Type* at his office in Fareham in Hampshire and he also had Dave Jolliff there with him coding the C64 version. He was wondering if I fancied coming to Fareham and working alongside the two of them on the Spectrum version as well?

Karl threw in a few sweeteners, first off he had an arcade *R-Type* board there in the office so instead of squinting at a copied video I'd be able to play the game proper whenever I wanted and get everything first-hand. Secondly Mark Jones lived quite close to Fareham so I'd be able to talk to him face to face about the graphics and any problems I was having with them - and instead of having to wait days for the post to arrive he could drive over with them when they were ready. Activision were still based in Southampton at this time, which being about fifteen miles away from Fareham was another incentive since it meant I could start getting some proper feedback from them as to how they liked the way the game was going (or not.) One last thing was that Southsea, where Catalyst was based, was also nearby which wasn't really a plus but it did mean I could get away from Wales and having to deal with Kightley whenever he wanted to butt in.

All in all it was a bit of a no-brainer and a few days later I packed some clean clothes, my Spectrum and Amstrad and all the documentation I had and caught a train to Fareham.

Today the Climax Group is a major independent software house successfully producing software for modern game consoles for such well known companies as Microsoft and Konami and running under the control of CEO Karl Jeffrey, but back in 1988 the company was called Images Software and things were a lot different.

Karl's father had a shop selling kitchen units at one end of the main street in Fareham and Karl was using the upstairs part of the building as the offices for Images. Entry was via a heavy door on the side of the building and up some stairs which brought you into a reception area, leading off from that was a small conference room and beyond that a large open plan area where the programmers were set up. From there you could go down a small corridor to the rear of the building passing a toilet, a kitchen area and a big room filled with junk and boxes of stuff that no one dared go into for fear of getting lost until finally you were right at the back of the building and in the sleeping quarters.

Karl had envisaged all three of us living at the office while we finished the game so had filled the back room with some beds and a few wardrobes to make up our very own school dorm. That back room turned out to be the calmest place in the whole building, away from the traffic and the sun at the front it was a cool chill-out area that I often used when things got a bit stressful or I needed to get away from the keyboard and have a think. When it came to actually sleeping there then for most of the time I was the only who did, not really having any alternative. Karl would usually leave in the evenings to spend time at home with his parents and Dave would adopt this strange method of sleeping on his chair in the office, kneeling on the floor and resting his head on the chair seat which looked uncomfortable as hell but seemed to work well for him.

So there we were, Karl, Dave and myself all set up with our development systems in a room above a kitchen supply shop in Fareham and working on a game that very few people had even heard of. Dave had a friend\apprentice working with him by the name of Jim Smart, the two of them had done some early work on a C64 version of *Time Scanner* that never saw the light of day and also an original futuristic *Ikari Warriors*-type game that they were polishing up in hopes of a sale. Karl was coding the Atari ST version with occasional help from one of the nicest guys in the world, Rob Hylands (who I will come back to in more detail later on) and running the business side of Images at the same time.

Of course the first and most important thing I did on arrival was to see how well I was doing against the other two versions and was pleased to see that I was way out in front. The ST was a relatively new machine at the time so I think Karl was still on a bit of a learning curve with it, but all that sixteen-bit power meant that he wasn't always having to find ways to get around the limitations of the hardware as us eight-bitters were so he pretty much knew that what he wanted to do could be done, he just had to sit down and write it.

During the time I was there I watched Karl's code output accelerate as he got more and more familiar with the ST until at the end of my stay in Fareham it was almost effortless. Karl would sit down, decide what he wanted, start coding and in a few hours it would be done. Compared to my stop-go method of coding (i.e. first find enough memory to fit a routine in, work out how the hell I was going to do it, then go back and optimise it as much as possible to get the speed up while balancing space considerations) it was no wonder Karl managed to get such high quality output ready in such a short time.

Dave and the C64 seemed to be cracking on, all the background graphics for Level One were in place and he had a ship flying around the screen with some aliens but he did have a major obstacle to overcome, the sprite multiplexer routine he was using just wasn't capable of handling the large amount of sprites necessary for the game. The C64 can only display eight hardware sprites a frame unless you implement a sprite multiplexer to ‘recycle’ those sprites again that frame but it's a very intricate and time-critical thing to code, all the more so if you want a lot more than eight sprites on screen in any one display frame. Get it wrong and sprites don't just flicker they vanish altogether. So before he could do anything else Dave had to completely rewrite his multiplexer, and more importantly show Activision that the new sprite engine he had was capable of handling everything in the game.

We knew Activision were prepared to accept some differences between the arcade and home computer versions (it was made clear they'd accept ANYTHING halfway decent bearing the name *R-Type* that appeared on the Spectrum) but more was expected from the C64 version. The question then became just how many sprites could Dave implement per frame to get away with a game that would satisfy Activision?

Things at Fareham soon settled into a regular routine. Monday to Friday would usually involve getting up around 9am, grabbing a cup of tea and some toast and starting coding. This would go on until maybe 11 or 12 in the evening with the odd break of an hour here or there for a wander around the town to have a think or a bite to eat or just get some air before off to bed ready for the next day. Saturdays were usually the same but I did manage to get up to London on the train now and again for a day out - since I didn't drive I was pretty much stuck in Fareham for the duration so any chance to get clear of the place for a few hours was more than welcome.

Sundays would find me all alone in the office with Karl at home and Dave spending time in Southsea though I did tend to take things easy with the Sunday papers and a trip to one of the only shops open, McDonald's, for my Sunday dinner of Chicken McNuggets and chips.

Something the office did lack was proper washing facilities so once a week an early morning visit to Fareham Leisure Centre was on the cards, ostensibly for a swim in the pool but in reality for a nice long soak in the hot showers along with a bar of soap and a bottle of shampoo. Clean clothing was catered for thanks to a handy Laundromat across the road from the office.

Highlight of the week was definitely Friday evening since it involved a trip to the Chinese takeaway a few doors down and the subsequent consumption of Beef with Bamboo Shoots and Water Chestnuts poured over a plate of chips and a hot pineapple fritter in syrup for afters. Since Karl didn't have a TV licence for the office we weren't supposed to watch any television but it didn't matter, the reception was so dreadful that it was a waste of time trying to, which meant that other than coding the game there really wasn't much else of a distraction there.

We'd make the odd trip down to Southsea to the Catalyst offices or over to Southampton to see Activision when we had something to show them but we were left pretty much alone to get on with things ourselves. Nowadays a game will be Project Managed down to the last minute and maybe Wainwright was taking the flak for us but no one was on our backs, there were no milestones or formal deadlines to stick to other than since the game was due to be released in November for the Christmas market it all had to be ready by October.

Compared to Level One, Level Two was pretty straightforward to code, but getting the large snakes that appear at the end of the level to curve smoothly took a bit of work especially with the lower movement resolution. There are five different snake patterns on this level but since most people know the trick to finishing the level quickly a lot of the work I put into the later ones is never seen. Oh well. Level Two has what I think is Mark's best work in the whole game, from the large glass domed background graphics to my favourite of all the alien sprites the ‘Brains in Cases’ topped off by the giant End of Level

Boss that *R-Type* fans will tell you is called Gomander but we just called The Giant Mollusk.

The ironic thing was we actually did have all the proper documentation from IREM available in the form of a large manual packed with names, graphics and information about the game but we never got around to seeing it. Activision had decided to 'help' us out by sending the manual off to be translated - it being written in Japanese of course - and even though we kept asking for a copy it never appeared. In a masterpiece of timing just as all versions of the game were about to be released we got a message that the translation was now finished and did we still want to see it?!

I have to say that IREM were always pretty good at supplying support materials and documentation for developers, later game projects saw hard copy printouts of all the code used in the arcade boards including page after page of raw number data that were the graphics. For another project we received a colour print of every sprite used in the game in every animation frame it had. But while coding *R-Type* our primary source of information was the arcade machine that was set up in a corner of the office which luckily came with a user manual and through that we learnt which DIP switches on the board controlled free play, invulnerability, difficulty etc.

If any of us wanted to check on later levels we would have to start the game in invulnerable mode and then wait a while until it got to where we wanted but things could get a bit hairy as we'd often need to flip DIP switches while the game was running which involved leaning over a live hot board trying to push a tiny switch with a highly technical non-conductive tool - a plastic pen top! We'd have to do this to get past the End of Level One Boss as in invulnerable mode it would

Mark's ST graphics for the Giant Mollusk, the end Boss of Level Two

just sit there spewing out aliens for a while before it scrolled off the screen leaving the ship in a black void until a long time later some internal counter rolled over and started Level Two. One of the DIP switches would activate a Pause Mode so anyone working on a specific level could stay on it all day if they wanted to as it wasn't just the graphics and sprite routines we were trying to copy, we were all looking to get the game-play down as close as we could and the only way to do that was to play the game over and over again.

Along the way we'd notice small things that weren't visible in the video of the levels - sprite sequences that only happened when you killed an alien, unseen animation frames, how the alien AI worked when you did certain things - even down to seeing which kinds of explosion happened for which sprites. I knew I was making a rod for my own back trying to get as much of this as I could into my version but I had the time and it wasn't as if I had anything else on my plate.

We hadn't all been at Fareham long when Karl got a message from Activision saying "*can you all come over to the office, Rod wants to say Hello to you*" which got us all wondering. 'Uncle' Rod Cousens was the then head of Activision UK and didn't really get involved face to face with the coders, that's what Producers were for, so we were all trying to work out what this Hello REALLY meant. Since our monthly monies from Catalyst had been a little late in coming recently I was of the opinion that this was going to be some kind of Golden Welcome bonus, a cash sweetener to keep us happy and on-side, so one hot sunny afternoon we all piled into Karl's red open-top 'babe-magnet' sports car in high hope and drove down to Activision with the music of Yello blasting out at high volume over the car speakers.

As I mentioned before Activision UK at the time were based on the top floor of an office block in a pretty bland part of Southampton and unless you knew where it was you'd never find it, a small nameplate over a door intercom was the only identifier. The offices themselves were pretty chaotic, basically just one big rectangular space with a most depressing view out of the windows of next door's angled corrugated roof.

Everybody was looking for coders back then. Perhaps if I'd gone and worked for Activision directly things would have worked out a lot better than they did.

The lift doors opened out into a small reception area with Rod's office behind a reception desk and a small locked room tucked into the corner holding a treasure trove of printed T-Shirts, jackets, posters, promotional material and free games (none of which any of us were ever allowed to touch.) Beyond this was the main office area which was a long rectangular room with reception about halfway up the left hand side, above and below that running along the side walls were a small number of glass fronted conference rooms\offices and the windows-with-no-view taking up the right hand side wall. Toward the top of the rectangle were a row of grey filing cabinets, almost like a steel wall, that cut off a portion of the office which had it's own, unusual, purpose.

Most of the office space was comprised of cubicles, partitioned off individual work areas which housed the Activision staff and whatever coders happened to be in-house at the time. This was a practice Activision were quite keen on as it allowed them tight control over both the content and production of a game once it got to the closing stages (as well as holding the coders hostage until they had finished the game!) Anyone in need or working overnight could avail themselves of the 'sleeping area', which turned out to be the floor behind those grey filing cabinets. Those in the know brought a sleeping bag to help ease the pain of sleeping on a hard floor but everyone learned to make sure their heads were positioned well beneath one of the desks for added sound reduction. It was quite common to see a prone body or two behind there during the daytime trying to sleep amidst the noise of a full blown software house!

When we arrived that day the office was its usual chaotic self and we were now really anxious to find out what this Hello was really all about and after waiting a few minutes Rod came out of his office and spoke to us. "*Hello*" he said.....and that was about it! After a few minutes of small talk he left to deal with another matter leaving the three of us standing there wondering why he hadn't just picked up the phone and called us! To say the rest of the day was an anti-climax is perhaps understating things.

Around this time the three of us took delivery of proper PC based development systems from Catalyst. Each of us received a then state-of-the-art Opus 80286 PC with monochrome monitor running DOS and a professional cross-development package.

The business end of the PC development was handled by a PDS (Programmer's Development System) board, which was really just a parallel I/O card that would connect to the target machine (at the time a Spectrum, Amstrad or C64) via a simple interface. On the target machine you'd load a small piece of code that would sit there polling the lines and waiting for a signal supplied by a custom assembler running on the PC - as soon as you'd assembled on the PC and used the SEND Command the target machine would transfer the object code and you'd be up and running in about a second. The download code for the target machines came in three versions,

Nobody bought a Multiface 1 add on from Romantic Robot for the joystick interface and I'm guessing very few bought it for the 8K of internal RAM it came with. No, it was usually bought for one unique feature, "100% reliable saving of anything, anytime and onto all types of peripherals" as the advertising promised which of course translated to 'copy any game' to everyone else. Connect a Multiface 1 to your Spectrum, load in a game, press the Red Button, put a blank tape in the cassette recorder and minutes later you had a copy all ready for trading in the school playground the next day. It dumped everything.....RAM, Z80 Register values, System Variables, Main Screen display, the lot, and it even came with a switch that made the hardware totally invisible to anything. Nobody ever did, but if you could come up with some sort of hardware check for the device - a kind of *IF Multiface 1 detected THEN reset Machine* - you'd have to make sure it ran in every single part of your code and couldn't be removed by a determined hacker. What you ended up with was a program on cassette that when loaded put the computer back to EXACTLY the same state it was when you pressed that Red Button and you couldn't beat it. But one man could. The only game I've ever seen that could beat a Multiface 1 was written by one of the most amazing coders I've ever met and also one of the nicest people around, Rob Hylands.

As I mentioned earlier, Rob was helping Karl with the ST version while taking time out from his day job which was working at the submarine museum in Gosport. Rob had first met up with Karl when he advertised for coders and this tall smiling cross between Shaggy from Scooby Doo and a Teddy Boy with a heavy South of England accent walked through the door. Rob is the perfect example of how looks can be deceiving as, at face value, he appears to be a bloke who should be selling sherbet lemons at some old fashioned sweet shop, he's just so....amiable. But I don't think I've ever come across anyone as natural and gifted as Rob when it came to writing code coupled with an ability to think around corners and come up with ways of doing things that nobody else had spotted.

Rob has a quirky sense of humour that mirrors my own at times so we got on really well together, it didn't hurt that we both shared the same first name either and were roughly the same age. At a time when most games coders were in their teens or early twenties I was approaching my 30th birthday and it was good to talk to someone who at least knew the names of bands from the 60's and 70's. Rob has a way of seeing through problems and coming up with a solution seemingly with no effort whatsoever. We were talking about a demo he had written that had several large balls moving around a screen, the balls interacting perfectly and bouncing off each other with pixel-point accuracy. I assumed he was using some complex algebraic maths for the collision detection - remember this was in an age when everything came in straight lines and boxes - but the solution was so much simpler and intuitive, he explained that all you had to do was draw an imaginary line between the middles of two balls and if the length equals the diameter of a ball then you know they are touching. Obvious, at least to him.

It was the same when he told me of his way to defeat the Multiface 1, it was so obvious but so impractical that you wouldn't have even given it a second thought had it crossed your mind but Rob had, and he had the ability and sheer nerve to pull it off. Rob put his protection system into play in the only Spectrum game he wrote, *Super Wonder Boy (in Monsterland)*, a multi-load version of the SEGA sequel to their 1986 *Wonder Boy* game released by Activision in 1989.

To see if the method still worked I recently go hold of a copy of the game to run on a PC emulator, set the emulator up to mimic the presence of a Multiface 1, loaded the game and pressed the key to invoke the Multiface. Doing nothing else I exited the Multiface and returned to the game where, just as it had with the cassette original in 1989, the game tore itself apart for several seconds and then completely crashed. The brilliance of Rob's method was that even if I had saved off the game to cassette it was already tainted and loading it back in would have led to the same result, a totally unplayable game. I don't think many people are even aware of the protection on the game since the multi-load nature of it meant it wasn't exactly a game you could easily use a Multiface 1 to copy anyway. I like to think Rob put it in there just because he could and I know if he had used it on a single-load A-List title then it would have caused a lot of headaches to the casual pirates of the time. No, I'm not going to tell you what the method was, since if you know nothing about Spectrum Z80 coding it won't mean a thing to you and if you do then think of it as a puzzle to try and solve yourself.

Rob rightly went on to bigger and better things in the software industry but it was lucky for me he was there as he would go on to play an important role in getting the Spectrum version of *R-Type* finished and would make a vital contribution to the game that was beyond my ability at the time.

'dumb', 'smart' and 'interrupt driven' with the latter running under interrupts allowing the PC to monitor, control and even change the code on the target machine while the game was running, which wasn't really a lot of use if you were trying to write time critical code but it did have a nice line in real-time Trace functions.

Most coders created their own custom download code based on the 'dumb' version, the good thing being that you could send code to anywhere on the target machine which, in the Spectrum's case, meant you could use the whole of the BASIC RAM allocation without worrying about corrupting your downloader. I think I had my download code load into the screen RAM so I could use every last byte of memory for the game itself.

The PDS system was originally developed by Andrew Glaister, Argonaut's Jez San and Fouad (Foo) Katan of Bits to help create the game *Skyline Attack* but ended up becoming a business in it's own right, P.D. Systems Ltd. The going rate for a PDS system was £500 and the cost of the PC was £1200 so Wainwright was definitely spending some serious money on us which was probably another factor in why our pay seemed to be taking longer to reach us with each passing month.

A trip to Activision was usually on the cards when one of us had finished a level so we could give it to them to play-test but they soon decided they wanted to see us once a week. I think they were worried about the progress of the C64 version more than anything, but this became a major pain as we had to get all our versions into a workable demo regardless of what we were working on at the time, so we offered them an alternative. Instead of us having to stop development in mid flow to put a demo together and waste half a day driving to Southampton and back why not come to us instead? Our Producer at Activision for *R-Type* was Saul Marchese and we quickly conned\convinced him that it would be a great time-saver if he came to us. It meant we could show him all sorts of work-in-progress just by assembling and running the code in front of him there and then.

A Producer's job is pretty thankless, they're a kind of middle man between the coder and the software house and have to appear to be supportive of both sides. From the coder's perspective a Producer will get you what you need in order to write the game, pass along your concerns, be a good listener, agree that you're working far too hard for a company that's really a sweatshop and a hundred and one other things to show that he’s really on your side.

Of course since a Producer is usually employed by the software house you're developing the game for he's really only got one thing on his mind - making sure you meet that final deadline - and he'll do whatever it takes to achieve that since he'll have his bosses breathing down his neck if he doesn’t. As I said it's a pretty thankless task as the Producer will get grief from both sides even though they are trying to achieve the same outcome, though for some reason this often results in the Producer being one of the quietest and calmest

people around and thankfully Saul was just like that. If you shout at coders under stress you just end up with more stress which doesn't solve anything so perhaps it's something they teach you at Producer School.

But we had a few tricks up our sleeves as well and what we hadn't told Saul was that by coming to see us we had total control over not just what was shown but how it was shown. There's an old game developer's excuse that goes…..

Q:"Why is there only one sprite on the screen?"
A:"There's actually ten sprites but as each one shares the same X and Y coordinate it only APPEARS that there's one on the screen"

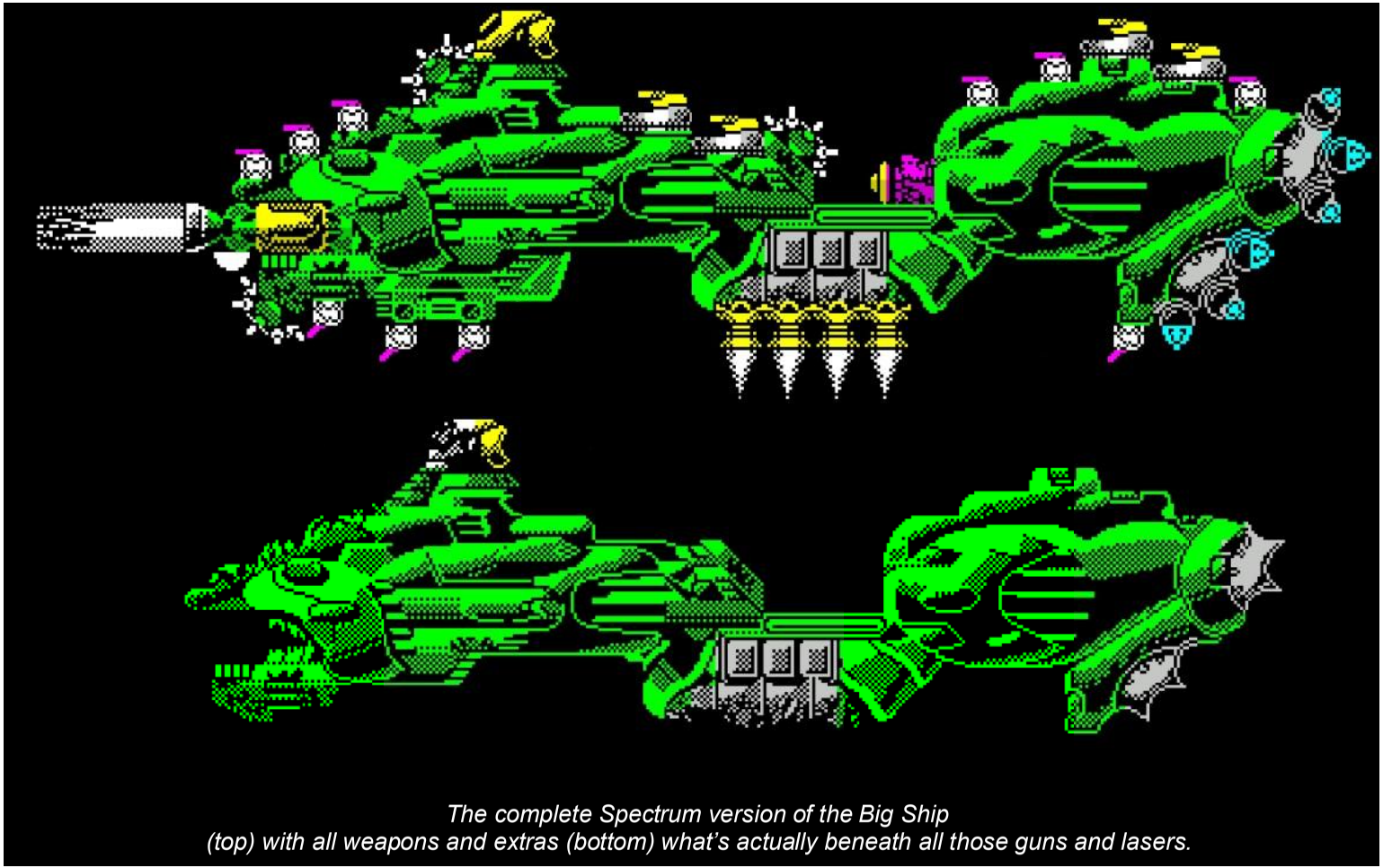
..... and I think every time Saul visited we had a new one to spring on him. I know Dave used "*well the sprite multiplexer is set up for sixteen sprites and the reason there's only eight on the screen is because I haven't set the display values for the others yet*" and I definitely presented part of a level that I knew didn't have certain weapon pickups because I hadn't written them yet but told Saul I had. But the best idea we came up with was the lighting.

