JON WELLS

Presents

SENSIBLE SOFTWARES SIDEWAYS SHOOT EM UP CONSTRUCTION KIT

SIDEWAYS S.E.U.C.K PROJECT

The Diary

By Jon Wells © 2008

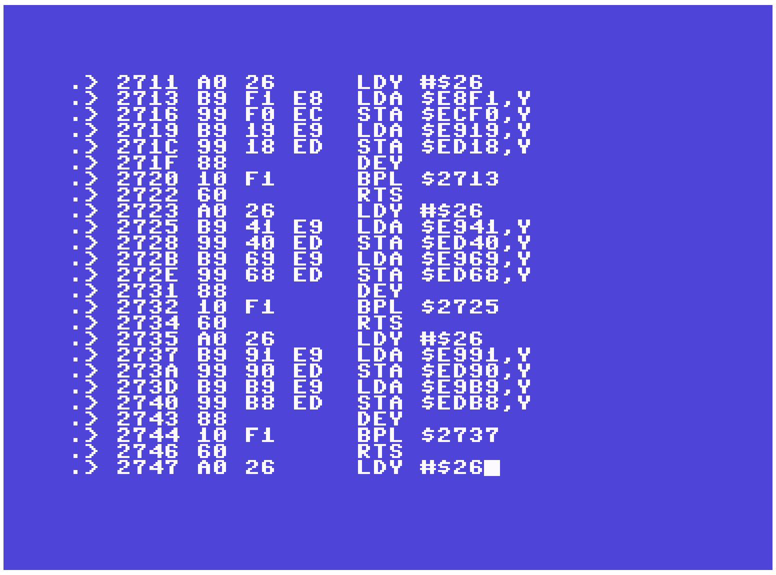
INTRODUCTION

This diary covers my pursuit into turning programming Legend Jon Hares classic Commodore 64 version of the vertically scrolling S.E.U.C.K (Shoot Em Up Construction Kit) into a fully-fledged sideways scrolling SEUCK! It's a project I've wanted to undertake for a while now, so I will be doing this over the coming months in my free time until it is complete. Once it is finished it will be available as a FREE download for everybody to use and enjoy!

Day 1 - Jan 5th 2008

It's been a while, the last thing I did on the 64 must be around 6 years ago so it's nice to be working on it again! Firstly before I even start any coding I need to completely refresh myself with the locations of the SEUCK scrolling source. Using the CCS64 emulator on my PC with the Action Replay cartridge .CRT file bolted into it I loaded up a game created with SEUCK. Firstly I started off loading my Breakthrough preview, which had a basic form of sideways scrolling. I needed to re-look at how I'd previously tackled the sideways issue, making lots of notes of the locations concerned. I noticed several flaws in my old routine so I looked at how I could improve on the method. Firstly, the old Breakthough character block scrolling routine consumed all the space of the original SEUCK character and colour scroll routines combined! Secondly, each block could not have a unique colour and it wasn't even compatible with the Map editor. These are issues I will vastly improve on.

The new source code that I will create will need to use the same style of coding so it links together perfectly with the editor. If I can create space in the existing source to hold the new adapted routines this will also be a great advantage. I need more location information so the next thing is to load a proper version of a SEUCK game that had the original vertical scrolling routines. Alf Yngve's excellent Twin Tigers would do just nicely. My first task here is to decipher all the locations of the dependent scrolling routine, this includes where the map data is located, the colour data, the 5x5 block data, how these are implemented and used in the scrolling routines etc. This is something that I ran out of time on, so decided to tackle it on my next attempt.



```
.> 2711 A0 26       LDY #$26
.> 2713 B9 F1 E8    LDA $E8F1,Y
.> 2716 99 F0 EC    STA $ECF0,Y
.> 2719 B9 19 E9    LDA $E919,Y
.> 271C 99 18 ED    STA $ED18,Y
.> 271F 88          DEY
.> 2720 10 F1       BPL $2713
.> 2722 60          RTS
.> 2723 A0 26       LDY #$26
.> 2725 B9 41 E9    LDA $E941,Y
.> 2728 99 40 ED    STA $ED40,Y
.> 272B B9 69 E9    LDA $E969,Y
.> 272E 99 68 ED    STA $ED68,Y
.> 2731 88          DEY
.> 2732 10 F1       BPL $2725
.> 2734 60          RTS
.> 2735 A0 26       LDY #$26
.> 2737 B9 91 E9    LDA $E991,Y
.> 273A 99 90 ED    STA $ED90,Y
.> 273D B9 B9 E9    LDA $E9B9,Y
.> 2740 99 B8 ED    STA $EDB8,Y
.> 2743 88          DEY
.> 2744 10 F1       BPL $2737
.> 2746 60          RTS
.> 2747 A0 26       LDY #$26
```