Look at a TV screen displaying a picture on a bright sunny day with the light hitting it from all angles and what you have is one washed out piece of crap. You can display the most colourful, crisp, high resolution image going and it still looks like you've smeared the screen with Vaseline and turned the brightness down to barely enough. When Saul came to see us he walked into a near dark room with the curtains closed, TV screens that had the contrast ramped up to maximum and the brightness and colour up fairly high as well. We could have shown him any old rubbish (and we did) and still it looked fantastic, especially on the Spectrum. After a few minutes you'd forget you were looking at a half finished piece of code and think you'd just seen the most vibrant, glowing, colourful, nearly-there game imaginable! Hehe.

Activision may have been keeping track of us but Catalyst were holding their distance, and I say "*their*" because by now Wainwright seemed to have acquired a business partner of sorts. I don't think anyone ever found out where Sed came from, and you'll have to excuse my spelling since I never saw his name written down, but wherever Wainwright went it seemed that Sedghat Hosseini went with him. Just who Sed was and what he did was a big mystery and of course all sorts of rumours soon sprang up but the general consensus amongst the coders and artists was that he was either injecting money into Catalyst to keep things running smoothly or he had the capability to do so.

Whether the 'Doctor' title that Sed like to use was real or not no one knew, and if it was real then just what he was a Doctor of I never found out, and that also went for what he did, with rumours ranging from him being the Southampton version of a Venture Capitalist to a small shop owner.

What was obvious though was that just like Kightley Sed knew hardly



The complete Spectrum version of the Big Ship
(top) with all weapons and extras (bottom) what's actually beneath all those guns and lasers.

anything about computer games. Speaking of Kightley he was still in Wales but he did like to phone up every few weeks to make sure 'his' game was running OK and to point out that it was he and not Wainwright who was in charge. Unfortunately for Kightley when he was in this Boss Mode he liked to show just how in charge he was by talking about dealings with Activision and Catalyst that he would have been better off keeping silent about. I usually just agreed with everything he said to get him off the phone as quickly as possible but there were a few things he mentioned that I mentally filed away for possible use, just in case.

With Level Two finished there was no way to forestall the inevitable, it was time to try and crack Level Three. I think Activision were worried about all of us being able to pull this off but since I was the first one to approach it I seemed to get most of the attention, which wasn't exactly overt but every time Saul visited or I went to Activision a question along the lines of "*how are you planning to do the Big Ship*" always seemed to pop up. As I said before, Level Three is basically the Big Ship and nothing else - but the damn thing is so large that just moving it around seemingly becomes the biggest problem, on the Spectrum. It finally came in somewhere around 392x120 pixels in size which puts it close to being almost two complete game screens in length.

The funny thing was that what I was worried about and what Activision were worried about were two different things as I knew I didn't have a problem moving the Ship around, the backward character-based graphic engine I'd written meant that everything balanced itself out in the end. Think of it this way....with a big empty screen like you get at the start of some of the levels there is actually more room for bullets, weapons, aliens, scrolling stars etc. to be in play each frame and putting all them down takes game time. With the Big Ship taking up most of the screen there was no need to put down the majority of my stars, bullets and weapons didn't have that much room for movement so they got destroyed quickly and most of the sprites on screen when the Big Ship is around are the guns on the Ship itself.

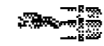
If you add to that the fact that the Big Ship doesn't really move that much - it slowly inches itself on to the screen and then tries to crush you - meant I only needed to update the Ship's position every second or so. In fact most of Level Three actually runs faster than some of the later alien-packed levels! No, my problem wasn't in moving the ship it was actually finding the space to get TWO of them to fit into what little memory I had.

Looking back now with more experience I can honestly say "*what the hell was I thinking*" as my original plan was to have two ships, one pristine and untouched and the other distressed and damaged. As the ship got fired upon and the gun sprites got destroyed I planned to chop out the clean graphics and replace them with little blocks of damaged ship to give that shot-to-pieces look. By the time the player got to the end of the level most of the ship had suffered damage

of some sort or another so it made sense to have two complete versions of each state making the replacing all that easier to work out. Mark even went so far as to supply me with complete graphics of the ships in both states.

Luckily seeing the beat up version of the ship brought me to my senses and all I did was use that one and made sure to put the guns and other ship sprites, weapons, exhausts etc. over the pre-damaged parts. Then, when the Player blew up one of them, the sprite disappeared in a cloud of pixels and the ship got damaged with no extra help from me.

The clean version of the ship did make an appearance though in a special demo I prepared for Saul, which was basically me just showing off. When he came over to the office one week I showed him this lovely, clean, Big Ship scrolling quite quickly across the screen and as it filled the screen suddenly another Big Ship appeared at the top of the screen and moved down diagonally under it while at the same time a third Big Ship appeared at the bottom of the screen and moved up diagonally over the other two. Again it was the fact that since the screen was mostly filled with Big Ship characters it didn't really matter which one they came from since it was only ever the topmost Ship's characters that would get put down, the others would just get skipped over and ignored. After seeing that Activision seemed to ease up on me a bit as they stopped asking me how I was going to do things and changed tack to "*when will the game be finished?*"



To my mind the hardest parts were over so now it was just a long slog to get everything finished. That's not to say everything was going to be easy as each of the levels had unique elements that would have to be addressed and the rapidly reducing RAM meant constantly having to go back over already written routines, shrinking and optimising them again and again. Unfortunately the code was now starting to get more and more tangled as part of one routine now doubled as a subroutine for another and I was making more use of self-modifying code in order to shave both bytes and time off routines. Routines would have multiple entry points, data that started out as bytes would get packed into nibbles (four bits), I would go over the graphics for everything again and again shaving off pixels so that the compression system would save a few bytes here and there and if all else failed seeing what symmetrical sprites could be chopped in half and then drawn as a mirror image.

I think pretty much everything that moves in the game is animated in some way even if it's just alternately drawing an existing sprite flipped or mirrored or by adding a bit of colour cycling. These techniques cost very little in time and memory overheads but help sell the idea that a whole lot is actually going on and that the conversion is close to the arcade original - where lack of space constraints means everything can be animated for real. Luckily Mark knew all the tricks when it came to minimizing sprite sizes and saving

memory so I didn't have to do that much extra work but sometimes I ended up changing things only for him to protest that I was messing up his graphics which, to be honest I was, but in my defence I didn't have that much of a choice.

I also had to leave a chunk of RAM free to be used later for the sound and music routines, something that fortunately was going to be delivered to me by Catalyst via Activision. How Wainwright was going to work this I don't know, I guess the plan was for Activision to sub-contract the sound to a professional music coder and then deduct that amount from what they were paying Wainwright. All I knew was that sound and music would be turning up shortly so I better have the room ready to slot them in as well as have enough time in the interrupt routines to handle them without messing up any of the other time-critical stuff.

Apart from coding the Levels there was plenty of other work to be done in putting the game together, the usual stuff of adding a High Score table, writing an Attract mode (which got dropped due to lack of space,) sorting out the multi-load tape routines and all of the front end - these usually tended to get done when I was getting fed up with a level so they sort of dribbled their way slowly into the game. My knowledge of joystick and keyboard handling had improved since *Rampage* where, thanks to the generosity of fellow coder Mev Dinc, my original buggy Kempston joystick routine had been replaced with one that actually worked. Another coder who was more than generous with his time and knowledge was Mike Archer, a Portsmouth-based Catalyst employee who I got to know when he, Wainwright and Dave Jolliff were finishing off the C64 version of Rampage in-house at Activision. Mike and I became friends and in later years I'd often have the use of his Mum and Dad's sofa to sleep on when I paid a visit to Portsmouth.

It was while I was finishing off Level Three I got a call from Mike swearing me to secrecy and asking if I was interested in a trip up to London with him to see a literary\computer agent and possibly signing on as clients. When Mike mentioned the agent's name and the company it rang some bells, I knew I'd seen the names Jacqui Lyons and Marjacq before somewhere but couldn't quite place them (turns out it was in a book called *Beyond The Arcade* by Nicholas Palmer) and I knew a few details about agents and agencies through some early Business Studies work I'd done so I wasn't exactly going into things with my eyes closed. After an introductory phone call to Jacqui and a quick run down of who I was and what I was doing Mike and I paid a visit to London without letting anyone else know and tried to track down the offices of Marjacq Micro Ltd. which was easier said than done since both of us had neglected to bring an *A to Z* and phoning up for directions just seemed to confuse things further.

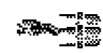
In one of those weird coincidences that seemed to keep popping up we walked into a large nondescript office block, just one of many we'd passed, and asked the receptionist if she knew where a

company called Marjacq Micro was. Not only did she know where it was but she told us how she knew, she'd once worked for Jacqui Lyons! Put on the right track and not believing our luck we had our meeting with Jacqui and she told us her terms and what she could do for us and what was expected of us in return - which could be summed up pretty neatly as ‘keep doing what you're doing but get paid a lot more for it.’ She did take pains to point out though that as I was still contracted to work for Designmaker\ Catalyst she couldn't represent me until *R-Type* was finished.

I hadn’t signed a contract or any kind of agreement for *R-Type* but I had when starting at Designmaker so technically I'd have to resign after finishing the game and then put myself in Jacqui Lyons hands, which was a huge leap of faith as it meant handing back all the Development kit to Catalyst and adhering to god-knows what other confidentiality clauses I'd agreed to. Since I wouldn't have to make a decision between Catalyst and Jacqui for a few months she was a handy insurance policy to have if things went wrong and all I had to do in the meantime was keep tight-lipped about the situation.

While I'd been taking it easy(!) Activision had been stoking the publicity fires for the game and had sent off demos of the first two levels to the magazines resulting in some nice previews and lots of great colour shots of the game. EMAP, publisher of Sinclair User for the Spectrum, decided to send someone down to Fareham to take some publicity shots and get some more information about the conversions so we all dutifully trotted out to the garden at the back of the office and had our pictures taken messing around on and under the trees there - one of which ended up in the magazine. Unfortunately the write-up that went along with it is complete nonsense and a lot of it just plainly made up e.g. “*I used to own a BBC computer*” (no I did not), “*most of the aliens use up to eight frames of animation*” (I wish) and my favourite "*From their base in Swansea [the programmers] regularly commuted to London in order to play the coin-op*" (I guess no one was listening to what I was actually saying.)

This was not to be the last time that magazines would print plain untruths about games that I was either working on or connected with, probably best summed up by one ‘review’ of Karl’s ST version of *R-Type* that gave it six out of ten for Sound. The only problem was that as I was reading this in the office Karl was still working on the totally sound-free and music-less game code and didn't expect to have a beep out of the machine for at least a couple of weeks! He did have to endure a few digs from me though as I kept asking him to turn the volume up so I could hear it properly and suggested he got a new ST, since as there was no sound coming out of the speakers it was obviously broken!



I skipped over Level Four and started work on Level Five, an idea I had to alternate between coding what I considered easy and hard levels. Since Level Five had large snakes as

One Saturday when I'd travelled up to London to take in a Computer Fair, usually at Ally Pally or the Royal Horticultural Hall, I picked up a Sixword Swiftdisc to replace the shaky microdrive system I was using on the Spectrum end. The Swiftdisc was basically a 3.5 inch floppy drive, Shugart standard I believe, which connected to the Spectrum via a very familiar looking Sinclair Interface 1 box but which boasted microdrive emulation (which I never used) and a rather nice press-button Multiface-type capability which I found ideal for quick loading the PDS download code and perfect for messing around with the blocks of code I was loading and saving all the time. It was great for demonstrating work in progress to Saul as well as I could pre-prepare a number of different things, dump them all to a floppy disk and pick and choose what I wanted him to see and have it up and running in seconds. It beat the hell out of microdrives, cut a lot of time out of the write-assemble-download cycle and I was a lot happier trusting my work to a reliable industry-standard floppy disk rather than Clive Sinclair's hybrid piece of stringy tape!

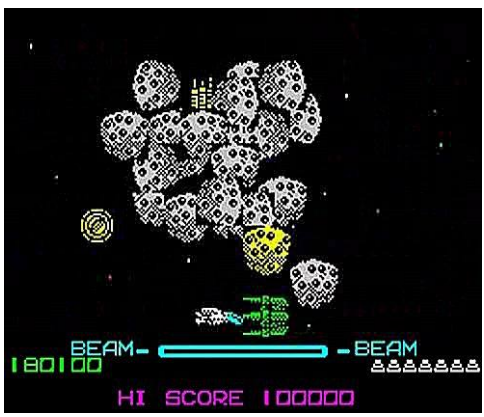
Since I was using the Swiftdisc so much I had the idea that I better get hold of another one as a backup just in case the one I was using failed, an idea brought about when the Power Supply of the Opus PC I was using packed in and I had to spend a couple of code-less days waiting for a replacement. So I called Sixword up and told them where I was and they said they could get one to me no problem, in fact I could even go and pick one up that night if I wanted since.....weird coincidence time again.....the registered address of Sixword Ltd. was a solicitor's office in Fareham!

per Level Three and a ridiculously simple to beat End of Level Boss I thought I'd start off with an easy one. I'm guessing that the original developers must have been thinking along the lines of "*well they've made it halfway so let's give the player a rest*" by including what has to be the blandest and least interesting of all the levels in the game and topping it off with an End of Level Boss that seems to be a single sprite covered in frog spawn! Not that I was complaining at the time because it meant I had a bit of an easy ride for a few days, but it does mean that there's very little else to say about it.

I do have a couple of regrets about the level though, it's a pity I couldn't implement the parallax scrolling of the arcade original as the fabulous background graphics here are it's one saving grace and I do regret not making the background safe to move through. In the arcade original you can fly behind the plant-like graphics at the top and the bottom of the screen which gives you a bit more room to manoeuvre but means you can't quite see what you're doing or what's heading toward you. To do the same I would have had to rewrite every routine that dealt with the background putting in exceptions just for this one level and at the time it didn't feel worth the effort. I know some gamers picked up on this and if I had the chance again I'd go back and change it but I think it's just a little bit too late to do that now.

Having Activision come over every week or so to see what we were doing was turning out to be a bit of a waste of time for Saul so he decided to delegate this task which is how Nick Dawson a.k.a. 'Badger', an Assistant Producer at the time, got the thankless task of having to visit us instead. Assistant Producer was really just a fancy name for trainee as Nick was still learning the ropes but we all got on well with him and he was a really nice guy. I think we did trade on his naivety a little at first as one of his jobs was to keep everything running to schedule, which was made all the more difficult by none of us actually having made any sort of written plan involving dates, targets or goals.

Later games would be Project Managed and Gantt Charted down to the individual day but back then we



What hideous alien nasty waits to be uncovered at the end of Level Five?

Watch it, you could have somebody's eye out with those spikes!

were just winging it and making it up as we went along. Just so Nick would have something to show Saul and Activision I put together a very simple schedule for him which had me finishing the game mid August, which sounded reasonable but was really just a wild stab in the dark to keep everybody happy.

It's around about this time that something happened which, on the face of it, doesn't seem that much but had a major impact on the way I saw myself as a person. This is going to be a bit on the long side so forgive me. Dave and Karl were invited\ ordered to Activision one day to talk about how things were going for them and I decided to tag along for the ride. While the two boys were taken out to lunch and grilled about how their games were progressing I was left in the office on my own to wait for their return. I was just sitting there killing time when this guy wearing a long black leather coat with matching leather fedora hat and an entourage of five or six other people swept in through the door into the office, all talking loudly and behaving like they owned the place.

There's a saying that first impressions count and my first impression was "*what a dick-head!*" The guy was just so full of himself and the people with him were just… well… fawning all over him and there was this whole atmosphere of 'LOOK AT US WE HAVE ARRIVED' that I had to ask someone who it was. It turned out the Great Man was none other than adventure author Fergus McNeill, presumably there to talk about *Mindfighter*, a text adventure he was 'programming' for Activision - if using a modified version of Gilsoft's *Quill* follow up program *P.A.W.* could be classed as programming that is. Thanks to my fiddling about with *The Quill* and *The Enhancer* I knew just how easy it was to customize and add specialised routines and functions to code like this so if I'm sounding less than excited about seeing a man famous for using somebody else's Adventure Writing System behave like he was God's Greatest Programmer then it's not without reason.

I guess Activision decided that being so important deserved a free lunch as well because it wasn't long before Fergus, his entourage and a large number of staff left the office amidst much loud laughing and shouting....and things went quiet. Very quiet. I realized that apart from another chap sitting on his own the office was totally deserted, no Activision staff or anybody, just us two. There's not much you can do in a situation like that other than go through the polite "*Hi*" and "*did you just see all that*" type stuff and after a few minutes the door buzzer\intercom went, so with no one else there I answered it. Someone was outside trying to get in but the door was locked so I went down in the lift to open it...but you couldn't unlock the door from the inside either without a key(!) which meant that me and this other man were trapped inside the building and no one could get in until someone from Activision came back from their long lunch!

I went back up to the top floor and told this other guy - Nick - that we were locked in so we just chatted

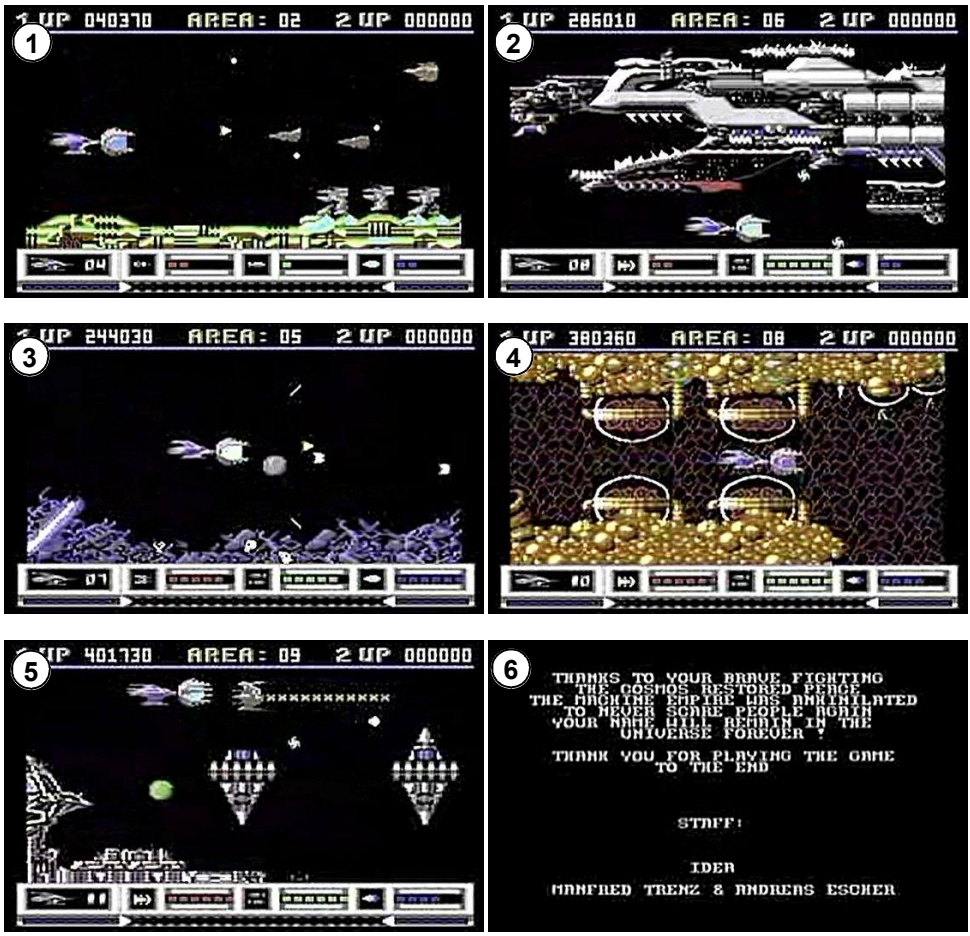
about games and stuff for a while. Nick shared a Coconut Snowball with me and was a real nice friendly person, saying he'd written some games in the past while I was probably a bit of bore going on about *R-Type* all the time and how great it was because everybody said so. It wasn't long before some people came back from lunch and opened the doors, whereupon Nick said his goodbyes and quietly left. I asked one of the staff who the chap I'd been talking to was - *"that's Orlando"* he said. Bloody Hell...I'd just been chatting to Orlando....twittering on and on about how great my little Spectrum game was!

If you don't know who Orlando\Nick Pelling is then I suggest you look him up online, it may not seem much now but at that early stage of my coding career Nick was really the first major star-name of the industry I'd met (along with Fergus that is) and if I'd only known more about Mev, Zari, Chris Wood and the other coders I'd met I might not have been so starry-eyed about it. But that's how it goes. The way both of those industry names had behaved themselves got me thinking about myself, about how I was starting to get a bit big-headed about how 'great' my game was and the things being said about it. Who was I to argue with people who knew more about the business than I did? It was time to look at things objectively, I was writing a game for people to play on what was basically a toy computer and nothing more.

I'm not going to beat you over the head with my epiphany but I did come across a quote that seemed to sum it all up at the time - *"you're in trouble when you start believing you're as good as people tell you you are"*. Since that day with Fergus and Nick I've looked on the games I write as nothing more than the sum of the code and if other people enjoy playing them then that's a bonus.

Later that afternoon Saul loaded up a demo of a new game on the Commodore 64 from German company Rainbow Arts for us to have a look at. Rainbow Arts had been in trouble they year before with Nintendo for releasing *The Great Giana Sisters* on home computers - a game so similar to *Super Mario Brothers* on the NES that at times it felt like you were playing a direct port - and had to yank it off the shelves almost as soon as it went on sale under threats of legal action by the big N. Now it seemed they were going to have to do so again as what we saw that afternoon may not have been an out and out identical copy of *R-Type* but it certainly had all the elements that made *R-Type* what it was. The game was called *Katakis.*

A definition of the copyright-testing phrase 'look and feel' goes something like this: *'a term used to describe the main features of the appearance and the experience a person has using a product'* and *Katakis* certainly had the 'feel' part of it down 100% being a continuous side-scrolling shooter set in a bio-mechanical world. You could pick up add-ons to give your ship extra weapons, control a detachable orb you could use as a weapon, hold down the fire button to build up an



Screens from Katakis: (1) Red Spinners come at the player accompanied by Missile Walkers. (2) A Big Ship poses a threat. (3) Armed with a rotating Orb and firing off lasers and missiles a Pick Up is within reach. (4) Brains in Glass Cases obstruct the way. (5) An alien leaves a trail of small round objects behind it. (6) A very familiar looking End Game message.

extra powerful shot and battle your way through hordes of different aliens before meeting a large End of Level Boss. We only got to see the first level that afternoon and what we didn't appreciate at the time was just how much of *R-Type* had seemingly been appropriated for *Katakis*, it's only now through the use of emulators that you can see the overall picture.