Part of the old BreakThrough scrolling source

Day 2 – Jan 19th 2008

Right, today I've got a couple of hours so I continue from where I left off last time. I have a go at finding the map data first, after about a half hour of searching I locate it starting from $0900-$19FF in memory. Now that I've found this I start looking at the source code which references that point to this area, it takes quite a bit of analysing and following the assembly instructions, but I eventually find that the byte locations $408f and $4090 hold the position of the low and high bits for the current position in the map data.

Each single number represents a strip of 8 – (5x5) blocks across the screen. With this information in hand I look even further into the source and locate that the block data is located between $1A00-$267F and the colour data is at $1900-$197F, so the Map data is actually held in memory between $0900-$18FF.

This information will help me immensely to locate every part of the scroller routines when I come back to this project, so everything is logged for next time.

Day 3 – Jan 26th 2008

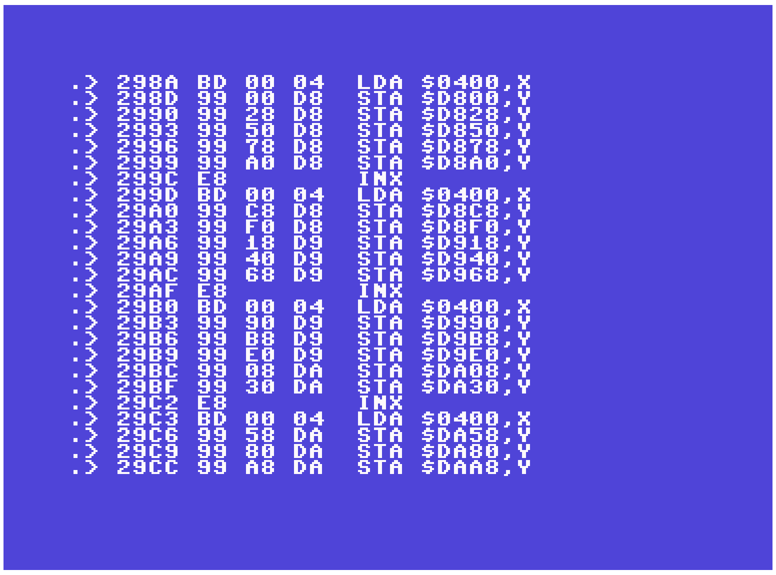
With the new information I re-load Twin Tigers and start the real tricky task of locating and understanding exactly how Legendary 64 programmer Jon Hare constructed the scrolling engine.

An obvious start is the location I originally worked on with Breakthrough, the original character and colour scroll source is from $2680 to 2C60, quite a large chunk of code. This is built up of two screen buffers that update the scroller a few lines at a time and copy the chars shifted onto the next screen buffer. These two buffers are screen locations $E800-$EBFF and $EC00-$EFFF. Looking back through this I quickly realised I could actually use the original character scroll routines and just change the location values so it works horizontally instead. This is an ideal situation as it doesn't take any extra space and uses the exact same instructions, only referenced to different address locations. This means no additional source changes are required! After spending some time changing the values I restart Twin Tigers. It still scrolls vertically in pixel form, but on every character scroll it shifts right, which is exactly what I was hoping for.

Next I looked at the colour routine, it's quite large, from $297B-$2C60. It uses a very smart way to scroll the colour, which doesn't actually scroll the entire screen. Because of the 5x5 blocks each being a single colour, it uses a technique where it only needs to scroll one row of colours for each of the 5x5 blocks down the screen. This means that at most it only ever needs to update a maximum of 6 character rows of colour instead of the full 25 rows that make up the screen. It's an excellent way of processor saving!

At this point I get quite excited as I come to the conclusion that I can re-write this routine for horizontal scrolling, but using a fraction of the memory size and still keeping the same processor time. It's thanks to the way in which the screen paging is mapped, it works out easier to do and the same result that Jon Hare did can be achieved horizontally in a smaller routine because of it! This is brilliant, because now I have over $280 hex bytes for any adapted routines that I need extra space for!