It's true that Level One of *Katakis* looks more like the arcade game *Nemesis* than *R-Type* but once you hit Level Two then a feeling of familiarity starts to creep in and you begin to spot the layout similarities. You soon find yourself flying past a huge ship that takes up several screens, a level where aliens leave a trail of small orbs behind them you have to destroy and the appearance of some alien sprites that wouldn't surprise fans of *R-Type* in the least. For the final 'homage' when you

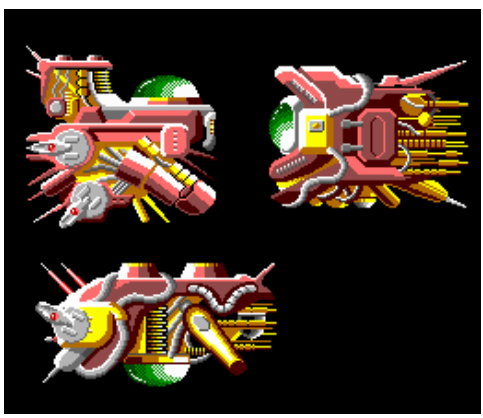
complete the game you are greeted with a scrolling Congratulations message that, barring one word, is EXACTLY the same as the one you get when you finish *R-Type*, down to the weird 'JapEnglish' use of phrasing and grammar.

It's no wonder Activision were a bit put out after paying money to IREM to licence their game only to see this looky-likey go on sale. They decided to have words with Rainbow Arts who, under threat of legal action, pulled the game from European shelves and promised to halt distribution of the game.

With the easy Level Five out of the way it was time to get one of the harder Levels done and Level Four certainly qualified on that count, the main problem being the Fish Eggs. Immediately on starting Level Four the player is faced with a small red alien that flies around the screen laying a trail of green globes behind it, or Fish Eggs as Mark and I called them, which then become part of the background and scroll along with the rest of the graphics a pixel at a time. It's not long before the screen is full of Eggs as more and more Egg Layers appear, to be joined by even more Eggs that scroll on as part of the regular background graphics, but unfortunately they don't act like regular background graphics as they can be destroyed just like sprites.

To further complicate things only some weapons can destroy the Eggs, the Bounce Laser has no effect on them and when it hits one it's just like hitting a regular solid piece of the background. And if that wasn't enough you have Egg Eaters, more red aliens that fly around removing any eggs they pass over. Before this my code was already looking a mess and now I had to go back into pretty much every major routine and set up all kinds of exception events just to handle this one damn level!

It wasn't feasible to make every Egg a sprite, there could be up to two hundred on screen at any one time and the game would have slowed to a crawl, so I ended up amending the code I was using to handle the weapon Power Up system which at



Original ST graphics for the End of Level Four Boss….

…..and the same for the Spectrum. An incredibly faithful reproduction by Mark.

least had the benefit of being already written. This code inserted the Power-Ups that appeared when you destroyed a Podder into the background itself so that they scrolled on pixels but were treated as sprites by the collision detection system, a necessity since treating them as normal sprites would have meant the Power-Ups racing across the screen in eight pixel increments. I still had to make sure that the Eggs appeared on character boundaries - I couldn't just put one down where and when I wanted - which was OK when there was nothing next to it but empty space but got complicated real quick if there was an Egg already there. And just to make things worse there was the background scroll to allow for.

Eventually I came up with a neat little system of paired Eggs pre-shifted by one, two, three etc. pixels and with partial graphics that would overwrite any adjacent Eggs. I checked to see where the Background scroll was with regard to Character boundaries (a number from zero to seven), checked to see if there was an Egg to the right of where I wanted to go, picked the correct pre-shifted character pair if there was an Egg or a single pre-shifted Egg if not and punched it into the Background. And away it would scroll. To remove an Egg I just did everything in reverse with a different set of pre-shifts, this time with a space instead of an Egg, and that would clear it off the screen. It worked really well and was glitch free - it's just a pity that this routine never saw the light of day.

It was July when it all went wrong. For me, spending most of my time in front of the computer in Fareham meant I wasn't going through a lot of money but for Karl (and to some extent Dave) the regular monthly money transfer from Catalyst was what was keeping them and their businesses going. We had got used to our monthly pay turning up late but as we got further into July with nothing appearing in our bank accounts we started to get a bit worried. Then Wainwright dropped the bombshell and told us that there was going to be no more money and we weren't going to get paid anymore!

From what we were told at the time by Wainwright and by talking to the other coders at Catalyst we managed to get some idea of what had been going on, which could be summed up by that old English metaphor "*robbing Peter to pay Paul*." At some point in the past it looked like Wainwright had used the money from a future project to help pay off a current one and things had gone quickly downhill from there. As soon as he signed a contract for a game and got an advance payment the money apparently would go to pay off outstanding bills and fund the coders working on current games instead.

Now to be fair to Wainwright I will say that all the money that was coming into Catalyst was being used to keep the company going and he seemed to be the victim of rising costs and bad cash-flow management rather than over-extravagant spending, but those are not things you want to hear when being told that the kitty was empty and nothing else would be coming your way.

Why everything came as a surprise to us I don't know. The March 1988 edition of The Games Machine magazine carried an interview between Wainwright and the founder of Automata Software, Mel Croucher in which Wainwright admitted to ripping EVERYBODY off. Croucher had set out to write what was basically an exposé of how young games programmers were being exploited and taken advantage of (under the headline 'Industrial Child Abuse') and after hearing from several coders, including a young Jason Austin, he approached what he called "*a mature businessman*" for his side of the story.

Croucher and Wainwright had known each other for years, both were situated in the Portsmouth\ Southampton area and Wainwright had worked on some Automata game titles for Croucher. Perhaps the friendship between the two men fueled by Wainwright's lack of sleep was the cause of him saying more than he really should have but in a jaw-dropping admission that surprised even Croucher the truth came out....

DW *- ....After I went to Martech I started abusing schoolchildren for myself.*
MC *- I know you've not had any sleep for a few days, but you don't want me to put that into print do you?*
DW *- Why not, it's the truth. I think I've ripped off every programmer in Portsmouth, but I've changed now that my business has expanded. I'm ripping them off all over the country! I've got five offices nationwide, thirty programmers, and a tame lawyer. I learned early on that payments don't lead to good programs. Programmers have to learn from their own mistakes, it's the only way. To get anywhere in software you've got to get ripped off before you learn anything.*
MC *- That really is an incredible attitude, why are you telling me this?*
DW *- Because it's true. Look. I'm still being exploited today, by ******** it's the same with all the big companies. They don't give a toss about programmers, if we've got any food inside us, if our eyesight is short because of permanently staring into monitors, about anything. I've been on four hours sleep for the past few days now and they couldn't care less.*
MC *- OK. how do you get out of this situation?*
DW *- I'm going into publishing on my own. 'Wicked Software', something like that. Others produce crappy little games and tens of thousands of kids buy them.*
MC *- And you are not going to do that? You've learned from your experience?*
DW *- You've got me wrong. Mel. That's exactly what I'm going to do.*

So according to Wainwright's logic he was really doing us a favour by ripping us off, and now that we had been we should be thankful we had learnt something and could make our way in the world of software with our heads held high! You'll have to excuse me for not patting him on the back and thanking him for my education and you'd be right to question why I even bothered to keep working for the man after the interview got published.

If I sit and think about it today then I'd have to say that the urge to

see the game finished and released overrode all other considerations and until you've done it for yourself you'll have no idea just strong that urge can be. I don't believe it's unique to programming, I guess it's the same for authors and artists and can apply just as much to someone who makes a giant matchstick model of the Eiffel Tower as it would to a computer game. I would hesitate to say it takes on a life of its own but it does sometimes feel like you are dealing with Frankenstein's creature, carefully patching it together piece by piece and then forcing electricity through its veins to make it live. LIVE!

Alright, definitely a bit of an exaggeration there perhaps but not far from the truth, and if I really didn't want to finish the game then Wainwright's interview and this subsequent drying up of money would have given me an ideal out. Besides, I suppose it could be argued that running out of money was not exactly the same thing as ripping someone off, but it sure felt like that at the time.

None of us really wanted to stop coding but there was no way we could carry on working for free. I think Activision had an inkling of what was going on since Wainwright admitted to me that he had approached them asking for more money citing the quality of the Spectrum version and pointing out that it was going to make a lot of money for them so it was only fair he should get a bit extra. I don't know the reasons they gave but no more money would be forthcoming from either them or Wainwright so we were all stuck.

Anger soon took over from shock when it appeared that Wainwright expected us to keep on coding for nothing and complete the games so he could live up to his end of the contract and deliver a finished product to Activision and get his Delivery Payments (usually the largest block of the contracted price.)

It was Karl who realized that the answer was literally staring us in the face - the PCs and the PDS systems. You know that old adage, "*Give a man a fish and you feed him for a day. Teach a man to fish and you feed him for a lifetime*"? Well the PC and PDS kits we were using were our fishing rods and we realized that if we could come out of this with a full development kit each then we'd have everything we needed to set ourselves up for future projects.

Taking a rough figure of £400 a month salary, and since we were paid in arrears, I estimated that by the time I finished my version I would be owed £1600 which was about what the development kit cost anyway so maybe offering a straight deal - we finish coding the game and keep the development kit in lieu of pay - would be something Wainwright would go for. Dave and I agreed with Karl's thinking, which was easy really as it was that or nothing else. What I couldn't let on to both of them though was that I had another motive for going ahead with the deal, that this would be a chance to cut myself free from Wainwright and Catalyst and walk away clean with a development kit to go on and be represented by Jacqui Lyons at Marjacq.

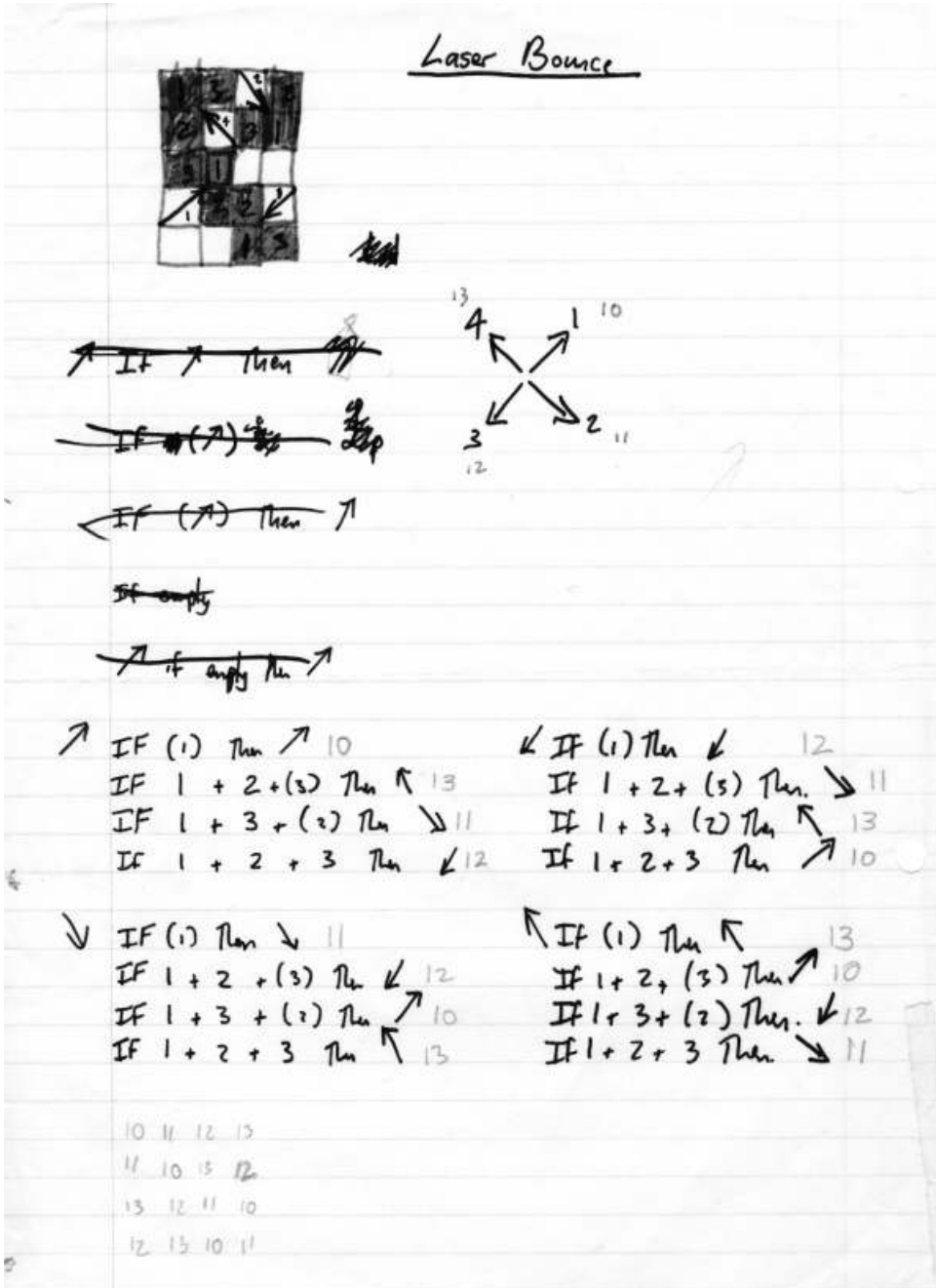
It may sound like the deck was stacked in our favour, do a deal with

us or the game doesn't get finished, but Wainwright could easily have called our bluff as he was well within his rights to ask for the existing source code and the return of the development kits and pass the whole lot along to one of his cheaper coders who would at least finish the game for him no matter how it turned out. Besides, after all the work we'd put into it none of us wanted to turn our backs on the game especially if it meant leaving without money owed to us or the PDS setup. No, somehow we had to get Wainwright to beg us to stay, let us keep the equipment and feel like it was his own idea all along. After the way he had worked us over when negotiating the money at the start of the project it was finally payback time.

Wainwright and Sed arranged to meet with us at the office in Fareham to talk over what was happening so this was the ideal time to put our plan into action. The setup was simple, Dave and Karl would be the voice of calm reason and I would be the one who lost it, shouting and swearing and threatening all kinds of things only to be regularly calmed down by the other two before going off on one again. Furthermore, at a prearranged signal, I would storm out of the meeting and they would run after me to try and make me see sense, talking me around to their point of view. With everything mapped out and after a little rehearsal we waited for the marks to arrive.

Wainwright was doing all the talking with Sed backing him up when needed, usually to point out things like we had a (verbal) contract to finish the game and that we would get paid....eventually. Unfortunately for the two of them Kightley's loose tongue over the phone now worked to our advantage as every time they raised what seemed to be a valid point I would shout it down with "*but Kightley told us.....!*" Wainwright kept insisting that Kightley really had nothing to do with the production of the game but my constant insistence that as I worked for him how was I to know what he was telling me hadn't come straight from the horse's mouth was the perfect stonewall answer. I played my part to the hilt, swearing, sulking, shouting and threatening with the other two boys calming me down and wondering perhaps if we couldn't work out some kind of deal. Wainwright wanted us all to finish the game and try and renegotiate with Activision once he had the final product in his hand but had no money to pay us with, so it was now time for the second half of the plan.

On cue I stormed out of the room trying to give the impression that it was all over and I was on my way back to Swansea while Karl and Dave raced after me begging me to reconsider. Sufficiently far away from Wainwright and Sed we all had a good laugh at what had just transpired, waited an appropriate time and returned to put the final touches to our little play. I was now the quiet one, sullen, seemingly annoyed and fed up with Karl and Dave. They told Wainwright that they had managed to talk me round to some kind of deal, that if he agreed to us all keeping the PCs and PDS systems we would finish the game and not ask for any money. I was all "*yeah...yeah… whatever*" and doing my best to look



It took a while to work out how the Bounce Lasers should behave, I had to write it all down so I'd remember it.

The Flame Weapon (officially the Ground Hugging Lasers) was a little more difficult to work out - but I got it in the end.

really pissed off with the boys and only going along with the deal to keep them happy. After the in-your-face performance I put on before the break Wainwright probably wanted to get out of there as quickly as possible so he took the bait and agreed to our proposal.

Later it became apparent just why he had so readily agreed, but for now we had gained everything we had set out to achieve and were feeling on top of the world. A few days after I had Wainwright sign a short agreement I had written so he couldn't back out of the deal when it was all over, it was amateur at best and I suppose if he wanted to he could have contested it but it made me feel a bit safer. I didn't know it at the time but that badly written piece of legalese was to get me out of some serious trouble months after the game was finished.

Deal done all I wanted to do now was finish the game and get out of there as quickly as possible, so things got a bit silly from then on in, I was now working about sixteen hours a day and heading off to bed around two or three in the morning. Looking at the three levels left to code I could see there was nothing in them that would require anything new or difficult to implement, it was basically just the boring rote of putting in one sprite routine after another day after day. It can be hard enough to motivate yourself when faced with this but what made it worse is that I'd just about reached the three-quarter point and game-fatigue was setting in.

In my experience there comes a time in coding any large game or program, usually around the three-quarters mark, where you've just had enough and it can be so easy to say to yourself "*well, that's most of it done, I'll go and do something else and finish it off later*", which of course you never do. The funny thing is it's very seldom down to being faced with some insurmountable hurdle or major problem it's just that you can only spend so much time looking at the same thing for months on end before you get tired\bored\demotivated or just plain hacked off.

It's a bit like reading the same book over and over for six months, that when you get to the last page you go right back to the beginning and start again. And again. And again. I think most coders get this at one time or another and I know of several projects that have never see the light of day because the programmer has lost the spark that set everything off, what was originally seen as challenging is now just long boring work. Then you see something or read something or come up with an idea that excites you, that you want to put into action NOW.....only there's that damn program you've been working on for all this time getting in the way, the one that still needs two or three months work on it, the one you can always finish off once you've put that exciting new idea to bed. I've done it a few times myself, for example there's my *Quill*'ed *Colossal Cave* from 1986 that I still need to finish, I'll get round to completing it one day.....maybe.

Fortunately for me(!) I was stuck in an office in Fareham, home was a couple of hundred miles away

and as I've said before it wasn't as if I had anything else to do. But now I had the incentive of walking away with my own PC and PDS kit if I could just get everything done and there wasn't anything left in the game that would require major work like the Level Three Big Ship or the Eggs of Level Four. What was required though was to code all the Aliens that appeared for another three levels (over two hundred in Level Seven alone) which was not something I wanted to dwell on but had to be done.

Despite what I just wrote about the boredom of coding Level Seven it does contain what I consider to be the best few seconds of the entire game when around halfway through the level almost fourty of what we called Yellow Butterflies stream from the top of the screen, do a ninety degree turn and head off across the display. In the arcade game proper this comes across as just another sequence of sprites but on the smaller more tightly packed display of the Spectrum it pretty much becomes one of those WTF moments.

Things start to get a bit ridiculous halfway through Level Seven.

It was this sequence that got Saul shouting out for people in the office to come and see what was happening when I took the finished level over to Activision to show him, and it's the perfect example of how you can get so involved in a game that you don't notice what else is happening which, in this case, was the game slowing down to an absolute crawl. There is so much movement on screen during this sequence that the Spectrum just runs out of steam and the time it takes to display one game frame moves away from 5/50ths of a second and starts heading towards 7/50ths or even 8/50ths. I used to joke with Dave and Karl that at certain points in my game it no longer ran at frames per second but at seconds per frame, a bit of an exaggeration but that's what it felt like to me.

Earlier I touched on the fact that the graphics in the game were compressed to save space. For the Spectrum version this was a necessity since there was only a maximum of 15K of data available for each level of the game and it just wasn't possible to fit every graphical element of the arcade version into such a small amount of memory. On the Spectrum the smaller sprite sizes and their lower resolution certainly helped but more was needed so Mark cut down the number of sprites in an animation cycle - how many individual sprites make up something like a flapping motion or a circular arc - to the bare minimum.

This meant that when it came to something like the Homing Missiles I ended up with eight frames of animation (the Missile facing Up,

Down, Left, Right, Diagonal Up/Left etc.) compared to the arcade version's sixteen directional frames. Since the Spectrum sprites were based around eight pixel character movement there really was no need for graphic smoothness on the lower resolution screen, and trying to draw intermediate sprites with subtle angles would have been a waste of time, but even with eight frames the graphics were still taking up too much memory.

If you can imagine an arrow pointing to the Left and you put a mirror next to it then the arrow in the reflection will now be pointing to the Right, it's exactly the same arrow it's just been visually flipped along the vertical axis of the mirror. This is something you can do in software quite easily - taking a sprite and drawing the pixels in it Right to Left instead of the normal Left to Right gives you a mirrored version of the original sprite - in effect giving you something for nothing.

But it doesn't end there as, with a bit of thinking, you can mirror that sprite along the horizontal instead of the vertical axis which gives you an upside down or flipped sprite. Then you can combine both methods to give you a mirrored and flipped sprite, you're just taking the original sprite and manipulating the data that makes it up in order to produce new orientations of it quickly and easily.

So now you don't need eight frames of animation for a Homing Missile, you only need three, since through software Left becomes Right, Up becomes Down and Up/Left becomes U/R, D/L and D/R - a saving of over 60 percent. Of course there are other things the Spectrum version uses to save space, larger sprites for example often have a common non-moving element which means you can save off that part as one sprite and then add the moving elements to it as separate sprites. This is how the Pistons in Level Six are coded. As it's only the end of the sprites that move there's no need to have the big static part of each sprite in the sequence taking up memory so that part of it is stored as one sprite and the moving parts added to it as required.

For *R-Type* compression was not a choice, it was the only way I could make sure everything I wanted to get into the game was going to make it, but there is a downside to this approach in that what is compressed needs to be decompressed and that takes time. The routines to flip and mirror the sprites, as well as the decompressing of the sprite data itself, all take longer to carry out than a normal sprite routine being fed uncompressed data and that can have a big effect on a game. I was lucky with *R-Type*, or I should say my take on it, since I went with the unconventional approach that how fast the game ran wasn't going to be my primary concern, I just wanted to put as much of the arcade version into it as I possibly could.