I run slightly over time so I back up what I've done and look forward to continue next time round, where the hard work really starts!



```
.> 298A BD 00 04  LDA $0400,X
.> 298D 99 00 D8  STA $D800,Y
.> 2990 99 28 D8  STA $D828,Y
.> 2993 99 50 D8  STA $D850,Y
.> 2996 99 78 D8  STA $D878,Y
.> 2999 99 A0 D8  STA $D8A0,Y
.> 299C E8        INX
.> 299D BD 00 04  LDA $0400,X
.> 29A0 99 C8 D8  STA $D8C8,Y
.> 29A3 99 F0 D8  STA $D8F0,Y
.> 29A6 99 18 D9  STA $D918,Y
.> 29A9 99 40 D9  STA $D940,Y
.> 29AC 99 68 D9  STA $D968,Y
.> 29AF E8        INX
.> 29B0 BD 00 04  LDA $0400,X
.> 29B3 99 90 D9  STA $D990,Y
.> 29B6 99 B8 D9  STA $D9B8,Y
.> 29B9 99 E0 D9  STA $D9E0,Y
.> 29BC 99 08 DA  STA $DA08,Y
.> 29BF 99 30 DA  STA $DA30,Y
.> 29C2 E8        INX
.> 29C3 BD 00 04  LDA $0400,X
.> 29C6 99 58 DA  STA $DA58,Y
.> 29C9 99 80 DA  STA $DA80,Y
.> 29CC 99 A8 DA  STA $DAA8,Y
```

The new colour scroll takes up much less memory

Day 4 – Feb 3rd 2008

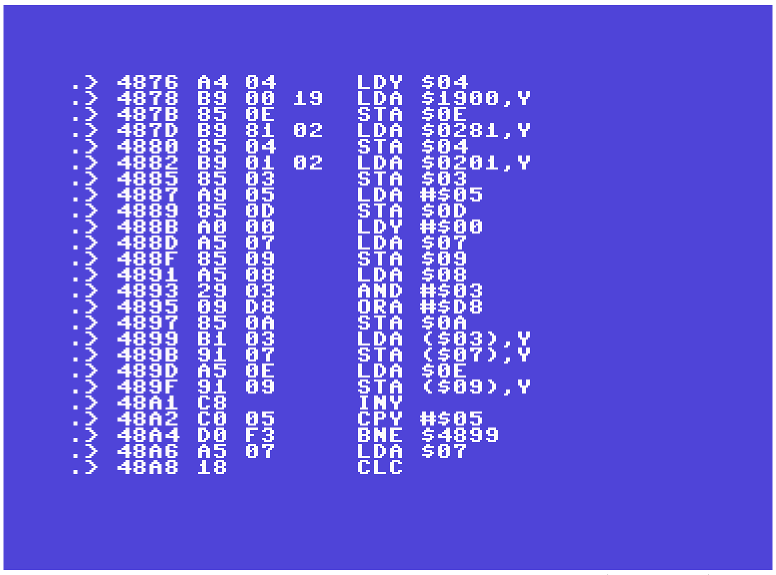
Yep, this is where the really hard work comes in now! So far the main character and colour scroll routines have been fairly easy to understand and modify, albeit time consuming. The main routines that handle the actual map, block and row update is next and these routines prove to be much more of a challenge.

After spending some time scanning through the main part of the engine between $4000 and $6500 I identify a segment of routines between $4604 - $49E0. This appears to be the main area that holds all the code that handles the pixel scrolling, including all the map, block and row updating. They're all written using some extremely compact and clever code that is making it quite tricky to follow.

After quite some time analysing and following through each instruction, I write down my findings and work out what is required to convert this to work horizontally. To make life easier I'm going to create the left scrolling version first (opposite of Breakthrough). That direction works with the existing values in the same way as the original. Later once complete I can modify these values to work in the more popular right scrolling fashion.

Each map row of 8 blocks that originally updated along the top of the screen need to be changed so they update vertically down the left hand side in order to make this work horizontally. I spend some time working out and testing the main formula to handle this change. Following that I also make the same modifications so the colour scroll works in the same way, it takes quite some time to rework this into the main scroll routine between $4604 -$49e0, and I use some of the spare memory I freed up from the colour scroll between $297b - $2c60. Once I get to a stage where I feel it should work, I save everything and start the game to test it out for real….