If you stuck with me through the earlier parts where I tried to cover the technical aspects behind the game then you'll have some idea of how important it is to make sure each game frame runs as quickly as possible - in graphic intensive arcade type games that is. For a while there was an almost obsessive search by Spectrum programmers to have their games run at 50fps, so they could

boast of ‘super smooth graphics’ and show how clever they were but when you're coding down to the wire like that, keeping track of every last instruction or clock cycle and timing things to the microsecond you risk the real fear of losing synchronisation and can end up destroying your game. Because of that, fast, silky-smooth running games tend to consist of just a handful of small sprites and a screen that doesn’t scroll, easy enough to do when you're coming up with an original game but totally unsuited for any kind of arcade conversion where the screen is moving around.

Very often a choice has to be made between what you'd like to see on screen and what you can get away with. Yes, you can have hundreds of sprites on display each frame, but you'll be able to go and make a cup of tea in the time it takes for you to move from one side of the screen to the other. Solution? Lower the number of sprites. Easy. It's not what you wanted but now it works. If you're not careful you can let this chase for perfection take over everything else.....screens and sprites get smaller so it takes less time to put them down, colour gets dropped in favour of monochrome, backgrounds flip or push-scroll in eight pixel character increments to give the illusion of movement.

I could have shaved a couple of characters off the edges of my screen display, made it more square, that would have saved some time. Or I could have halved the number of Aliens that appear, giving the player a taste of the game rather than the whole thing. Or scrolled the screen on character blocks. Or made the game completely monochrome. Or reduced the size and number of the player's weapons. I won't go on, you get the idea. All I knew is I wanted as much of the original game as I could to make it into the Spectrum and if that meant things slowed down to a crawl when the screen got busy then so be it. It was a price I was ready to pay and I really expected people to pick up on it when it was released but somehow the majority seemed not to notice it and I got away with it. A pity really as I had all my "*yeah I know it runs slow but look at all those sprites*" arguments ready to go.

As coding came to a close on Level Seven I went back and had a look at Level Four. Usually once I'd completed a level I left it alone but I wanted to reuse some of the sprite movement patterns that were in that level. Right away it was obvious there was something very wrong. Egg Layers were moving around the screen dropping Eggs and they were scrolling on as part of the background but.....if you shot at them or did anything that resulted in their destruction they just stayed there and weren't removed from the screen.

At first I thought it was something I'd changed in my main code, something to do with the collision detection routines or the weapon logic, but the more I checked the more I found everything was working OK. My next step was to check that the code that handled the removal of the Fish Eggs was working properly, or rather I would have if I could have found it - it just wasn't there any more!

The code that handled the removal of the Fish Eggs, all that carefully worked out pre-shifted pairing of characters, was gone and what was worse it wasn't in any of my backups either. At the time I was running a daily Grandfather\Father\ Son system of backing up with a similar weekly one that was stored well away from the others, a practice I'd learned from my days on mainframe computers. Unfortunately I had limited myself to how many weekly backups I could store so I only had two or three weeks worth which shouldn't have been a problem, after all who needs to go back three weeks to restore data? Somehow, and I have no idea how, the code to remove the Fish Eggs got lost and never made it into any of the recent backups and the one backup that possibly might have been of any use had been overwritten just a few days earlier.

I stopped coding Level Seven and went back into Level Four. I knew what I wanted to do and a good idea how to do it but the code that had come so easily to me the first time now didn't seem to want to work. Here's a piece of advice they don't teach you in University courses on writing games - don't code when angry - a state I was in trying to work out what the hell had happened and why I couldn't duplicate what seemed such a simple piece of code that I had written once already. I'm not talking kilobytes of program code either, looking at the source now it runs to just over a hundred lines of code, but for some reason it never came back to me. The patched up piece of coding that eventually made it into the game does pretty much what it's supposed to but it's not as smooth or crisp as the original was and the detection is just a little bit off here and there. Most people don't notice it, but I do.

Having to go back into Level Four cost me a few days and I wasn't in the best of moods when I returned to Level Seven to finish it off. Things didn't get any better when the fallout from Wainwright's lack of money started to hit. I had been waiting for the music and sound FX for the game to arrive for some time but I needed to know now just exactly how much memory they were going to take up as running out of RAM was turning into a daily occurrence. I got my answer all right when Nick Dawson casually dropped into conversation the fact that Activision wouldn't be supplying anything of the sort and that if Wainwright had told me that then he was totally wrong!

Whether Nick was correct or not I don't know, it may very well have been the case that Wainwright had never done a deal with them to supply the sound and music and had been stringing us along, and I do mean us, it wasn't just the Spectrum version that was going to remain silent. It could also have been the case that someone at Activision decided not to give any more money to Wainwright on principle, perhaps they thought that he'd have to find the money from somewhere so that he could deliver what he'd contractually agreed to.

While Activision weren't in the graphic supply business they did have a sort-of audio department, which amounted to a big fancy electronic

keyboard on a desk that I assumed was used to produce in-game sound and music, but getting a tune out of that was not the same as writing eight bit sound drivers and interrupt driven code - a fairly specialised and complicated aspect of the programmers art. But whatever the reason the outcome was the same, Activision would not be supplying any audio for the game and since Catalyst didn't have the money to pay for someone to do the job then it looked like *R-Type* was going to be one of the quietest games around.

The three of us tried to work out what we were going to do - and since I was looking to finish the game first I was the one most in need of an answer. The best I could come up with was reusing my *Rampage* sound routines and somehow making them multi-channel, but just how I'd go about mixing them together so they came out recognizable as *R-Type* I had no idea. Luckily the answer was closer to hand than any of us thought.

How it came about that Rob Hylands agreed to write an interrupt driven multi-channel sound driver for the game as well as supply the necessary music and FX required is one of those memories that I no longer have but I suspect he thought it was something to do with his time that seemed interesting. I wasn't even aware that Rob could code Z80 (and knowing Rob he probably couldn't.... no doubt he read a book or something and learnt it as he went along) but if he said he could do it then I believed him. As to recompense, well he certainly wasn't going to be getting any money from Catalyst or Activision and I wasn't exactly in a position to pay out anything but I did promise that I would come up with a way to get him the £200 he was asking for even if it meant he'd have to hold on for a few months or so for it.

Rob and I worked out how much memory he had for everything, what sound effects were needed as well as what short jingles and tunes we could fit in. Since Rob was promising a three channel interrupt driven sound routine it would have been possible to have in-game music playing along with the game but the large amount of data required for this wasn't going to fit into what little RAM I had left to me so it was dropped.

We eventually came up with over thirty different sound effects ranging from the short click of a bullet being fired up to the longest piece of music in the game, the tune that played when you successfully completed the last level. We tried to stick as close as we could to the arcade audio, in what I wanted and what Rob was able to coax out of the machine, but we did tailor things a little - FX that occurred repeatedly such as weapons fire and explosions were shortened since you'd be hearing an awful lot of them while one -offs like Jewel pickups and the jingle that signified the end of a level could be extended and made a bit more tuneful.

In these days when games boast of DTS soundtracks and orchestral quality music and sound effects it's difficult to comprehend just how basic game audio was back then. While machines such as the Commodore 64 came with SID (Sound Interface Device), a built-in Programmable Sound Generator,

```
SCROLL
        LD IX,SAD
SCROLL0
        LD A,(IX+3)
        AND A
        RET Z
        LD D,A
        ADD 32
        LD H,A
        LD E,(IX+2)
        DEC E
        LD L,E
        LD BC,30
        LDDR
        LD A,D
        ADD 38
        LD D,A
        LD E,(IX+2)
        LD H,(IX+1)
        LD L,E
        LD BC,4
        ADD IX,BC
 SCROLL1
        LD A,E
        AND 31
        JP Z,SCROLL0
        LD A,(DE)
        CP 220
        JP C,SCROLL1F
        LD A,(NOSCROLL)
        AND A
        JP Z,SCROLL3
        DEC E
        DEC L
        JP SCROLL1
;
SCROLL1F
        PUSH HL
        AND A
        JP Z,SCROLL1D
        LD A,H
        ADD 32
        LD B,A
        LD C,L
        DO 7
        LD A,(BC)
        LD (HL),A
        INC B
        INC H
        LOOP
        LD A,(BC)
        LD (HL),A
        LD B,D
        INC D
        INC D
        INC D
        LD A,1
        LD (DE),A
        JP SCROLL1E1
;
SCROLL1D
        LD B,D
        INC D
        INC D
        INC D
        LD A,(DE)
        AND A
        JP Z,SCROLL1E1
        XOR A
        DO 7
        LD (HL),A
        INC H
        LOOP
        LD (HL),A
        LD (DE),A
 SCROLL1E1
        LD D,B
        LD A,(DE)
        CP 156
        JP NZ,SCROLL1J
        LD A,(FRAME)
        AND 7
        CP 7
        JP NZ,SCROLL1C
;
SCROLL1J
        XOR A
        LD (DE),A
 SCROLL1C
        POP HL
        DEC E
        DEC L
        JP SCROLL1
;
SCROLL3
        LD BC,SCROLL7
        PUSH BC
        PUSH HL
        LD B,1
SCROLL4
        DEC E
        LD A,E
        AND 31
        JP Z,SCROLL5
        INC B
        LD A,(DE)
        CP 220
        JP NC,SCROLL4
;
SCROLL5
        LD A,B
        XOR 31
        LD L,A
        ADD A
        ADD L
        LD L,A
        LD H,0
        LD BC,SCROLL6
        ADD HL,BC
        POP BC
        DO 7
        PUSH HL
        LOOP
        LD H,B
        LD L,C
        RET
;
SCROLL6
        DO 30
        RL (HL)
        DEC L
        LOOP
        RL (HL)
        INC H
        LD L,C
        AND A
        RET
 SCROLL7
        LD H,B
        LD L,E
        JP SCROLL1
```

The Z80 source code for scrolling the screen that's at the heart of the game and this is exactly how it looked at the time. Programmers usually annotate their code to make it easier to understand but the PDS assembler could only handle source files up to a certain size so the more you commented your work the less room you had for the actual game code itself.

Spectrum users had to make do with single channel output that you programmed via a BEEP command. What made things worse was that while the Spectrum was BEEPing you couldn't run any other code, it was either sound or movement and if you wanted to give the illusion of both happening together then you had to do some pretty complicated and time-critical coding. That Rob managed to achieve in a few weeks what other coders had spent months and months creating and perfecting is a testament to his abilities, and while it may sound like I'm over-praising the man I can't state just how important he was in adding that vital ingredient to the finished game.

With Rob taking on the job of producing the audio it meant the pressure was off me and I now had a definite idea of how much memory was available for the completed game code - not much was the answer. Finishing off Level Seven was a lot easier now but it was still a stop-start thing, finding enough memory by going over and over existing routines, optimising and shortening them again and again, so if *R-Type* appears 'polished' (a word used by several reviewers later on) then it had nothing to do with making the game stand out but was rather a byproduct of getting the damn thing to fit into 42K of memory! Some routines were getting just plain silly, self-modifying code was getting more and more unreadable and the Stack was being used as a quick and dirty way to loop, saving bytes and registers. If that doesn't mean anything to you then don't worry but if you do understand it then you'll realize that debugging and error finding was starting to get much more complicated. I think the only reason I could keep track of everything was because I was never away from the computer long enough to forget it!

While I was finishing Level Seven Saul asked me if I'd put a demo together of the game, nothing fancy just a playable version of Level One that would be given away free with a magazine as a cover mounted tape. I managed to negotiate a fee of fifty quid with him for doing the demo which would only take me a few hours to produce as it was basically just taking the tape code out and making sure there was nothing left that could be used to turn it into the full version. An unfortunate fate that befell some other game demos.

I could have produced the demo right away but I wanted to wait until I was nearer to finishing the game to make sure it was as close as possible to the real thing. This wasn't as petty as it sounded, later on one magazine review of the game was illustrated with shots of Level One that showed the original graphics I used for the Power Ups which I had ditched for something else months before, so I don't know what version they had been playing. Besides, there was a good chance that in order to make room for more game code I'd have to change or delete something already written which wouldn't look too good if people ended up buying the game on the strength of a demo that was different from the finished product!

The cassette demo was given away with the November 1988 issue

Activision wouldn't run to actually giving me a free copy of a free program, I had to buy the magazine and rip it off the cover myself.

of Computer & Video Games magazine and a demo of the C64 version was on the other side of the tape. The magazine also had a full review of the Spectrum and ST versions of the game but not the C64 version, which it told readers to "*watch out for*". At the time of the review the ST version was still in the process of being written which makes the 'review' of it all the more suspect.

If I was having it easy for a while then it was Dave's turn to struggle. The C64 version was inching along slowly but he was still having problems with his sprite multiplexer and unfortunately I didn't know enough about the machine to make any kind of helpful contribution. While the hardware seemed to offer everything a game coder could want - more RAM, smooth hardware scrolling, raster line interrupts, eight hardware sprites - it was those sprites that were causing the problem. Multiplexing is easy in theory, you tell the processor to put down the eight sprites in the top part of the screen, do an interrupt on the display line immediately below the last one, quickly set up the sprite hardware with another eight sprite positions and data and fool the machine into putting down a second set on the same screen in the same frame. If you were really good at it you could even fit in a third set of eight before you ran out of time though it was pushing it to put down more than that.

The difficult part comes when sprites start to overlap, or one that was at the top of the screen moves down to the bottom, then you've got to start sorting Y position values and coming up with all sorts of ways to cut down on the processor time required. It also means that if you want more than sixteen or twenty four sprites on screen then you have to start changing the way the game plays, which isn't too difficult on a simple single-screen game like *Pacman* but on *R-Type* was turning into a bit of a nightmare.

It didn't help that my version was being used by Activision as a stick to beat Dave with, the argument being that if the Spectrum version did what it did purely through software then the hardware enhanced C64 version should at least be up to equaling if not surpassing it. What Dave was aiming for was an all-in-one routine where you set up the sprites you wanted, where you wanted, and the multiplexer took care of it all for you but every time he thought he had it sorted something would come along and mess it up. If I was succeeding because I was too inexperienced to realize that what I wanted to do was too much then Dave had the misfortune to keep running head-first into the brick wall of hardware that offered to do it all for you - as long as you did exactly what you were told that is!

As July ended and August began I started on what was to turn out to be the longest of all the levels to finish, not in terms of size but in the sheer time it took to put everything into it. Level Six is basically just a simple maze comprising walls and floors through which large aliens resembling giant pistons move up, down and around while small crawling centipede-like aliens hug the floors and ceilings and move back and forth repeatedly. The Pistons only move in straight lines and the Crawlers bounce back and forth against the walls which made syncing everything up to the background properly a time-consuming experience.

I had to program the movement of each alien a small section at a time and make sure they didn't end up moving through a wall or ceiling or sinking into the floor before moving onto the next part of their animation path. The only way to do this was to constantly assemble and download again and again and watch to make sure they only went where they were meant to go. With the other alien sprites as long as I got them to do generally what they were supposed to then I could get away with it however with this level everything had to be spot on, I couldn't even get away with re-using a sprite movement pattern since the motions of each alien were pretty much unique.

There is some confusion as to just what happens when you get to the end of Level Six since it doesn't have an End of Level Boss like the other levels, it just sits there pumping out Pistons for a while but some people insist that when they got this far there was indeed an End Boss. This all stems from the PC Engine\TurboGrafx-16 version where after blasting away a number of these Pistons the screen starts scrolling again and introduces a large alien that hugs the ceiling and occasionally splits in half shooting off fireballs. I suppose somebody somewhere decided that the original arcade ending wasn't exciting enough or lacked symmetry with the rest of the game but it's not there in the original so we didn't put it into our versions.

While I was putting Level Six together Rob was finishing off the sound and music. It took him only a couple of weeks to code a three channel sound system for the decidedly monophonic Spectrum, an amazing feat coming as it does with all the music and sound effect we had agreed on. I was particularly anxious to hear the sound the large Circle Lasers made when fired as I think I may have gone a little over the top in describing what I wanted them to sound like to Rob...."*I want it to sound like the Universe is splitting in half and screaming in pain!*" While it didn't quite live up to my hysterical demands both it and the rest were miles and miles away from the simple bleeps and blurps that I had come up with for *Rampage*. Plugging Rob's code - which he had christened BINASCAN - into my own was pretty straight-forward, his full and complete instructions on how to use the code were…..

MAKE SURE THAT SOUND WAVE PATTERN BYTES DO NOT CROSS PAGE BOUNDARY. ALSO MAKE SURE THERE ARE 2 FREE BYTES BEFORE THE ORG OPCODE (FOR USE IN PROG REVECTORED STACK)

LD HL, A*256+B+C WHERE
A = SOUND NUMBER.
B = FULL/HALF OR QUART NOTE.
C = CHANNEL NUMBER.
THEN CALL BEEP

…..and that was it!

Rob's code slid smoothly into mine, no loose ends or corrupted registers, Rob took care of everything properly and when I set up all the routines and tried it out it all worked beautifully. We had one problem right at the beginning when some of the sound effects sounded the same no matter what value we made B above but Rob soon fixed that and it has worked perfectly ever since. I had come up with an idea on how I could get Rob his money but since it involved me lying to Activision I wanted to keep it to myself for a while, besides it only stood a chance of working when the game was finished. IF the game was finished I should have said, for while Rob and I were coding away Wainwright had come up with a last ditch plan to regain control of what he considered his game code and make some money out of Activision in the process. Unfortunately it had the drawback that I would end up in Court if I didn't play along.

Throughout most of the coding so far I had been sending all new builds of the game directly to Activision as Catalyst didn’t seem in the least bit interested in seeing what I was producing, but with more and more previews appearing in magazines and Activision happy with what I was supplying them Wainwright started to take notice. And he came up with an idea. His plan was simple, after I had finished the game I would deliver all the code to him rather than Activision so he could approach them for more money using it as a bargaining chip. Oh, and if I didn't do what he said he'd sue me!

While I was digesting this threat, and just a few days later, I received a call from someone at Activision who wanted to speak to me about the game. After the usual small talk he wanted to know when the game would be completed and, seemingly as an afterthought, when I would be delivering it to them? I suppose Wainwright had been testing the waters as to his big idea with them hence the phone call because as I explained that once it was finished I would be delivering it to Catalyst (since I believed I worked for them) things got a bit frosty. In no uncertain terms I was told that once the game was finished I was to deliver it directly to Activision not Wainwright. Oh, and if I didn't do just that Activision would sue me!

It was an...interesting...position to be in. If I delivered the finished game to Wainwright then Activision would sue me but if I delivered the finished game to Activision then Wainwright would sue me. Now I'm a bit older, if not wiser, I realize that things weren't as black as I thought they were at the time. Legally Activision couldn't touch me since I worked for Catalyst not them and the

most they could have done if I gave the code to Wainwright was sue him for it, there was no legal agreement between myself and Activision so how could I break it? But the question was did I in fact work for Catalyst?

I didn't have a written contract to produce *R-Type* and if I had then it had certainly been breached when they hadn't paid me. The most that could be said was that there was a verbal agreement and I'd accepted money from them over the months to produce the game. Also Wainwright could have used the document I'd produced when we negotiated for the PC and PDS hardware in lieu of pay to show that it was actually he who was entitled to the finished code.

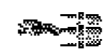
But there's an interesting point that none of us considered at the time and that was since I didn't have a written contract with Catalyst or Activision and Wainwright had signed a document saying that he would give me the development hardware when I finished the game then it could be argued that until I did so I was the legal owner of the code and could do whatever I wanted with it!

Unfortunately back in August 1988 I didn't even stop to consider any of the possible legal ways out of the situation I was in, all I knew was that whatever choice I made I was going to get sued by somebody so I had to come up with a third alternative. What I needed was some expert advice, someone who knew all the ins and outs of the software business and who would be on my side, which is why I put a call in to Jacqui Lyons at Marjacq and asked her what I should do.

I told Jacqui that I really wanted Activision to have the code since I wouldn't be working for Wainwright again and that they would be more likely to offer me future work if I was in their good books. I also said that if Wainwright got the code and somehow managed to get money out of Activision he would probably insist on paying me off and asking for the PC and PDS back, which I needed if I was to go out on my own.

While this was all new to me I suppose Jacqui had heard it so many times before that it didn't come as much of a surprise, so her suggestion was simple and direct - when I finished the game I should give it to her and tell Activision and Wainwright that they could fight it out between themselves if they wanted to and leave me out of it. This was exactly the answer I wanted to hear especially since she rated Activision a lot higher than Catalyst Coders!

I decided I wouldn't tell Wainwright or Activision what I was going to do, just let each one think they were going to get the code at the end and hopefully they'd stay off my back until the game was finished by which time it would be too late for them to do anything. Knowing Jacqui's ties with Activision at the time I suspect she told them what I intended to do anyway but it did mean I could now work on the game in relative peace and quiet.



It was all starting to catch up with me - the long nights, the worry over final delivery, the arguments about money and hardware, being away from home and the fact that Level Six still needed to be finished -

but all it did was make me more determined to finish the damn game and get the hell out of Fareham as quickly as I could. There was the rudimentary development schedule I gave to Nick that I was trying to stick to, and if the note in the back of my Z80 manual was correct then the game was due to be finished on August the 9th which it definitely wasn't going to be! Realistically Level Six needed to be finished, some routines that I'd been putting off for weeks because of their difficulty written and tested, Level Eight to be done and the final End of Game scrolling message to be included.

One of those routines I was having trouble writing was control of the Homing Missiles, part of the weapons system that launched and then locked on to a specific alien, tracking and following it around the screen until it hit. I did try and work out ways to do it correctly but gave up and went with a simple approximation since I didn't have enough memory available to do it properly, didn't have enough space on screen to do it justice and was getting so fed up with the whole thing I couldn't be bothered to spend any more time on it.

I think most coders eventually reach a stage where getting it done rather than getting it right takes precedence - you're faced with spending days coding some intricate routine or alternatively hacking something together that looks similar but can be written in a few hours instead. Sometimes it's even easier to drop it altogether, a move similar to the cliché of a movie director who finds himself a week behind only to reach for the script, tear out a handful of pages and announce that everything is now back on schedule!

I suppose I was lucky that I hit my brick wall when there was very little left to actually do, the Flame Weapon could have done with some tweaking and there were glitches in some of the pixel sprite routines, the End of Level Boss from Level Five could do with a makeover and I could always have gone back and fixed the Fish Eggs from Level Four but enough was enough. All I wanted to do was finish the game and go home.

I spent a few days bouncing between finishing off Level Six and putting in the final pieces of code, the last major block to go into the game was the tape loading system. Since the game was going to be multi-load from cassette I needed to have my own built-in tape routines, it wasn't as if I could exit to BASIC to do a quick LOAD "" CODE. Most coders utilised the existing tape load routines that could be accessed through the Spectrum ROM itself but using those meant having to deal with slow transfer speeds and a relatively insecure system.

Before *R-Type* I had been messing around with these routines, copying them out of ROM and fiddling with some of the timing values to increase the transfer speed and as a byproduct changing the colour of the usual blue and yellow loading lines to something different. There wasn't anything special about it, it was a quick and easy way to come up with customized Load\Save routines and a lot of coders were doing it, the faster loading speeds and unusual loading patterns also made casual piracy of the games just that little bit more

difficult. If you knew the right people you could buy tape handling code that was custom written and which offered greater stability and security over the tweaked-ROM versions but I had the code I was going to use all ready to go so slotted it into the game, wrote some routines to control everything and it seemed to work just fine.

Just one more level to go and it would all be over.

I had kept Level Eight for last since I considered it the easiest of all the levels to code, which in my state of mind at the time was something I really needed. It's the shortest level in the game and there aren't that many aliens appearing, those that do tend to follow a very simple movement pattern and the End of Level Boss just sits there opening and closing its mouth waiting for you to destroy it. I was so confident about this last level that I even set myself a target, from start to finish I was going to code the whole thing in just one day.

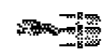
I almost made it. I put in a good sixteen hours determined to finish it but tiredness beat me. I usually take it as a sign I need to get some sleep when I look at a piece of code I wrote a few minutes earlier and wonder what the hell it does. More than once I've deliberately not saved off the code I was working on because I wasn't sure of what I'd just written - it's much safer to waste an hour or so of effort and rely on an earlier save that you know is correct than mess everything up.

I picked up where I left off the next morning and finished it by the afternoon. I can't go into any more detail about the Level Eight because, to be honest, there's not really a lot there. Compared to the earlier stages Level Eight is quite bland and lacking in any originality, you get the feeling even the game designers at IREM had left early for the weekend when they got this far.

The final scrolling message took a bit of thought since I wanted the ships that fly across the screen at the end while it scrolls up to move both in front of and behind it. The only way to do that was to make the whole block of text a sprite which means the end 'Well Done' text has the distinction of being the second largest sprite in the whole game. In case you never made it that far the end message reads -

THANKS TO YOUR BRAVE
FIGHTING THE COSMOS
RESTORED PEACE.
THE BYDO EMPIRE WAS
ANNIHILATED TO NEVER
SCARE PEOPLE AGAIN.
YOUR NAME WILL REMAIN
IN THE UNIVERSE FOREVER.
THANK YOU FOR PLAYING
THE GAME TO THE END.
THE PROGRAMMER.

All I had to do then was time things so that Rob's end game music finished playing when everything was clear and that was that, Level Eight was done. And so was the game!



At this stage the game was in an Alpha delivery state, a phrase the meaning of which I had to work out for myself. My take on the three final delivery states of a game are:

Alpha - The game is finished but not necessarily complete. Placeholder graphics may be used while waiting for the real ones, some routines may be missing but shouldn't be detrimental to the game, there may or may not be music and sound and there will definitely be bugs some of which may be critical. In short you can play the game and get a feel of it but don't expect it to work properly.

Beta - The game is finished and complete. Everything that's going to be in the release version is there and the game should play accordingly. There should be no critical bugs but there will be others to be discovered while testing, no Beta should be delivered knowingly with bugs. The Beta gets tested, bugs are found and reported back, fixed and the game resubmitted. This can go on until every bug is fixed, or someone higher up loses patience and says *"we can't afford to spend any more time on this, ship it as it is!"*

Master\Delivery - The final version everyone agrees on, this is the one that gets shipped to the duplication plant for mass production. It's often called a Gold Master, a reference to the CD\DVD manufacturing process.

While my game had everything it wasn't yet linked together properly, every level I had been submitting to Activision was standalone not multi-load and while the code to handle it was in the game I hadn't implemented it yet. It did mean that Activision could load up any particular level and show it off without having to play through the whole thing which was handy for those magazine screenshots of later levels but it wasn't a playable game as such. Not yet.

I decided that I was going to leave Fareham and return to Swansea where I could finish the game off in peace and quiet, I was fed up with spending my life in a room above a kitchen showroom so I didn't have to convince myself too much that now was the time to go. Activision weren't exactly shouting for the game as they intended to release all three versions at the same time in November and as it had only just turned September there was no big rush. I put together a demo of Level Eight for Nick and left it at the office for him to pick up the next time he came over and called my brother-in-law who offered to drive a van down to Fareham to pick me and the PC up and bring us back to Swansea. I deliberately didn't tell Wainwright I was going, I hadn't heard from him since his attempt to get the game off me and if he wanted to take it any further he'd have to come to South Wales and meet me on my own turf.

I had a couple of days grace before the Saturday my brother-in-law was due to arrive so I decided now was the time to try and get some money for Rob. When I spoke to Saul and Nick to let them know that the game was finished (if not quite ready) I managed to slide into the conversation that while I was prepared to sign off on the game I really couldn't since I didn't own the copyright to the sound code inside and that I'd have to get the OK from Rob first. Rob, I continued, was happy to sign the code over to me for a nominal amount which alas I wasn't in

any position to provide due to Wainwright's actions but if Activision could come up with the meager sum of £200 then everything would be fine.

I did lay it on a bit thick but we were both dancing around, £200 to Activision was chicken-feed seeing how much they were spending on advertising for the game and they knew they had a 99% finished product ready and waiting, they weren't going to jeopardise things for such a relatively small amount. Saul told me to tell Rob to put an invoice in and they'd pay it, which they could have said when all the trouble started but then that was Activision for you.

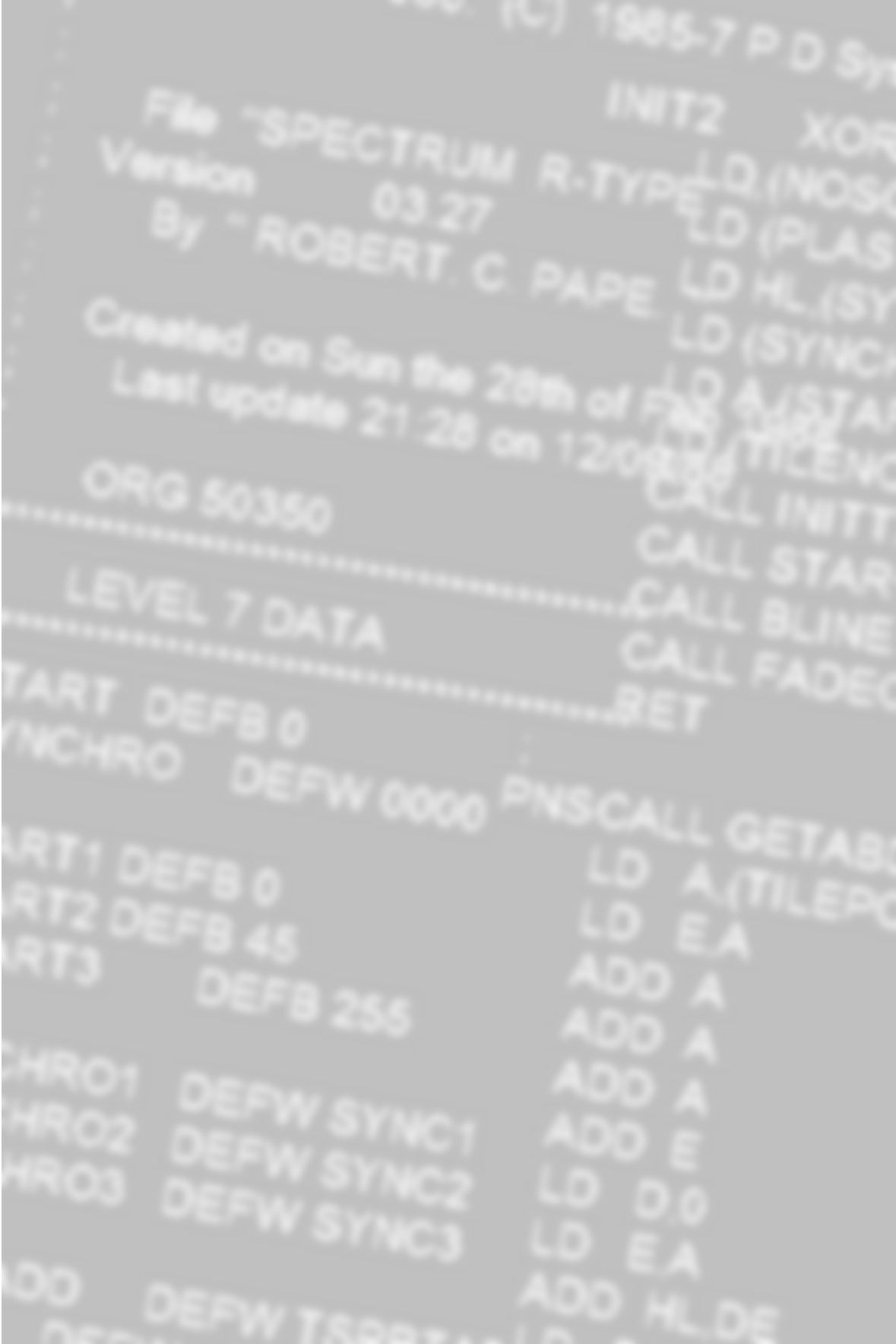
My last Friday in Fareham started off well but ended badly, even today I clearly remember how it went. As per usual Friday evening was time for my one proper meal of the week - my favourite Beef with Bamboo Shoots, Water Chestnuts and a plate of chips - and as it would be the last time I'd be having it I was really looking forward to it. I had just started eating when the phone rang and one of the boys handed it over saying Kightley wanted to speak to me. What followed was one of the nastiest telephone conversations I've ever had in my life with Kightley shouting down the phone accusing me of betraying him, taking him for a ride, ripping him off, dropping him in the shit and capping everything by demanding that since I worked for Designmaker I was to give him the finished *R-Type* code NOW!

When someone is yelling down the phone at you you can't talk back quietly so it wasn't long before I was shouting as well, and not having anything to lose I wasn't holding back. After telling him that I stopped working for Designmaker when the money ran out I pointed out that my dealings were directly with Catalyst now and if anyone had dropped him in the shit it was Wainwright. Furthermore he could say and do anything he wanted but he wasn't getting the code off me and that if he wanted to sue me for it he could join the end of the queue behind all the others!

For twenty minutes the two of us had a slanging match over the phone until I got so worked up I went dead calm, always a bad sign since it means I've gone past hot anger and into cold rage. I told Kightley that I was going to hang up now otherwise I was going to say something I'd regret and frankly he wasn't worth the ******* trouble, which I proceeded to do. And that was the last time I ever spoke to Richard Kightley.

I had to chuck the food, it had gone cold and the office didn't run to the luxury of a microwave oven, which wasn't the best of ways to end my stay in Fareham. When they come to make a film of all this I suppose it will be rewritten so we all end up in a group hug or something before I drive away into the setting sun, which will be a lot more dramatic than eating cold chips but not as truthful. The next day my brother-in-law turned up in his van and I loaded the boxed up PC into the back along with the rest of my stuff, said my goodbyes and started on the long drive back to Swansea.

AFTERMATH


(C) 1985-7 P D Sy
INIT2 XOR
File "SPECTRUM R-TYPE
Version 03.27
By " ROBERT C PAPE
Created on Sun the 28th of
Last update 21:28 on 12/08
ORG 50350
LEVEL 7 DATA
TART DEFB 0
YNCHRO DEFW 0000
RT1 DEFB 0
RT2 DEFB 45
RT3 DEFB 255
HRO1 DEFW SYNC1
HRO2 DEFW SYNC2
HRO3 DEFW SYNC3
DD DEFW TSRB
LD (NOSC
LD (PLAS
LD HL,(SY
LD (SYNC
CALL INITT
CALL STAR
CALL BLINE
CALL FADEC
RET
PNSCALL GETABS
LD A,(TILEPO
LD E,A
ADD A
ADD A
ADD A
ADD E
LD D,0
LD E,A
ADD HL,DE

Over the weekend I unpacked my stuff and set up the PC in my bedroom. I didn't have anything approaching a proper computer desk just an old kitchen table with fold out leaves that needed a bit of bracing to take the all metal PC, monitor and keyboard that threatened to collapse it at any moment. There really wasn't much left to do, just implement the tape system, make sure that nothing got overwritten that wasn't supposed to when you loaded in a new level and finally add all the copyright and trademark text at the beginning of the game. Of course if Activision found something wrong while testing the game then I'd have to fix it but I was aiming for a beta version in a week or so and that seemed achievable.

As I mentioned before I had a fast customised tape loading routine in place based on hacked code from the Spectrum ROM so the first thing to do was give it something to actually load. Since I was going to be messing around quite a bit with blocks of code and cassette tape I decided to save myself a lot of time and effort and write a small program that would create and save the levels automatically to tape for me.

Each level was loaded individually in the game, assembled and sent to the Spectrum where it would be saved off as a block of code onto the Swiftdisc floppy disc. My small program would then read each level off the floppy disc and proceed to save it out to tape, suitably sped up, along with a small block of code that acted as a header to make sure you were loading the right level during the game. I also had a block of code that was just the loader which I was going to use for the main bulk of the program itself, so not only were the levels going to load faster than normal but the game itself would as well.

There wasn't much else to be done with the tape routines since I'd taken care when building the levels to make sure that everything that needed to ran indirectly and through pointers, which meant that while all the sprites, background graphics, routines etc. were in different places in each level there were tables in those levels at fixed locations that never changed which told the main program where they were.

For example, if I needed to draw (say) alien sprite twenty six in Level Four then all I had to do was check the sprite table loaded in as part of Level Four and look up the twenty sixth entry, which gave me an address where that sprite was in the Level Four code. This is where a good assembler comes into its own as instead of laboriously entering each address one at a time manually you can, through the use of labels and the appropriate directives, set up the source code to generate the values automatically when assembling.

For some reason one of the last things to go into a game are the Game Credits, the bits which list who worked on the game, the official texts, the thank you's, the copyrights and any other cryptic messages or in-jokes. Usually these get put at the end of the game, after all who wants to have to sit through a load of boring stuff every time you load the game, but unfortunately Activision wanted just that for *R-T*ype so I had to oblige. Once the main block of code has loaded the player has to wait almost

fourty seconds before being able to play the game during which time a stilted piece of text scrolls along the bottom of the screen, that you can't even bypass by pressing a key. If you've never seen the game it says:

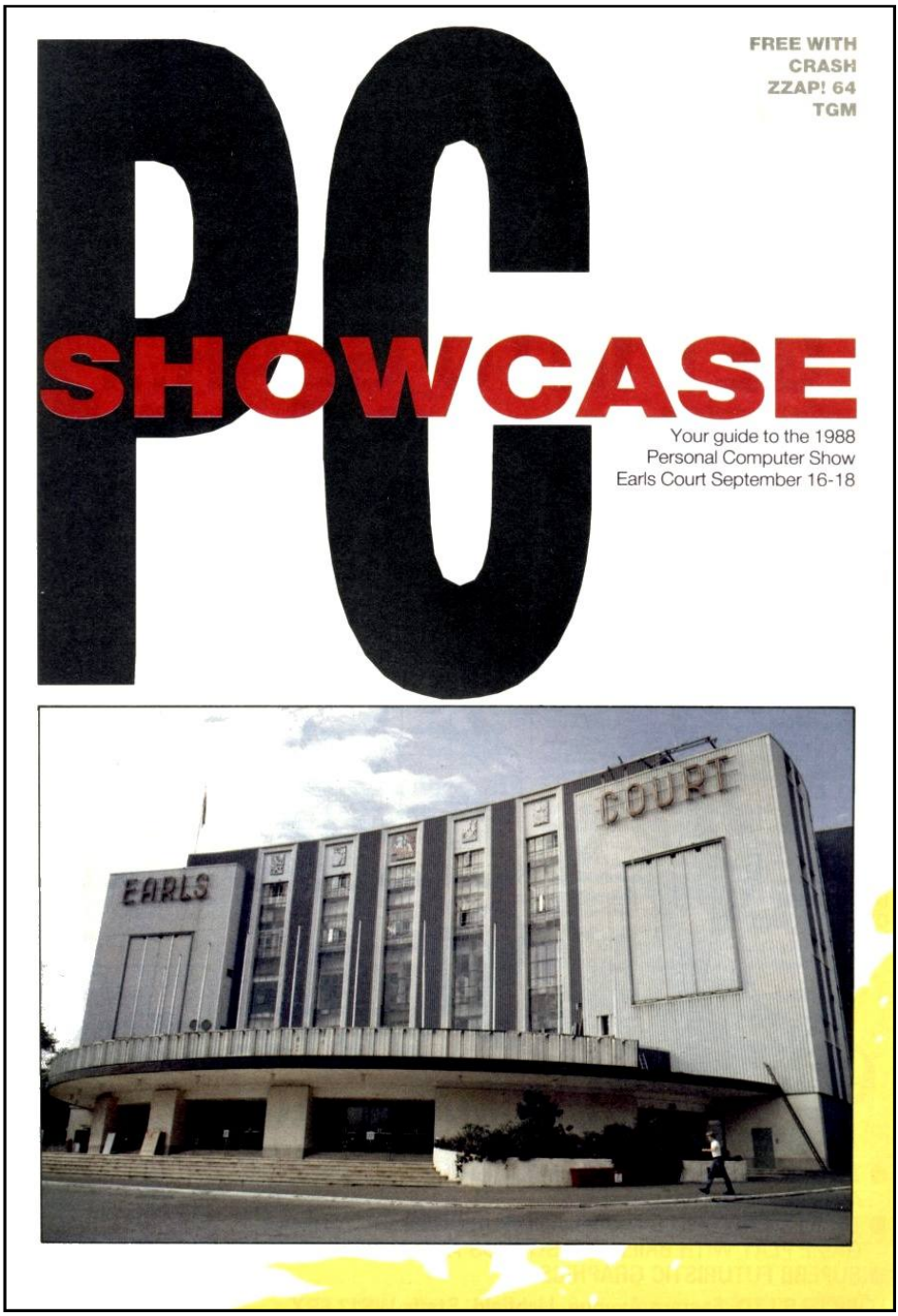
R TYPE tm HAS BEEN PROGRAMMED BY BOB PAPE .....WITH GRAPHICS BY 'MAK' COMPUTER GRAPHICS & SOUND CODE BY ROB HYLANDS. R TYPE tm IS A TRADEMARK OF THE IREM CORPORATION, COPYRIGHT ©1987, AND IS LICENSED TO ELECTRIC DREAMS SOFTWARE. THIS PROGRAM IS A SOFTWARE STUDIOS PRODUCTION & IS COPYRIGHT ©1988 BY ELECTRIC DREAMS SOFTWARE.

To break it down a little; **tm** is the abbreviation for Trademark and the use of both the word **Copyright** and the International Copyright symbol © together are to allow for those countries where the symbol on it's own is not recognized. **MAK Computer Graphics** is the name Mark Jones used to use to sound more professional (I used to use 'RCP Software', it came in handy when dealing with people who wouldn't ordinarily deal with a single bedroom coder). **Electric Dreams** was a subsidiary label of Activision owned by Rod Cousens though it was often hard to tell why some games came out on the former instead of the latter.

As far as I was concerned I had been writing the game for Activision because as everyone knew Electric Dreams WAS Activision. **Software Studios** however came completely out of the blue as I had no idea who or what they were and for a long time I believed it was another name Wainwright was using for Catalyst Coders. It turns out that it was actually Electric Dream's in-house development arm run by Producer John Dean and Game-tester Dave Cummins acting as a sort of umbrella for Electric Dream projects, but if either of them played any part in the production of the Spectrum version of R-Type then it was news to me!

With the credits in place, the tape turbo-loader working and everything running as it should the game was finally complete. I put together a master tape of the game, another tape containing all the source code and a list of POKE's to be used in a Multiface or similar so that Activision could play as invulnerable or through to any level required. I made sure the tape loaded OK, used the POKE's to run through all the Levels and packaged everything up. I was still going to give everything to Jacqui Lyons to hold even though I had a feeling she'd end up giving it to Activision as it meant I felt I had some control over what was happening.

Rather than post the game code off I was planning to give everything to Jacqui personally when I was next in London, which was going to be rather soon as the annual PCW Show was coming up and I intended to be there. For those too young to know or too old to remember the PCW Show was run\sponsored by Personal Computer World magazine and was THE computer event of the



year to attend in the UK. Usually held in a large venue such as Olympia or Earls Court Exhibition Centre the show attracted pretty much every major software house in Britain at the time and was aimed squarely at the average computer user who wanted to see what was going to be in the

shops at Christmas.

For a few years the PCW Show was an interesting mix of professional game companies, custom hardware suppliers, one-man-band entrepreneurs and smaller software houses but that soon all changed. Complaints that loud beeps and music from games were putting off the more business orientated customers led to the segregation of games stands and the 'proper' computer exhibitors would go on to complain about teenagers and oiks raiding their stands in the traditional search for free handouts and plastic carrier bags. Where before you could stop by a stand, buy a new piece of Spectrum hardware or games at a special exhibition-only price and even chat with the programmers responsible for what you were playing the PCW Show became more and more focused on the business side of things and a much less friendly event.

But back in 1988 it was an enjoyable place to be, even more so because of the nature of Activision's presence that year which was by invitation only and required another visit to Southampton to put into practice.

All of us had been trying to get invitations to Activision's PCW stand for months because that year they had decided not to have a stand as such at the show but instead a ‘presence’ at a luxury hotel in Mayfair, what's more they would be transporting everyone to and from the hotel via a fleet of stretch limousines which made it all the more imperative that we got hold of an invitation since no ticket no entry! All the coders who were working for Activision had been asking again and again for the invitations and when I left Fareham it was with the promise that one would be sent to me.

With the show just a few days away and nothing in the post I decided that the personal approach was needed so a return to Southampton was planned, and while I was there I'd find out what was happening with Dave and Jim and the C64 version of *R-Type*. I'd had a call from Karl that Activision had got fed up with their lack of progress, which wasn't at all helped by me saying I'd finished the game, and had ordered the pair to work in-house at the office in Southampton. This didn't sound too bad as we'd all had to do it some time but Karl went on to say that they were being held there virtually incommunicado and the pressure was beginning to show, which sounded really worrying.

The first day of the PCW Show was usually Trade Only, which meant nothing as it was so easy to get tickets to it that you had pretty much the same audience that day as you did the others. The day before, Thursday, I bought a train ticket to London and headed off to see Jacqui Lyons. We had a chat and I gave her the *R-Type* code package to hang onto, which meant if I bumped into Wainwright at Earl's Court demanding the game I had an out. I caught an afternoon train to Southampton from Waterloo and met up with Mike Archer whose sofa I would be borrowing that night to sleep on and we made our way to Activision's office.