YES! It's scrolling the blocks 5x5 horizontally, but the colour and blocks don't seem to be matching correctly, at least it is working to some extent though which is promising. I decide to finish the lengthy session at this point feeling quite pleased with the progress. I'll tackle the bugs on my next attempt.



```
.> 4876 A4 04       LDY $04
.> 4878 B9 00 19    LDA $1900,Y
.> 487B 85 0E       STA $0E
.> 487D B9 81 02    LDA $0281,Y
.> 4880 85 04       STA $04
.> 4882 B9 01 02    LDA $0201,Y
.> 4885 85 03       STA $03
.> 4887 A9 05       LDA #$05
.> 4889 85 0D       STA $0D
.> 488B A0 00       LDY #$00
.> 488D A5 07       LDA $07
.> 488F 85 09       STA $09
.> 4891 A5 08       LDA $08
.> 4893 29 03       AND #$03
.> 4895 09 D8       ORA #$D8
.> 4897 85 0A       STA $0A
.> 4899 B1 03       LDA ($03),Y
.> 489B 91 07       STA ($07),Y
.> 489D A5 0E       LDA $0E
.> 489F 91 09       STA ($09),Y
.> 48A1 C8          INY
.> 48A2 C0 05       CPY #$05
.> 48A4 D0 F3       BNE $4899
.> 48A6 A5 07       LDA $07
.> 48A8 18          CLC
```

Part of the original screen scroll update routine between $4604 to $49E0.

Day 5 – Feb 16th 2008

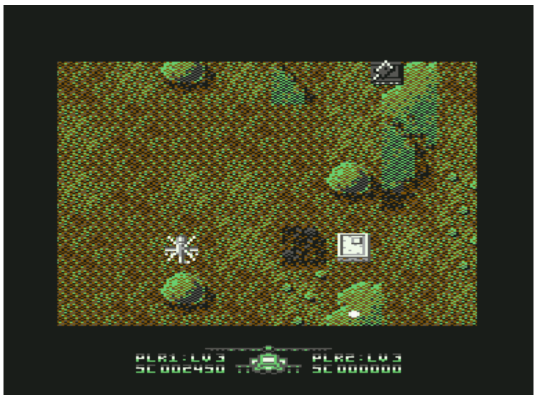
In the last session the updating of the blocks and colour were not working correctly, so after a quick refresh and another look at how the scroller was behaving I started to go over my code. I soon found that both these bugs were relatively minor issues regarding incorrect setting values and a minor tweak to the original source for the way in which the update event occurs.

Following the recoding and ironing out of these I manually placed some block data into the map in a sequence I would recognise. I restarted the game and hey presto, it worked! Twin Tigers, scrolling horizontally, but with attack waves approaching in a vertical fashion. How, very bizarre!

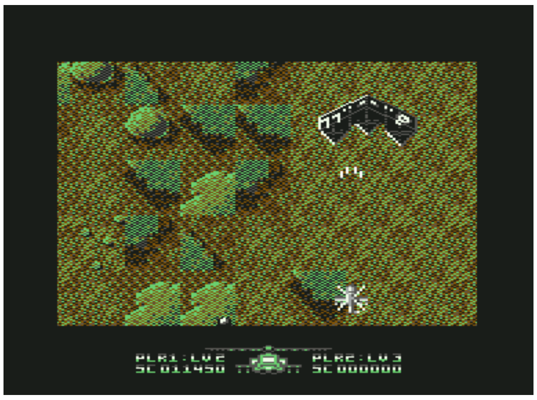
The next stage will be to get the sprites working horizontally. In the original SEUCK when an object sprite is on a scrolling screen the sprite moves down with the scrolling landscape. This needs to be changed so that it works horizontally in motion. I can't remember exactly how I did that before so I load up Breakthrough again and have a look at the changes I made. I eventually find the locations I'm looking for. $5302 and $543D both hold instructions that move the sprites down with the scroller. This references to the sprite multi-plexor table from $BC00-$BCFF.

I add in the required code making sure I include a switch to handle the 256th pixel position (as it's now moving in the X direction) and replace the original instructions with a call to the new source locations. After I re-run the game that proves to do the trick!

On my next attempt I'll be making this work in the editor itself, this is probably going to be the most exciting part of this entire project. If I can get that working SEUCK will finally become a Sideways kit. We'll have to call it S.E.U.C.K.S then, hehe…



Twin Tigers scrolling Sideways, but with Vertical attacks, how odd!



Twin Tigers Horizontal scrolling. Notice how the blocks are jumbled up, this is because they are now being read every 5 blocks instead of 8!

Day 6 – Feb 23rd 2008

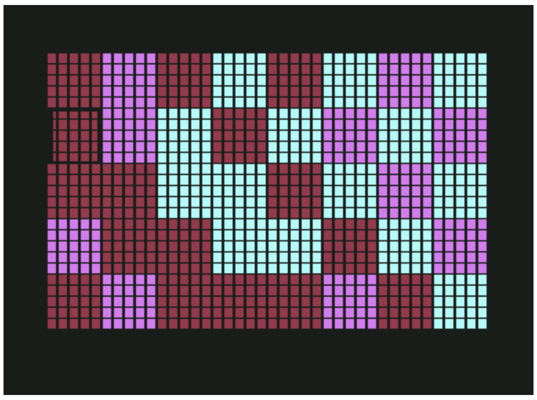
Right, this is the moment I've been eagerly anticipating, the programming session that makes the editor work with the new sideways map and format. There's a lot that will need doing here, it's not just a case of making the map go sideways, the parts where you choose the levels, insert the attack waves, define the map and construction all need to be changed. So without further ado I get cracking on with it.

Firstly I need to establish the memory locations of the actual source that handles the map editor. Loading up a blank version of SEUCK into CCS64 and going into the Action Replay I first load in the changes I've done to the SEUCK engine so far. From looking at the source memory there is a massive chunk of editor source instructions between $6500 to $A320. Searching this area for references that call on and store to the map and block areas result in a segment of code between $71D7 and $72DF. After some time analysing and following these instructions I'm happy that I've found the right area that handles the plotting of the map. He big cursor has an on screen X and Y offset relative to the map, held in bytes $7856 and $7857. After a good couple of hours I manage to make all the code changes and add additional calls to the $2b00+ area were I'd freed some space.

I start the SEUCK editor, choose an make a couple of basic blocks and go into map edit mode. Hmm.. will it work? I try plotting some blocks, then I move up to see if the map scrolls, YES! It scrolls sideways. I plot some more different coloured blocks and try moving back a forth to see if they are still there and yes, they remain intact! Now I have a look at the map location from $0900 to see if the blocks are plotting in correct strips of 5 blocks, again this is confirmed. It work perfectly, the only thing that needs to be changed that the map moves left and right, when you use the left and right directions. At the moment it's still stuck with the up/down directions.

Now this is where it gets much trickier, the source that handles the joystick movement in the editor was quite obscure. However, after quite a bit of studying the source I located a routine starting at $A13D. This was a reusable joystick control routine, which now found, I could easily follow for the rest of the editor. For the map editing it referenced to data location $E631, this just happened to hold the cursor offset boundaries. I changed these and then changed the source control to check for left and right detection. This proved to work, however the only problem with the new sideways map was that the default maximum values of the scroller for the entire map area $0900 - $18FF needed to be greater than the original values. This made the scroller end much sooner. The maximum setting for the map position is $01FC, this is because $01FC x 8 is approximately the range from $0900 to $18FF. However, $01FC x 5 is much shorter, about half that. I had to go over all the occurrences of this in the source and rewrite the new conditions. Including the in-game engine. In the end it had to be increased to $032C, and upon testing it worked!

It's been a very long day, but it's almost finished so I've decided to hammer on all day tomorrow in the hope of completing it.


Testing the map editor working sideways

Day 7 – Feb 24th 2008

Starting today with excitement my final tasks are to get the editor working with the Attack Waves, Levels and Player limitation parts. Once this is achieved my work will be complete and the Sideways SEUCK will become a reality at last!

Firstly, I aim for the Levels part of the editor, in this the player merely chooses what part of the map takes up each level. I check out in the editor to see how well it works with my existing source and notice a few bugs in the way it displays the blocks to the screen, this needs fixing so I perform various searches for the map references again and manage to find some locations between $8679 to $9000. It turns out that the routines beginning at $8679 are the ones that control the level position in the map. I spend some time checking through this, I find and rectify the bugs. I decide to leave the control system the way it is so that means the Level editor is complete.