What Karl had said turned out to be true as Dave and Jim were there in one of the glass fronted conference rooms all by themselves. They

weren't allowed to leave it and no one was allowed to enter or talk to them without clearing it with a manager first. The phone in their room had also been removed so as not to provide any distractions. At this point it looked like they had solved the sprite multiplexer problem bar a few glitches now and again and were trying to play catch up with the game itself.

I was only allowed to see them for ten minutes and it was obvious that they were both highly stressed and close to the edge and if there was anything guaranteed to fuel this it was the work conditions they had to endure. It's well known that this particular Commodore 64 version of *R-Type* was never released, instead another version coded by Manfred Trenz at Rainbow Arts who stepped in at short notice was, and I have to say that I doubted this story at the time and I don't believe it to this day. Even Activision were convinced that the C64 version would be finished on time as a press release they released at the PCW Show trumpeting the game included the line: "*Programmers Bob Pape (Spectrum), Karl Jeffrey (Atari ST) and Dave Jolliff (C64) came together to work as a team on the project.*"

With the game planned for a November release Activision were publicly announcing that everything was OK with the game as late as mid-September and you do have to wonder just when exactly it was that they started talking to Trenz and Rainbow Arts. It has been reported since that Trenz and his team put together their version of *R-Type* in only seven weeks, which means they were either coding their version while Dave and Jim were still working on the original or work started on it immediately after the original was canned. You do have to question Activision's behaviour in the first case and their loyalty to Dave and Jim in the second.

Now that the original version of the game has been posted online you can see for yourself how close Dave and Jim were from finishing the game. I really believe that if they had been given that seven weeks instead of Trenz their version of *R-Type* would have turned out just as good, if not better, than the version that finally went on sale.

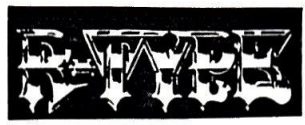
Although I spoke to Dave Jolliff many times after this I never asked him about his time in-house, why they cancelled his version or if he knew why Rainbow Arts - who Activision just months earlier were prepared to sue - were suddenly now their new best friends. The state Dave was in in that conference room was not something I wanted to remind him of but the game was progressing, slowly yes, but it would have come together. If Dave and Jim's version didn't make it it was because Activision pushed them toward a nervous breakdown getting them to finish the game, and the fact that they must have been in talks with Rainbow Arts while doing so makes everything about it just that bit more distasteful.

To rub salt into the wounds Dave and Jim were effectively barred from leaving the building so wouldn't be going to the PCW Show even though their game would be on display and was being promoted by Activision. Dave had worked the PCW Show the previous year as part of the

ELECTRIC DREAMS SOFTWARE

NEWS

PC Show, September 1988



Are you ready for R-Type?

Are you ready for the fastest, meanest arcade game around?

You're in control of a powerful R-9 interstellar craft, charged full of awesome firepower, which you must manoeuvre through the monstrous Bydo Empire.

Armed with the latest **R-TYPE** lasers, you possess spectacular fighting power. But your enemies are ruthless in their attack. To survive, you'll need to seek out the power crystals that give you added weapons strength and protection.

Can you endure the onslaught? Are you a mighty-enough warrior to survive eight stages of battle to face the final confrontation and win?

Electric Dreams have produced a convincing conversion from the IREM classic, that includes all the features of the original and more. **R-TYPE** will burst onto your screen in wonderful technicolour glory, and keep you fighting all the way. Programmers Bob Pape (Spectrum), Karl Jeffrey (Atari ST) and Dave Jolliff (C64) came together to work as a team on the project. Their previous collaboration was on the highly acclaimed RAMPAGE, which was a Christmas hit for Activision last year.

R-TYPE smashed the moulds of previous space games when first released by IREM last year. Featuring a multitude of power-ups and firing abilities, **R-TYPE** expands upon the straightforward shooting contest theme. One of the more unusual features is a probe that can be collected at the beginning of the game. This strange little craft increases firepower and acts to some extent like a shield. The player can manoeuvre the probe ahead of his own craft to clear a pathway, or detach and place the probe behind the craft, or even let it trail behind for further protection.

So forget all the feeble immitations - **R-TYPE** is the only one worth waiting for. The arcade scrolling shoot 'em up of the year is here - an outrageous game of cosmic encounter that has the entire galaxy talking!

Release Date: November 1988

ATARI ST	£24.99	COMMODORE 64	£ 9.99/£14.99
SPECTRUM	£ 9.99	AMSTRAD CPC	£ 9.99/£14.99

For further information, contact Amanda Barry or Janick Yeung on 01-431 1101

PC Show press release announcing the game's availability.

Activision stand team and now he was being treated like some kind of leper. It was just all wrong.

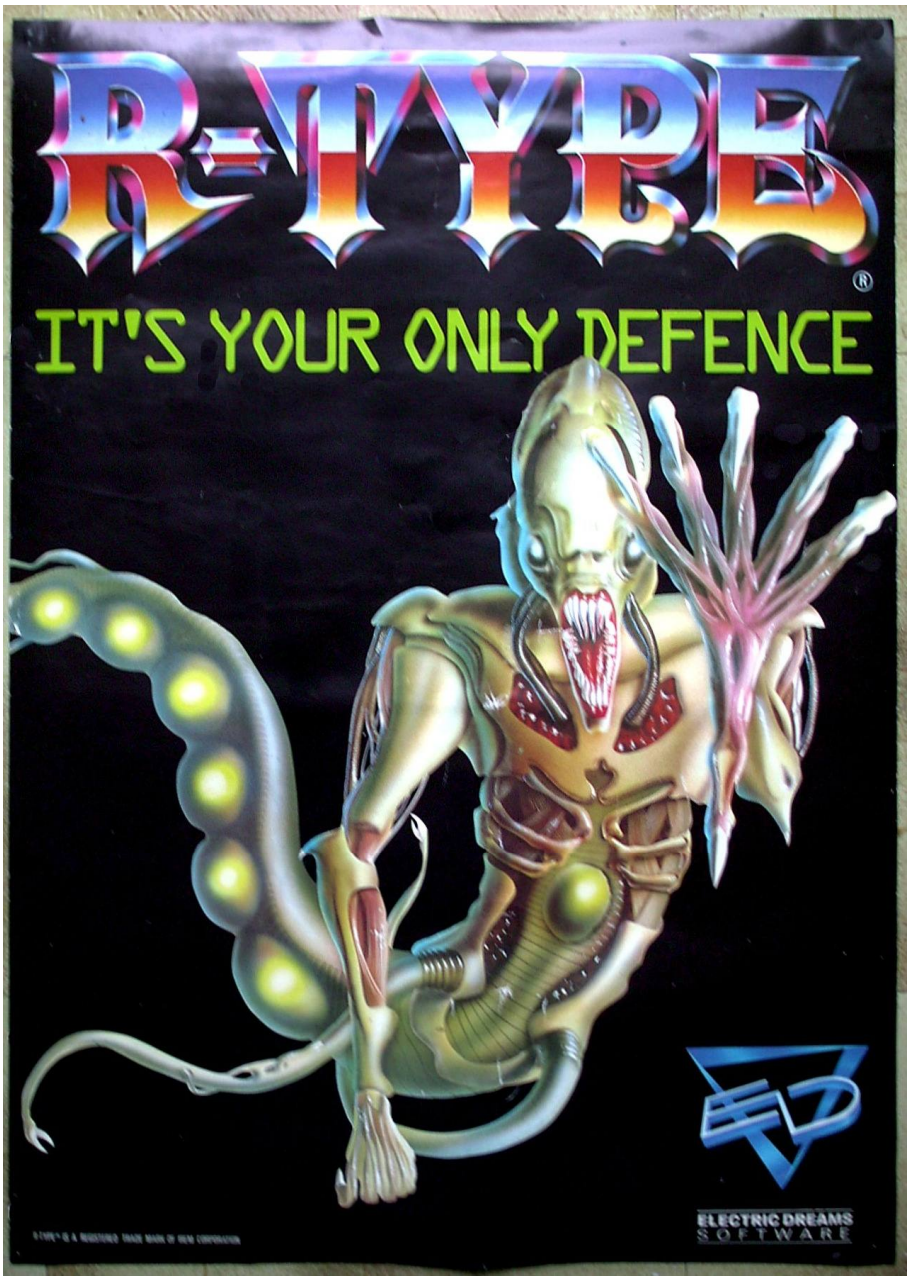
After talking to Dave and Jim, Mike and I picked up our show invitations, left the office and headed off for a bite to eat. Mike's sofa in Portsmouth was comfy enough to get a few hours sleep on and in the morning we caught a train up to London and headed over to Earls Court. There had been an informal agreement among some of the Portsmouth and Southampton coders that we would meet up outside and try and go over to Activision's hotel together after we'd had a walk around the show for a bit. We headed inside, found the location of a little booth Activision were using as a staging post for the limousines and arranged a time to meet back there later. At the agreed upon time we all congregated at the booth and handed over our invitations, which probably came as a bit of a shock to the young lady manning it as from what we had seen the usual invitees tended to be middle aged and wearing suits. A group of noisy coders in jeans and T-shirts was probably a little different from what she'd been dealing with but she validated us all and we made our way outside to the waiting limo.

Our group consisted of myself, Mike Archer, Mark Jones, Karl, Steve Lamb, Jason Austin and a few others I didn't know. Henry Clark, a coder from Catalyst's Glasgow arm was there also but he didn't have an invitation and was desperate to come as well but luckily for him I had picked up a spare from the office and literally threw it at him as he ran back to the booth to get the all important stamp.

After reading about teenagers buying expensive cars with all the money they were making out of games it was nice to be part of the lifestyle, if only for a while. Our black stretch limo ferried us over to a luxury hotel in Mayfair where Activision's free food, drinks and more importantly free goodies awaited us. They had to let us in of course, we had our invitations stamped and validated, but it was pretty obvious they wanted to see the back of us sooner rather than later.

Level One of my *R-Type* was on display and they were running some kind of 'Get a High Score and Win a Prize' with Karl's ST version of the same. Actually it was the second disk of the game that Karl had given them as somebody had walked off with the first, which seeing how the ST was in a sealed plastic box with a metal bar blocking the disk slot was quite an achievement. Taking pride of place however was a pneumatic version of the new SEGA arcade game *Afterburner,* on free play to celebrate the Christmas release of the game which turned out to be an instant, if noisy, attraction to us all.

After doing the rounds of the games being shown and waggling a few joysticks there wasn't much else to do but fill up with the free food and drink on offer and stuff a load of *Afterburner* T-Shirts into a carrier bag along with plenty of posters and the usual mix of fliers, press notices and catalogues. Everyone else was going to go back to the PCW Show but I had seen enough so I swapped the limo for the Tube and headed over to the West End to do some shopping before catching the train back to Swansea.



Original Activision R-Type poster. It looks like the graphic designer had to go home early that day, so did the guy who wrote the slogans!

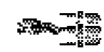
Amongst the promotional material at the PCW Show was a press release trumpeting the game and when it would be available. Added almost as an afterthought was the information that there would be a version of *R-Type* released for Amstrad CPC computers alongside

the others which came as somewhat of a surprise to Karl, Dave and myself to say the least. Since Amstrad and Spectrum computers shared the same Zilog Z80 CPU I had the feeling that I was going to be asked to produce this version as well - that is assuming no one at Activision had been working on a version and hadn't told us. This wasn't as farfetched as it sounds. Activision USA were marketing a PC version of *Rampage* that had been sent to them out of the blue by a complete stranger, they had liked it so much that they bought it and put it out alongside the other versions we had produced.

My initial thoughts were correct however, after speaking to Activision they told me that they didn't have anyone lined up to code the Amstrad and would I be interested in supplying it after finishing the Spectrum version? I didn't really want to take on the job, I had no desire to learn to code for the Amstrad and I wanted to be free and clear to start with Jacqui Lyons when I'd finished. Activision's second idea sounded a lot better, that they'd take my source code and give it to somebody to convert for the Amstrad, a common solution at the time for quickly producing an Amstrad game though it meant not taking full advantage of the computer's hardware capabilities. Of course this could only happen once the game was completely finished which didn't leave much time for the coder to do his work, but I never realized just how little time that would be.

Activision told me that they had someone in mind to do the conversion and would I liaise with him on my code and routines? I never actually met Keith Goodyer, only spoke to him over the phone, but he did a fantastic job of taking my car-crash mess of code and turning it into a fully working game for the Amstrad. I don't think my source code helped him much, restrictions on the size of the file the PDS software could handle meant adding comments to the code ate into memory, it was either code or comments, but Keith has been gracious to say that the labels I used in the code were somewhat self-descriptive.

What was amazing - and something I only found out quite recently - was that Activision had given Keith just twenty one days to do the complete conversion in order to release it at the same time as the other versions! That it stands today as one of the best arcade games for the Amstrad ever released (despite the unwarranted stigma of being 'just a Spectrum port') is a great tribute to Keith's skills as an accomplished coder.



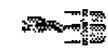
Things quietened right down after the PCW Show, Activision hadn't asked for any changes to the game and as far as I was concerned I was done with them. Jacqui Lyons was hard at work on my behalf and at the beginning of October she told me she had been speaking to Mark Cale at System 3 (a software house based at Pinner in Middlesex) and he wanted me to code the Spectrum version of a game called *Tusker* for them.

Tusker turned out to be a game everyone in the business worked on at one time or another - or so it seemed - boasting a sixty-plus page

game specification and seven graphic artists for the Commodore 64 version alone. Years afterwards I'd be talking to total strangers in the games business and they'd mention having worked on this game in some way, it was almost like being in a Secret Society with "*Tusker*" being the password to identify other members. While Jacqui was negotiating with System 3 I had two phone calls from Activision one of which was very good news indeed while the second would set in motion a series of events that would cause Activision a few headaches months later on.

First the good news. Activision wanted to give me £500 outright as a thank you for finishing the game early and for doing such a good job which, seeing as I hadn't received any money whatsoever for several months, was more than welcome. But could they just pop a cheque in the post and be done with it? No, to get the money I'd have to sign for it in person and at some offices I'd never even heard of which is why a few days after the call I was in London and heading for the Tube. It turned out that Activision's accounts department was based at their original office in Hampstead, North London, a building that no one even seemed to know existed and which resembled an old solicitor's office. But I turned up there, dutifully signed my name, received a cheque, got back on the Tube and headed back to Paddington for one of the most pointless 450 mile round-trip day outs I've ever made. But I was £500 the richer so I wasn't complaining too much.

The second call I'd received was from Nick Dawson telling me that the duplicators were having trouble copying my tape as it was loading too fast for them to master properly. While I'd speeded up the loading time, in my opinion it wasn't that substantial an increase and I could read it perfectly well on my old battered tape recorder, but it seemed it was just too much for a sophisticated modern tape duplication plant to handle. I asked Nick if he wanted a slower version or maybe one that ran at the standard Spectrum loading speed but he said that they were going to give the tape to somebody to look at and that they'd sort it all out. I did wonder how they were going to handle the multi-load for the levels since the code for that was buried deep in the game but Activision seemed to know what they were doing so I left it at that.



Coverage of computer games in 1988 was a lot different than it is today, for the casual surfer going online meant dialing up a one-man-band Bulletin Board with a flakey 300 baud modem (or for those who could afford it, 2400 baud) but those boards had better things to do than host news and reviews of computer games, especially those from the 'cheaper' end of the market. What reporting there was on TV and radio tended to focus on the social effects rather than the merits of the games themselves and it was a long time between *Commercial Breaks*, a BBC documentary that covered the operation (and fall) of Imagine Software in 1984 and the Dominik Diamond\Patrick Moore hosted Channel 4 show *Games Master* in 1992 before the media decided that a

programme that actually reviewed computer games and showed people playing them might prove popular to viewers.

In the meantime the only real source of information about computer gaming were the specialist computer magazines and the shelves of UK newsagents were packed every month with titles that covered the computer industry as a whole down to those that catered for one specific model of computer. In the computer leisure market the four main titles that ruled the Spectrum roost were Computer and Video Games (aka CVG or C+VG), Crash, Your Sinclair and Sinclair User.

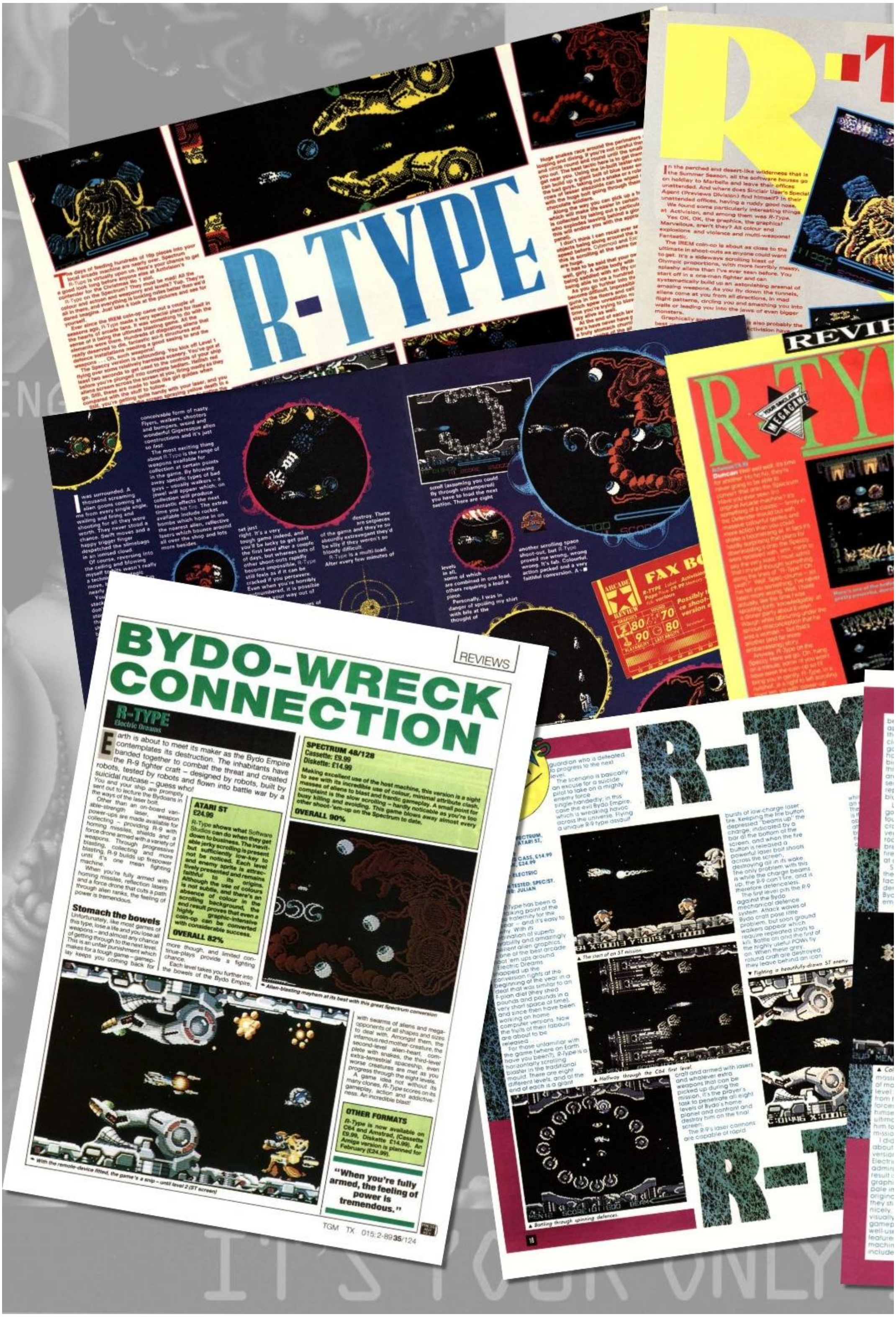
The monopoly magazines held on the way consumers obtained information about computer games led to some unsavoury practices at the time, rumours of links between the amount a company paid for advertising in particular magazines and how well their games then got reviewed in them were rife. There was a lot of competition between the magazines to be the first to review big new titles and again there would be whispers about deals done between software houses and magazine publishers for exclusive access to these titles - in exchange for favourable reporting.

The holy grail was to have your game featured on the front cover of a magazine which always then carried a glowing review inside and of course no magazine ever ran a review saying the game that was splashed all over their front page was a load of rubbish. The funny thing was that this front cover often featured original artwork or graphic layout that had to have been commissioned months before publication, at a time while the games were still being written. How exactly the magazines knew they were on to a future winner every time I just don't know but more than one magazine managed to print high scoring reviews of games that when you actually sat down and played them were just plain awful - which begged the question just what was going on sometimes?

I say this because when you read the following quotes and scores that the magazines gave the Spectrum version of *R-Type* I would like to think that the reviewers meant every word they said, but since that time I've had a few run-ins with magazines and now know that's not always the case. Those of you with a long memory may remember what I wrote in the Introduction of this book so please take everything that follows with a large pinch of salt.

As I mentioned before, Activision had started to whip up interest in the game months before with previews of the game in several of the magazines. This had been preceded by some general news about the game, usually accompanied by screen shots of the arcade version. Your Sinclair was probably the first to concentrate on the Spectrum version with a half page story illustrated with a couple of blurry photos. Sinclair User previewed the game in a two page spread which was mainly screenshots of Level One and then did the same thing again two issues later. In that later issue Sinclair User ran yet another two page spread this time concentrating on information they had

Overleaf: the media have their say.



BYDO-WRECK CONNECTION

REVIEWS

R-TYPE
Electric Dreams

Earth is about to meet its maker as the Bydo Empire contemplates its destruction. The inhabitants have banded together to combat the threat and created the R-9 fighter craft – designed by robots, built by robots, tested by robots and flown into battle war by a suicidal nutcase – guess who!

You and your ship are promptly sent out to lecture the Bydoans in the ways of the laser bolt.

Other than an on-board variable-strength laser, weapon power-ups are made available for collecting – providing R-9 with homing missiles, shields and a force drone armed with a variety of weapons. Through progressive blasting, collecting and more blasting, R-9 builds up firepower until it's one mean fighting machine.

When you're fully armed with homing missiles, reflection lasers and a force drone that cuts a path through alien ranks, the feeling of power is tremendous.

Stomach the bowels

Unfortunately, like most games of this type, lose a life and you lose all weapons – and almost any chance of getting through to the next level. This is an unfair punishment which makes for a tough game – gameplay keeps you coming back for more though, and limited continue-plays provide a fighting chance.

Each level takes you further into the bowels of the Bydo Empire, with swarms of aliens and mega-opponents of all shapes and sizes to deal with. Amongst them, the infamous red mother-creature, the second-level alien-heart, complete with snakes, the third-level extra-terrestrial spaceship, even worse creatures are met as you progress through the eight levels.

A game idea not without its many clones, R-Type scores on its gameplay, action and addictiveness. An incredible blast!

ATARI ST
£24.99

R-Type **shows what** Software Studios **can do when they get down to business. The inevitable jerky scrolling is present but sufficiently low-key to not be noticed. Each level and enemy sprite is attractively presented and remains faithful to its origins. Although the use of colours is not subtle, and there's an absence of colour in the scrolling background, the end result proves that even a highly graphic-intensive coin-op can be converted with considerable success.**

OVERALL 82%

SPECTRUM 48/128
Cassette: £9.99
Diskette: £14.99

Making excellent use of the host machine, this version is a sight to see with its incredible use of colour, minimal attribute clash, masses of aliens to blast and frantic gameplay. A small point for complaint is the slow scrolling – hardly noticable as you're too busy killing and avoiding. This game blows away almost every other shoot-'em-up on the Spectrum to date.

OVERALL 90%

▸ Alien-blasting mayhem at its best with this great Spectrum conversion

▸ With the remote-device fitted, the game's a snip – until level 2 (ST screen)

OTHER FORMATS

R-Type is now available on C64 and Amstrad, (Cassette £9.99, Diskette £14.99). An Amiga version is planned for February (£24.99).

"When you're fully armed, the feeling of power is tremendous."


BLUE PRINT
R-TYPE
SPEED:
CONTROL:
GRAPHICS:
PROGRAMMERS:
Britain's best selling computer magazine NOV 88 £1.20
COMPUTER
+video
GAMES
FREE SPECCY & 64 DEMOS
R-TYPE
Master Blaster!
R-CADE R-TISTRY
R-TYPE
Reviews
SCREEN TEST
ACE
GRAPHICS 7 8
SOUND 4 6
VALUE 8 7
PLAYABILITY 9 9
OVERALL 93% 85%
UPDATE . . .
As usual, the Amiga version should be the most impressive of the lot – although it doesn't look as though it'll be around until Christmas.
The C64 incarnation looks alright, but lacks the polish of other games such as Armalyte. Still, it's a reasonable enough rendition of the arcade game to warrant looking out for.
Work is well underway for the Amstrad version, which should be more colourful than the Spectrum version, and play just as well.
R-TYPE
PREDICTED INTEREST CURVE

gained from visiting us in Fareham a few months before. It seemed that someone at Sinclair User really liked us even though some of the screenshots contained graphics that never made it into the final version.

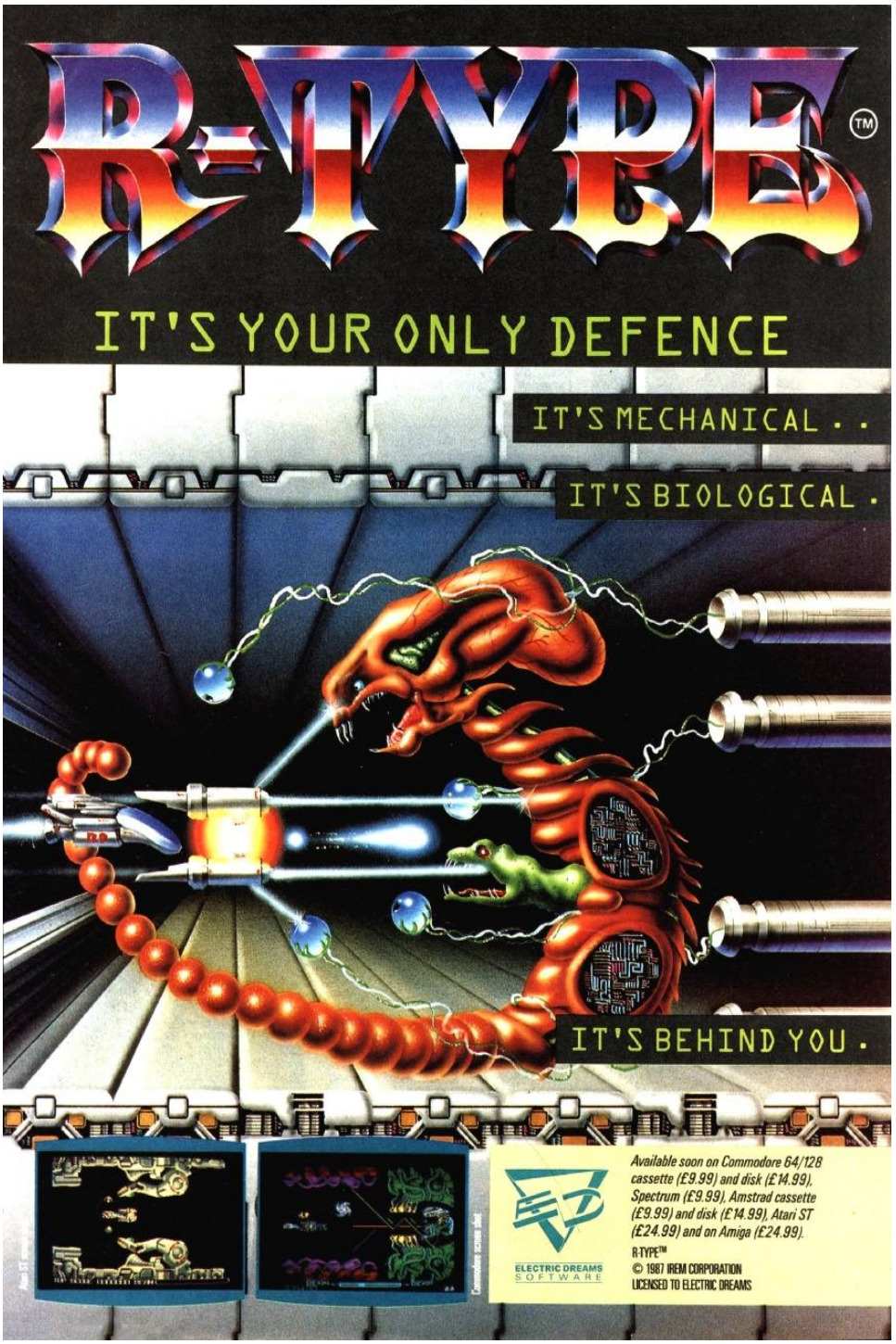
Spectrum *R-Type* went on sale late November 1988 in a large cardboard box for £9.99. Despite the size of the box - a blatant attempt to convince the buyer they were getting something special for their money - the contents comprised nothing more than an instruction manual printed in several different languages, a cassette tape containing the game and a plastic insert to hold the cassette safe. Activision and other software companies would receive a lot of criticism for the size of their packaging from retailers as they would, rightly, complain that these large boxes were taking up shelf space that could be used to display more copies of normal sized cassette cased games instead. I don't know how this affected sales of *R-Type* and other similarly packaged games but logic would suggest that a retailer with limited shop space would not be too keen to fill up what little room was available with multiple copies of just one or two titles.

With the game going on general release all the magazines were now free to print their reviews, even those that had not been looked on with favour by Activision. The way magazines, and computer magazines in particular, number and date their issues is one of those great mysteries which means that none of the following dates have any meaning really and are included just for completeness sake.

The November 1988 edition of C+VG had *R-Type* as the front cover game along with the free cassette containing both the Spectrum demo I had produced and the C64 version from Dave. Inside over three pages the review concentrated more on the ST version (along with some ST screenshots that clearly show Karl's development debugging information on screen.) The Spectrum version is given an overall score of 93% which, seeing how the magazine had given four out of ten for the sound, is pretty hard to understand especially as the ST version scores higher individual marks yet receives an overall score of 85% Dave's C64 version gets barely a mention - and isn't scored - even though there are more screenshots of it than the Spectrum version in the review. Information is very thin on the ground and the whole thing comes across as a slightly expanded preview rather than a review.

Crash ran a two page review giving the game an overall percentage of 92% and awarding it 'A Crash Smash' in their 1988 Crash Christmas Special. Like the C+VG review they marked the sound down (at 60%) again begging the question how come that can result in an overall score that is only 8% short of perfection? For a change it looks like the three people who reviewed the game actually did play it as there's much less of the usual press release text this time. The game is summed up as "*a surprisingly good conversion of the classic coin-op*". Classic?

Sinclair User ran another two page spread in their December 1988 Issue, this time an actual review even though it contained less actual text



"It's behind you" proclaimed the adverts, "Oh no it isn't" shouted back the Christmas panto crowds. It was still your only defence though.

than in their previous previews. What is printed is pretty meager, it's quite generic and contains very little that is specific to the Spectrum and it looks like somebody just dusted off one of the three previous previews and did a quick rewrite. It does get an overall score of 90% though and a quote that, if it were a play, would definitely be running outside the theatre... "*Possibly the best space shoot-em-up conversion ever!*"

Your Sinclair's January 1989 review is a single page, and once you get past the first few paragraphs of drivel that have nothing to do with the game it turns out to be a pretty good overview of the game. Identifying some of the game's strong points - the learning curve, the weapon systems, the limited play-ons - it's one of those reviews that you feel no matter what the final score, the reviewer has actually played the game and is telling it like it is. The game scores an overall nine out of ten with the comment "*as faithful a conversion as one could hope for.*"

ACE magazine was another magazine that gave Spectrum *R-Type* a two page review in their December 1988 issue but again like most of the others it was mainly a rewrite of the instructions that came with the game and a couple of sentences of observation thrown in to personalise it a little. ACE must be applauded though for the quality of their printing, the screenshots come across as colourful and detailed and the whole presentation is top notch. The other versions get a "*coming soon*" mention as well and the game itself gets an overall score of 871 out of 1000 despite scoring a lowly score of one for 'IQ Factor'. Just to be different ACE would also give a score for how close a game was to its arcade original, which here was nine out of ten and the comment "*A superb conversion, with little (if anything) missing from the game.*"

The Games Machine, an all-formats magazine similar to ACE was yet one more 'review' that for all intents and purposes just reprinted the instruction manual, though readers would have to wait until the February 1989 edition to read it. As I said about publication dates before this could actually have been published any time as some magazines would often pre-date their editions to give the impression they were ahead of everybody else with news and reviews. Despite being alone in correctly identifying the slow scrolling of the game at times it does award the Spectrum version a score of 90% while the ST version gets 82% and another quote guaranteed to increase sales .... "*This game blows away almost every other shoot-em-up on the Spectrum to date.*"

To paraphrase the American journalist H.L. Mencken: "*no one ever went broke underestimating the intelligence of the British computer game magazine buying public*" so what was written in these reviews, and the way it was written, just passed by unnoticed. I certainly wasn't complaining as it seemed like all those months working on the game to get it looking and playing as good as possible had paid off. Now it was up to the consumers to decide if they wanted to part with their hard earned cash and buy my game or not.

The Golden Joystick Awards a.k.a. The People's Gaming Awards is, according to the publicity put out by Future Publishing Ltd. "*the oldest gaming awards ceremony in the world and the most prestigious - voted for by the only people who truly matter - gamers themselves*". In 1988 however it was a little different. The C+VG helmed Golden Joystick Awards was a 100% industry ceremony with not a member of the public in sight. Held early December as part of the annual Industry Dinner colloquially knows as InDin - a night out for anyone connected to selling, producing or supplying games (but most definitely NOT creating them, coders were barred) - the event was one huge corporate backslapping exercise. Publicly it was a quiet affair, InDins never got much coverage in the retail computer magazines, being seen as pretty much the equivalent of the Christmas office party, any coverage of the event was left to trade publications such as Computer Trade Weekly (CTW). Despite the impression given today, back in the late 80's The Golden Joystick Awards were totally industry driven, with nominees decided and voted on by the staff members at Computer and Video Games Magazine only. And while they may have wanted to foster the impression that the readers of said magazine had some kind of say in the matter the fact that so many games launched onto the Christmas market ended up being lauded and applauded at the InDin just weeks after being released made it a hard claim to swallow.

When I opened my copy of CTW barely two weeks into January 1989 and saw a picture of a smiling Rod Cousens being flanked by 80's comedians Hale and Pace and clutching a Golden Joystick Award for Spectrum *R-Type* I thought it was a mistake that no one had been in touch to let me know my game had won.

Two comedians….and Norman Pace (only kidding Mr Cousens lawyers!) Rod picks up 'his' award for Spectrum R-Type.

I called Mark Jones and asked him if he knew anything about it but he was as much in the dark as I was, not only hadn't we been told our game had won but no one had bothered to let us know we had even been put up for the award in the first place.

Speaking to Activision later it was made clear to me that while my game had won the Award (for *Best Translation of a Licence (8 bit)*) there was no way I would be getting my hands on it since in fact I hadn't actually won it, Activision had, for doing such a great job. So the next time you see a film being given an award or statuette for being the best of the best then please remember that the director, scriptwriter, actors, crew and editor were not responsible for its success and really had nothing to do with it, No, that's all down the guy who signed the cheque in the first place. Apparently.

There was another reason Activision didn't want to part with a gold-paint covered plastic joystick... it was the first they had ever won and they wanted to show it off in their new office - Activision had moved. While I had been busy getting on with *Tusker* for System 3 Activision had shut down their Southampton office and relocated to somewhere equally as glamorous, they were now resident in sunny Reading. Based on the outskirts of Reading - and probably as far from the city centre you could get while still being inside the city boundary - the new office was on the first floor of a two-story anonymous glass and steel building that had more in common with a call centre than a top-flight software house. Keeping the open plan strategy of Southampton the office was all wall dividers and cubicles for the most part with a couple of closed off rooms huddled together at one end. Sticking out like a sore thumb was the full sized *Afterburner* arcade machine, last seen at a swank hotel in Mayfair, and now on permanent free-play as it awaited it's very own home computer conversion with graphics by Mark Jones. There was lots of floor space and many cubby holes flanked with large panels you could hide in if you wanted to. At a later date I did just that, crouching down behind one of these so as not to be seen by a certain person, but that's another story.

Climbing the stairs from outside brought you into the reception area where for a few months, in an almost bare glass cabinet, a lone Spectrum *R-Type* Golden Joystick award took pride of place. At some point the actual joystick disappeared - probably returned to be awarded to somebody else the following year - replaced with a simple framed certificate hung on a nearby wall divider.

I paid a visit to Reading sooner than I expected as just a few weeks after the InDin ceremony the postman delivered an invitation for me to attend the First Annual Activision Awards to be hosted by Rod Cousens at his prestigious new office. I didn't know if it was worth going or not, especially so soon after the Golden Joystick let-down but the invitation seemed to be offering a kind of apology, perhaps even a nod of recognition, at the least free food and drink. It was that last bit that clinched it for me.

I think the reason you may not have heard of the Annual Activision Awards is the same reason there was never a second Annual Activision Award - putting thirty-plus game coders into a single room and letting them drink as much alcohol as they want is inevitably going to lead to trouble. And it started off so well.

As we milled around with our plastic cups of wine and paper plates of nibbles there was a quiet undercurrent of one-upmanship, each of us checking out the others to see what they'd done, what they were doing and what they were going to be doing. Since *R-Type* had been received so well I now felt I didn't have anything to prove so was quite relaxed, unlike my *Rampage* days when I was definitely the new kid on the block I now felt I had earned my entry into the club. I knew a few of the other coders by name or reputation and a lot of small-talk about games and computers was going on.

When Rod Cousens stood on a milk crate covered in orange crepe paper (he isn't the tallest guy around) and started handing out the awards it became apparent that, in the words of Hot Chocolate, "*Everyone's a Winner Baby*" as it seemed that all of us who had been invited got to take something or other home. I stood up and accepted '*The Electric Dreams Excellence in Software Award for R-Type Program Conversion, Spectrum*', a gold painted cassette mounted in a glass and metal presentation box on a red velvety background.

What seemed like a never-ending mix of golden cassettes and glass framed certificates were handed out for seemingly the most minor of things while more wine and beer was consumed and the crepe paper started coming unstuck from the milk crate giving a faintly tatty air to the proceedings. Then we all went and stood on the stairs and had a group photo taken - as seemed mandatory for all software houses in the 80s - Activision's finest talent on display.

I don't know if it was meant to be a morale booster, free publicity, a genuine thank you or some kind of ego trip for Cousens but with nothing left to do but drink and wait for a go on the *Afterburner* machine things started going downhill. Why some were given awards on which quite a lot of money had obviously been spent while others looked like they'd been picked up in a Pound Shop caused a bit of friction, as did the fact that a few of the programmers there hadn't actually been invited but had tagged along with others and, having worked for Activision in the past, were wondering why they weren't being recognized as well. The cheap glass-framed certificates also appeared to be rather fragile as I witnessed one shatter when it was accidentally stood on and a few others got broken barely an hour after being awarded. Empty wine cups and beer bottles soon started to litter the brand new office and here and there small pools of spilt wine soaked into the carpet. Every now and then there'd be loud shouting as alcohol-fuelled coders aired grievances, real or imaginary, and several heated arguments broke out.

What added to the depressing air settling over everything was that it was still early afternoon and a lot of us were trapped there. I couldn't leave

What do you reckon...a visit to Antiques Roadshow or Cash in the Attic? There's no actual tape in the cassette just a short piece of plastic leader.

because I could only travel off-peak on the train back to Swansea and outside the building there was nowhere to go but the busy main road. I finally gave up after one particularly drunk coder wanted to find Rod Cousens and tell him what he thought of him (Cousens had, in true Elvis fashion, left the building) and caught a taxi back to the train station. The fact that I would choose to kill time at Reading railway station rather than at Activision's office should give you some idea of how bad it all was.

It was a reader's letter in one of the magazines that first hinted there was something very wrong with the Spectrum version of *R-Type*, that if you managed to finish Level Seven rather than move on to Level Eight the game just kept looping around Level Seven. Feeling very protective of the game and wanting to prove that it was the reader who was at fault and not my code I loaded up one of the free copies of the game I'd been sent by

Activision, put in a Multiface invulnerability POKE and started playing through the game.

Finishing Level Seven the game prompted me to start the tape to load Level Eight, which it proceeded to do, and then....I was back at the start of Level Seven again. OK, this was wrong. Each level of *R-Type* that's loaded in from tape had a small header block that tells you the level number and the size of the code that follows, this allows for easy tape navigation and for making sure that the game does indeed load in the correct block of data. At first I thought that Level Eight had been corrupted in some way, some kind of code merging resulting in half of Level Seven ending up inside it, but the fact that you could play through all of Level Seven without encountering any corruption or sign of Level Eight made that very unlikely. After a quick investigation I found that the eighth level on the tape was a duplicate of the seventh level - byte for byte - except for one thing, though it was Level Seven in every respect the byte that held the number of the level in the header block wasn't set to seven, it was set to eight.

I knew at the time I hadn't made a mistake, but there was a little voice in the back of my mind saying "*are you REALLY sure?*" and I had to find out if only for my peace of mind. I pulled out the source code and support programs I had written and built a new master from scratch, made easy because of the small program I'd written to do it automatically for me. The master I created played through perfectly which made me think, if there was one place it could have been my fault then it would be when I loaded, compiled and saved off the levels manually one at a time - it was possible that instead of loading the Level Eight source code I had instead loaded Level Seven again and saved it off as Level Eight. But even if I had done that the program that took those levels, loaded them in then saved them out to tape as my new high-speed versions had the length of code to save hard-written into the program itself. So yes, Level Seven could have been saved out as Level Eight but the length of that block of data would have been different to the real Level Seven and half the graphics would be missing. Plus both levels on the tape were exactly the same size so it couldn't have been that.

You may think that when you create a game master, be it tape, CD, DVD or whatever, the reason you play through it from start to finish before sending it away to be duplicated is to make sure the game plays properly, but that's only half the story. The other reason is to make sure the media itself is in fully working order. During my time writing software I've seen cassette tapes that are all transparent leader, tape oxide that lifts right off the backing, CDs with pits in the aluminium sub-layer, sub-layers that run out before they reach the edge of the disc, scratches, marks and stains galore. And all of these on brand new out of the packet media. If you're going to be sending a master away for duplication then you have to make sure that before anything else they're going to be able to load what's on it properly which is why you ALWAYS test your final Master Copy. This is why I'm so certain that the version I

produced and sent off to Activision had a working Level Eight on it, not because I don't believe I'd make a silly mistake loading and saving but because there is absolutely no way that master tape went out my front door without every single piece of code on it being loaded and played through at least once.

Maybe the duplicators fouled up and pressed the wrong button on the tape machine? No, that couldn't have happened otherwise that second Level Seven would be an exact copy of the first down to the byte indicating the Level number, and that wasn't the case. There was only one possibility left, David Aubrey Jones, Mr. Speedlock himself.

Without getting into too much technical detail Speedlock was a commercial method of copyright protection created by David Looker and David Aubrey Jones and marketed by Speedlock Associates that relied on various methods of encryption and non-standard tape loading to produce games that made tape-to-tape copying difficult. Speedlock games would load with unusual ‘clicking’ or ‘bubbling’ sounds anywhere from twenty five to fifty percent faster than normal games and were meant to thwart hacking and copying methods.

Software houses loved Speedlock and similar schemes because it also meant they could cut down on the amount of tape they had to buy, faster loading equals less tape equals more profit. When Nick Dawson had told me the duplicators couldn't handle my customised tape loading routine and he would sort it out I hadn't realized he meant he would be getting the whole thing Speedlocked.

On a later game I saw David Aubrey Jones do his magic first-hand - or rather I didn't. Visiting his house I was taken to the room where he did all the work, which involved him first loading the complete Spectrum tape into an Atari ST. Whether he actually loaded in the raw bytes or just sampled the audio signal I don't know because when it came time for David to do his magic I was ushered out of the room and the door firmly closed behind me.

I sat outside while David did what he had to do and after a short time the door opened and he handed me a tape, my game now a new, protected Speedlock version. To be honest I expected Speedlock Associates to be more than just another back-bedroom business, but I wasn't aware of this until long after *R-Type*. Had David Aubrey Jones or someone at Speedlock somehow messed things up while protecting Spectrum *R-Type*? Not knowing the full procedure I can't say for certain but I do know that as I sat there a couple of years later and watched him load in another multi-level Spectrum game of mine there was no logging procedure and no questions asked about code length or start addresses.

I took a good look inside Spectrum *R-Type*, not the source code but the actual retail cassette version, I wanted to see just how my multi-load code had been Speedlocked. I don't know what Activision were told but comparing my original code against the version on sale revealed some interesting points, mainly that the levels themselves had

no Speedlock protection on them whatsoever. My tape loading code had been hacked and two sections of it had been rewritten, the first changing the colour of the loading bars to something more in keeping with the Speedlock look and the second a change to the critical timing values that controlled how fast the data loaded to decrease the speed slightly. To load the code IN at a different speed means of course that at some point you have to save it OUT at that speed, which adds further to the suggestion that perhaps someone at Speedlock made a mistake when recreating the Levels. If they were messing around with my code and rebuilding the Levels then it makes it more likely that somebody somewhere made a simple mistake and put the wrong number into the wrong address with unfortunate consequences.

Despite it being a seemingly catastrophic error there was very little negative fallout from people who'd bought the game, perhaps because so few of them managed to get as far as Level Eight in the game, or were likely to. Activision offered to replace the bugged versions with new ones which had me wondering where they got a proper version from as they most definitely didn't ask me to provide any kind of replacement that had a working Level Eight. The same thought occurred to me when the game was released on a budget label just a few months later, obviously someone at Activision or Speedlock had a complete version of the game which convinced me once and for all that the original problem was indeed down to someone at Speedlock.

R-Type had had its fifteen minutes of fame in the spotlight, the next round of game releases were being previewed and reviewed in the magazines and I was putting all my effort into coding *Tusker* for System 3. Karl's ST version had been a success and could now be found helping to sell the Atari computer itself, being one of the lead titles on an incredible bundle deal that saw twenty top class games given away with every new ST as part of a 'Powerpack' deal. This deal came in for a lot of flak from computer retailers as the quality of the free games was such that it meant a new buyer wouldn't have to pay a visit to his local software shop for quite a while. To put it into perspective imagine going to the shops to buy the latest incarnation of Playstation or Xbox and finding it came with twenty free games that were being sold for full price on the shelves alongside it. No wonder a lot of retailers were getting annoyed!

As I've mentioned, the original Commodore 64 version was canned by Activision and eventually brought to market by *Katakis* developers Rainbow Arts. While this version was released on time it doesn't stand up well against Dave and Jim's original with several iconic sections of the game being lazily rewritten to accommodate the limitations of the hardware and programming. The rotating Alien Wheel of Level One doesn't rotate, the Level Three Big Ship is missing 90% of its guns, there's no Boss Alien at the end of Level Five, the Pistons that move around Level Six look more like

cabbages and so on. Manfred Trenz of Rainbow Arts has said that they only had seven weeks to do the job and it's by no means a bad conversion, it looks more colourful and plays a bit faster compared to the unreleased version, but at times you can see the limitations imposed on it by the short deadline.

With Dave and Jim's original version now available to download and play for yourself you can do your own comparison review and what is immediately obvious is just how much of the game had actually been completed. I said this before and I think it's worth repeating that you do have to wonder if they were given those seven weeks and told they too could bypass some of the trickier parts of the game whether they could have finished it in time. From seeing the state of the game when I left Fareham compared to when I visited Activision just a few weeks later I honestly think they could have done it.

The Amstrad version was seen as a bit of a cop-out in that it was not written from scratch but was a port of the Spectrum version. Comments posted online such as "*awful*" and "*the worst version ever*" smack a little of elitist snobbery and are very unfair on Keith Goodyer, though it is often accompanied by a begrudging nod to how close it is to the arcade version. While no one expected Activision to release the game with a sticker on the box proclaiming "*Coded In Only Three Weeks So Give Us A Break!"* what is often forgotten is that without these quick and dirty ports the number of major titles released for Amstrad computers would have been a lot less.

At the time *R-Type* was released the sixteen bit computer wars were in full swing, software houses supported either the Atari ST or the Commodore Amiga but seldom both and Activision were firmly in the ST camp, which is why the release of an Amiga version of *R-Type* shortly afterwards came a bit out of left field, However a quick scan of the game credits soon made everything clear..... "*a Factor 5 release via Rainbow Arts with thanks to Rob Hylands and Karl Jeffrey*." Activision had decided to do another port and handed over the ST code to Rainbow Arts, again making you wonder just what kind of deal had been done (and when) in negotiating the Commodore 64 version. Despite some rather unusual title music the game itself is very close indeed to the arcade original though again it suffers in a few places from hardware limitations; the Big Ship in Level Three has the screen all to itself with no sign of background and a few parallax screens are missing. However the in-game music is a pretty good copy of the arcade version and the whole thing plays very smoothly indeed.

I doubt whether Activision still have accounting details dating back to 1988 and even if they did I wouldn't be allowed access to them so I can't say if the Spectrum version of *R-Type* was a financial success or not, the same goes for the weekly game charts as printed in the trade publication CTW. The only chart information I have available is what was printed in New Computer Express, a weekly magazine that carried a Games Top Twenty as supplied by Gallup which shows

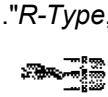
Spectrum *R-Type* entering the Top Twenty at number 11 on Dec 10th 1988 where it stayed for a second week. From this point onwards Gallup classified the game as being on all formats so it's not possible to say if the sales are purely Spectrum based or a total of all the computer versions. On December 24th *R-Type* moved up to the number 8 position where it stayed for three weeks before creeping up to Number 7 a week later. By mid January the game had dropped to 14th place before rallying again on February 11th and climbing back up the charts to number 6. From there it bounced around number 14 for a few weeks before disappearing from the chart entirely.

R-Type reappeared in March 1990 as part of a four game compilation package called *'The Biz'* alongside *Operation Wolf*, *Double Dragon* and *Batman The Caped Crusader* from Ocean Software and retailing for £14.99. The game had been remastered so it came with the missing Level Eight but I had no involvement with putting this together. In December later that year and just in time for Christmas the Spectrum version was released for the third time on the budget 'Hit Squad' label. Retailing for £2.99 it had shed the oversize cardboard box to appear as a normal cassette release and it too came with the proper Level Eight, again I had nothing to do with this.

By the end of 1992 the market for Spectrum games was dead, along with all the other eight bit home computers, and software houses had moved on to producing games exclusively for the sixteen bit Amiga and Atari. But even those were under threat from the encroaching consoles of SEGA and Nintendo leaving Spectrum *R-Type*, like pretty much every other game from that era, to just slowly fade away into memory.

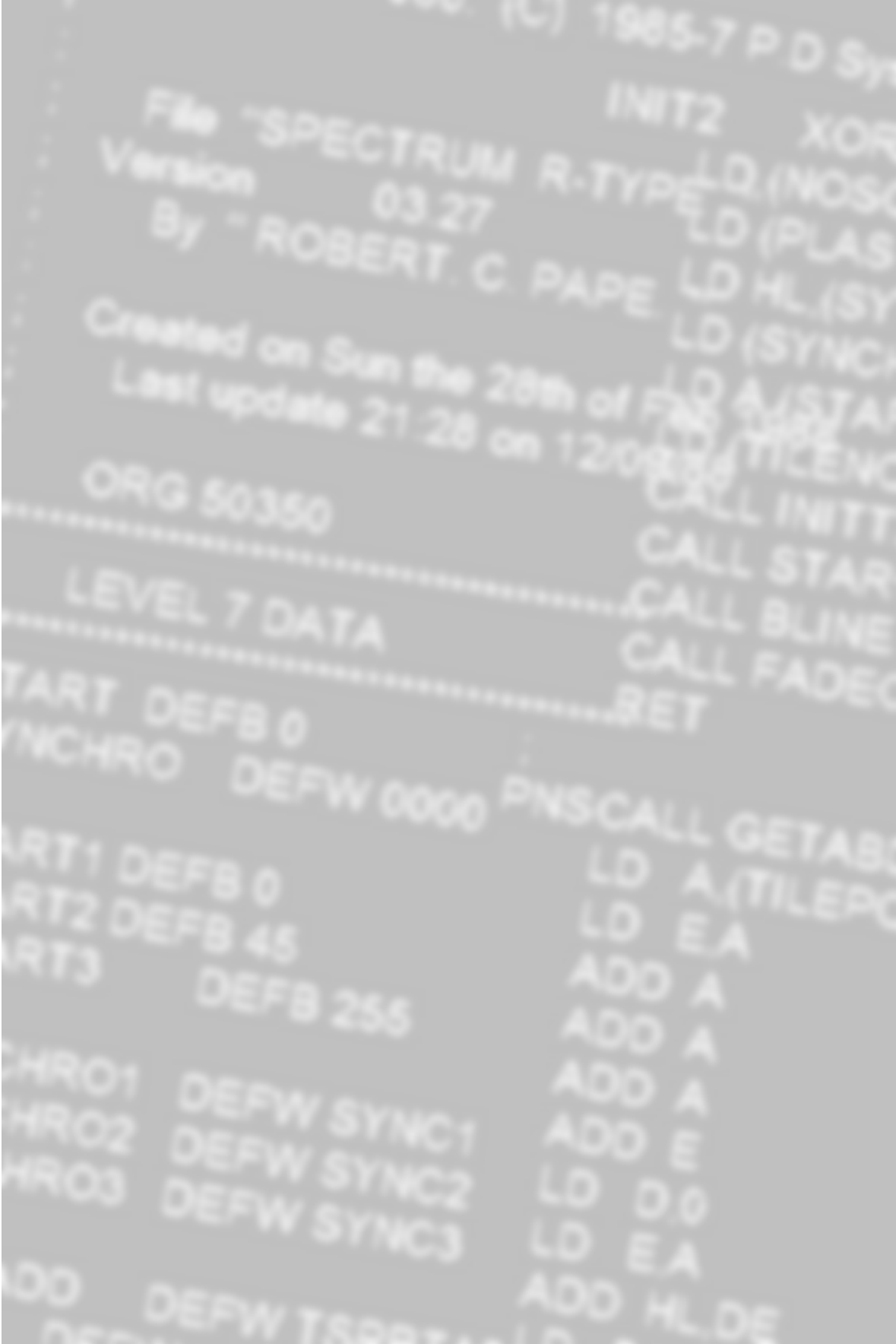
Over the years the Spectrum version has gained somewhat of a reputation for being one of the best eight bit shoot-em-ups ever produced but to be honest that's a bit like saying singer Limahl from the 80s band Kajagoogoo had the best mullet hairstyle there ever was - unless you care about it it doesn't really mean much, especially in the twenty-first century. In 1993 the readers of Your Sinclair magazine voted for their Top 100 Spectrum games of all time and *R-Type* made it into third position, deservedly beaten out by versions of arcade hits *Chase H.Q.* and *Rainbow Islands* - and that was pretty much the end of that.

The game gets a mention now and again online or in print as an example of what the Spectrum was capable of or how well it rates against other versions of the game (or how badly) and it often crops up in one of those perennial heated arguments about which was the best home computer of that age, usually held between ZX Spectrum and Commodore 64 owners. Now and again when I'm talking to people who know I used to program games I mention *R-Type* amongst the titles I had published but more and more I get the same reply as I did when I told people the first time what I was working on….."*R-Type*, *what's that?"*



Some people messing around in a garden in Fareham in 1988. The one on the far right is obviously the better looking of the group.

ENDINGS


(C) 1985-7 P D Sy
INIT2 XOR
File "SPECTRUM R-TYPE
Version 03.27
By " ROBERT C PAPE
Created on Sun the 28th of
Last update 21:28 on 12/08
ORG 50350
LEVEL 7 DATA
TART DEFB 0
YNCHRO DEFW 0000
RT1 DEFB 0
RT2 DEFB 45
RT3 DEFB 255
HRO1 DEFW SYNC1
HRO2 DEFW SYNC2
HRO3 DEFW SYNC3
DD DEFW TSRB
LD (NOSC
LD (PLAS
LD HL,(SY
LD (SYNC
CALL INITT
CALL STAR
CALL BLINE
CALL FADEC
RET
PNSCALL GETABS
LD A,(TILEPO
LD E,A
ADD A
ADD A
ADD A
ADD E
LD D,0
LD E,A
ADD HL,DE

There was a last bad joke to be told, one that had its roots buried deep. It was a Sunday in April 1989 when I received a phone call from Her Majesty's Revenue and Customs asking me why I had not paid any National Insurance since 1987? I explained that I had been employed since early 1987 by a company called Designmaker, that they had paid me properly and that I had payslips showing my monthly wage, National Insurance and tax deductions. HMRC informed me that they had no record whatsoever of my employment status, as far as they were aware I had dropped off their radar two years ago and they were chasing after me to find out what was going on. They said they would look into it further but I could expect a call from the Inland Revenue at some time as they were checking up on me as well. I didn't have to wait long, the very next day I received a call from my local tax office asking me pretty much the same question, why hadn't I paid any Income Tax for the last two years?

Trying to stay calm I went through the whole explanation again and was asked to come to and see them in person and to bring any payslips and documentation I had to verify my story. I contacted Andrew Parton my ex-Designmaker colleague and found that he'd had similar calls and was being chased for back tax and National Insurance contributions as well, but unfortunately he hadn't kept any of his old payslips so there was no way for him to prove he'd had deductions taken from his pay. What was obvious to both of us was that Designmaker\Catalyst Coders had been deducting tax and National Insurance from us but not passing it along to the government. Unfortunately while I could account for perhaps a year's worth of payments with payslips I had no proof for the time I was receiving money directly into the Halifax account from Catalyst, and just how was I going to persuade them that I now had an IBM PC in lieu of salary owed?

I gathered up all my old Designmaker payslips, pulled the payment details off my Halifax statements and wrote an explanatory letter stating that I had been given a PC and Development System as payment and stated the current value of the hardware. I went to the tax office, put everything on the table and said "*OK, what do I owe you?*"

The amateur bodge-up of an agreement I had got Wainwright to sign finally proved its worth as not only could I show that I had received the PC and PDS hardware in lieu of pay but that thanks to a throwaway line in the opening paragraphs about when I had started the game I could show I had actually been employed by Catalyst during this time. Fortunately for me they accepted that I was the innocent party in all of this and waived their demands for back tax and National Insurance agreeing that I wasn't liable and Andrew was also let off the hook, partially as a result of what I showed them that day. I did have to pay some tax on the PC hardware though but I was totally in the clear after that. I could have kept quiet about it but then I wouldn't have been able to show the agreement I made with Wainwright and prove I was an employee so it was a small price to pay in the end.

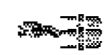
Was the fact that Tax and National Insurance deducted from our pay wasn't passed along to the government deliberate or accidental? I can't say for certain either way. I suppose it could have been an accident but you have to wonder what kind of procedures were in place and who exactly was running things to mess up on such a scale. I couldn't help thinking about Richard Kightley and his total lack of understanding of the games business, or business in general. If anyone could make a mistake like that then Kightley was your man but this had continued long after *Rampage* when I was being paid directly by Catalyst.

A conspiracy theorist could probably take several elements of this story - companies being set up in low employment areas, Government subsidies, Wainwright admitting ripping off programmers, money not being properly paid, Kightley not having a clue what was going on - and come up with any number of interesting explanations. I have my own ideas about what happened, I leave it to you to come up with yours.

If I was in the clear with the Inland Revenue and Her Majesty's Revenue and Customs then there was still the police to deal with. I was contacted by an officer from the Neath\Port Talbot Fraud Squad who wanted to interview me with regard to certain 'irregularities' concerning the funding Designmaker had received from the Welsh Development Agency. I was spoken to at home by two police officers and told them all I could remember of Kightley, Wainwright and the eighteen or so months I had spent working for them but I don't think I contributed much to their case, I knew nothing about the money or who was really in charge and of course I was a victim as well.

They particularly wanted to talk to Kightley but I had no idea where he was, even when we were shouting to each other over the phone I didn't know where he was calling from. Since I never read any headlines in the trade papers announcing arrests or a court case nor heard any more from the police I'm guessing there just wasn't enough evidence to warrant further action on their part - either that or everything really was honest and above board and I had just been unlucky to be caught in the middle!

And that was the final chapter in my involvement with *R-Type* for the ZX Spectrum, which seems as good a place as any to finish this account. Playing some of my old games again is quite disconcerting as it's like looking at somebody else's work, I just can't recall ever coding them (or how), but *R-Type* is like visiting an old friend and sharing the good times we had together. I can honestly say that of the seventeen or so games that I wrote in my career as a games programmer I never enjoyed myself more than when I worked on this version of *R-Type* and I do miss the time I spent in Fareham with Karl, Dave, Jim and Mark. I'm not one for long goodbyes so I'll just say thanks for coming along with me on this journey into my past and I hope you enjoyed reading about an old computer game as much as I did writing about it. Game Over.



WHERE ARE WE NOW?

Over the years people change, move around, get married, divorced and even die and keeping track of people has been difficult, even the ones I wanted to. What you end up with are whispers and rumours and brief mentions here and there so what follows is a by no means complete and far from accurate roundup of who's doing what now. I apologise if I get any of the details wrong, just blame it on my old age.

Catalyst Coders - The company limped along for a short while after *R-Type* was finished but was eventually dissolved in February 1991.

Nick Dawson - Currently acting as a technical advisor and game play evaluator for Jacqui Lyons at Marjacq Nick has been part of the games industry for almost twenty five years. His list of credits as a Producer is quite impressive.

Designmaker - Went bust, formally and legally in 1991. The carpet shop, taxi company and our tip of an office are now long gone, I don't think anybody misses them much.

(Dr.) Sedghat Hosseini - Seemed to just melt away as soon as the cash flow at Catalyst turned to a trickle. I may be doing him a disservice here but I never came across his name again and when I did ask, other people hadn't either. A man of mystery.

Rob Hylands - Went on to bigger and better things, writing games for the SEGA Megadrive then the ill-fated 32X add-on before moving into the production and management side of things with companies such as Take 2 and Codemasters. Has now left the games business and runs his own pub with the help of wife Lorna and is involved in local politics. May this particular Hylander live forever.

Karl Jeffrey - After Images Karl moved from above the kitchen shop apparently into the grounds of an asylum, still in Fareham and changed the company name to Climax Games. Still active and very successful in the games business today Karl is currently the CEO of The Climax Group and is based in Portsmouth.

Dave Jolliff - Hit a bit of a rough patch after the collapse of his version of *R-Type but* was still writing C64 games in 1992. We kept in touch for a while and I steered him toward a bit of work but then everything went quiet. Recently I learnt he had bounced back and is a Programming Consultant\ Programmer for a company specialising in bespoke training solutions. The whole *R-Type\Katakis* rewrite situation is one of the more well known legends amongst the C64 community and Dave is the man who has the definitive story to tell about it.

Mark Jones - Carried on producing graphics across a wide range of formats by day and working for the Post Office by night. I was lucky enough to work with him on a few more games and I consider him the best 2D Computer Graphic Artist the eight bit computers ever saw. Out of everything I ever did writing games I miss working with Mark the most. Eventually he quit the games business and is now working for the Post Office full time.

Richard Kightley - Laid low for a while after the Designmaker fiasco in a small caravan miles away from anywhere out in the woods but I soon heard rumblings that he was renting a house in Swansea and had a few novice coders living there with him trying to set up his own software company on the (very) cheap. Someone I know who was owed money by him also found out and beat me to the punch by calling the Inland Revenue, Welsh Development Agency, Customs and Excise, Swansea City Council and anyone else who would listen and ratted him out. Don't know what happened to him after that. Don't care.

Jacqui Lyons - I stayed with her and Marjacq up until the last commercial game I wrote. Whatever you may have heard about agents, Jacqui and her staff were always there for me and I feel very lucky indeed to have been associated with them for so long. She is still a computer and literary agent and probably shouting down the phone to someone over unpaid invoices even as you read this.

Saul Marchese - After leaving Activision Saul went freelance as a graphic artist and he helped me out on a SEGA game I later did before working overseas in Japan. Saul is still in the industry, his last credited game was *Ghost Rider* where he was Lead Environment Artist and which was produced by.....Climax!

Andrew Parton - After completing *Flying Shark* Andrew wrote *Maze Mania* for the C64. Then I believe he moved over to the Amiga and started writing games for Microprose but I haven't seen or spoken to him since 1989.

Jim Smart - Parted company with Dave Jolliff and wrote a few C64 games for Images and others. After that he programmed *Alfred Chicken* for the NES and then just seems to have dropped out of the business entirely. May have taken up a 'New Age' lifestyle.

David Wainwright - I used to hear the odd word here and there about what he was doing after Catalyst, which always seemed to be that he'd come up with some great idea and was trying to sell it to get himself up and running again. Recently I heard that he had got involved with some TV companies and made a name for himself running phone-in quiz shows and the like. Despite what I've written about him he was a nice guy at heart and he did give me my break into the business, if I met up with him today I'd buy him a pint and shake his hand - but I'd count my fingers afterwards.

Robert (Bob) Pape - After *R-Type* I wrote two more Spectrum games and then moved over to consoles, working on the Master System, Game Gear, original Gameboy, Colour Gameboy and a couple of Multimedia games on the PC. At the start of the new Millennium it was getting harder and harder for one-man bedroom coders like myself to compete with large established companies, and downright impossible to even consider doing that for newer consoles such as the X-Box and Playstation 2.

When the Gameboy Advance was released I approached a few people about writing games for it and them but unless you had an office staffed with artists, musicians, mappers, level designers, motion capture specialists and a tea lady called Doreen they just didn't want to know. I tried to hang on for a few years coming up with PC ideas but nothing took off and I didn't really want to move away to work as just another coder in an anonymous team somewhere so with a mortgage to pay and savings rapidly dwindling I had no choice but to look for proper employment instead.

I made a promise to a friend that I'd try and find a job where I wouldn't be spending most of the time on my own and in December 2003 ended up working as an usher at the Grand Theatre in Swansea - I went from the extreme of hardly seeing anyone to dealing with thousands of people a day! I did some stints backstage, appeared onstage a few times with the likes of Ken Dodd and others and got to meet a lot of interesting people. I can now name drop with the best of them - isn't that Joan Collins a nice lady when you talk to her?

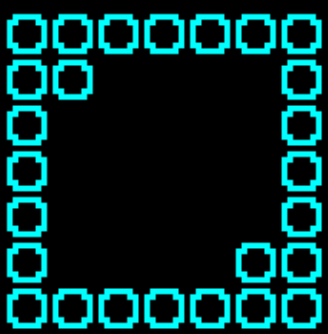
I progressed to Front of House Assistant at the theatre (basically the guy who does the stuff the managers don't want to) at the start of 2009 and managed to last about eight months before having enough of being treated like something you wipe off the bottom of your shoe and packing it all in.

More recently I've been working alongside adults with learning disabilities, using computers and IT to increase options and choices and hopefully make life a little easier for people who really need it.

I do a bit of coding now and again on the Spectrum and have written a few games just to see if I still had it in me but I actually get more enjoyment out of writing the support tools for them on the PC rather than the games themselves. What has given me the most satisfaction though is that I finally got around to completing what I started with *The Enhancer* all those years ago and wrote the graphic system I always wanted for it.

The future? Well there's something about the idea of loading up a copy of *The Quill*, *The Enhancer* and that graphic system I wrote and going back to the beginning to close the circle that appeals to me. *Return to Garstly Grange* anyone?

CONTINUE



PRESS FIRE BUTTON

CREDIT 0

In April 1982 Clive Sinclair launched the follow up to his successful ZX81 home computer, the ZX Spectrum.

In the ten years that followed over nine thousand commercial games were written for that iconic machine.

This is the story of one of them.