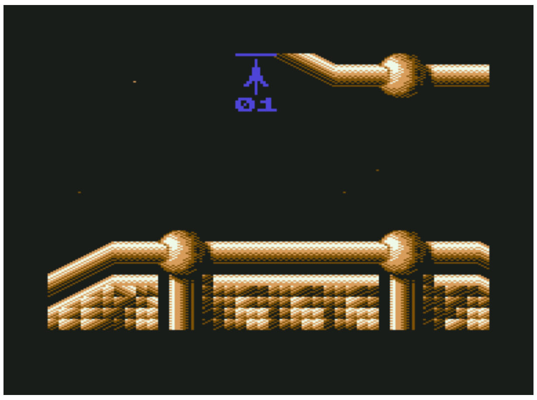
Next I tackle the Attack Waves part, which I soon found to be located from $8E6B - $8FA5. On trying the Attack Waves there are quite a few problems found. Firstly, when the ROUGH screen appears, it goes blank as if there’s something wrong with the interrupt. However, when I press fire the FINE screen kicks in perfectly, only it’s movement is the old up/down. I tackle that part first, and it doesn’t take long as I look for the joystick control call $A13D. Following the same technique as the map editor I make a necessary changes and re-test. It works, next I try and tackle the problem with the ROUGH screen. A good couple of hours later looking through my code and the editors for a problem, but it all appears to be ok…. then I stumble across a call from the editor source to the engine source that I’d changed, Aha! It was jumping mid between an instruction and setting the interrupt! All this while and it was a stupid single JMP instruction. Doh! I make the change.

At last it’s nearing the end, I try some tests, by placing some enemies on screen horizontally, oops! As soon as an object get near the top of the screen it attempt to scroll the map. Of course! In the original, when you’re doing an attack and moving the enemy object up the screen it had to scroll the map down to keep everything on track. Now I need to do the same horizontally. Hmmm.. This is proving to be the trickiest part of the project. I go back to the source again and manage to find some instructions that detect for the upper vertical position from the sprite multiplexor. I change this to work horizontally by detecting for the left most sprite position instead. Upon trying this out it does the trick and works, brilliant!

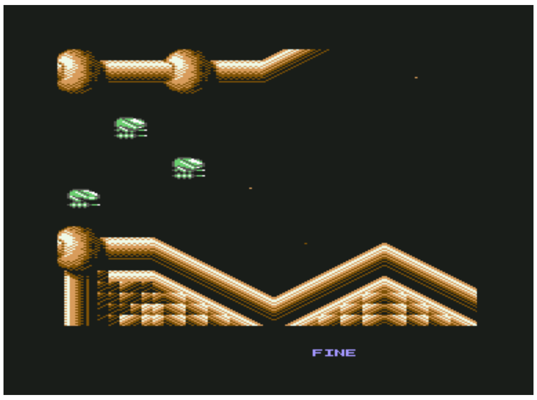
The last part is now getting the player limitation working right. Firstly, when a player’s bullet is set to no direction it automatically fires upwards. This was soon located and changed to fire to the left on the same setting! Next, when a level is set to push scroll, the map scrolls when a player hits the upper most position of the boundary set in the player limitations. This needs a complete re-write so it works when each player hits the left most position of the boundary. It’s turns out to be a much harder task as several changes within the engine are required to achieve this. When in 2 player mode if one is stationary and the other moves to the push scroll position, the other player will move with the scrolling background to simulate being stationary. I locate several different locations after lots of playtesting and searching and make the changes required.

Finally, I test for the last time…Yes, it’s finished!

As an example I’ve knocked up a quick demo using some of the BreakThrough graphics and made a very simple level. I’ve included this as well as the blank version of Sideways SEUCK in the download links below.



The Level editor working sideways



The Attack Waves editor working sideways

Donate and help towards maintaining this site, and to support FREE programs and updates in the future! Or if you prefer, you can send a donation Making it payable to 'Jon Wells' to:

9 De Grey Road
Kings Lynn
Norfolk
PE30 4PH.
UK.

My email address is:
seuck at gamesplaygames dot co dot uk

The adaptions and new source code that has converted SEUCK into a sideways editor is copyright Jon Wells. The downloads below are not permitted to be uploaded onto any other website without permission first.

Watch out for other versions coming soon I will be releasing a right scrolling SEUCK in the future.
http://seuck.gamesplaygames.co.uk/

SIDEWAYS S.E.U.C.K by Jon Wells DOWNLOADS

SEUCK LEFT SCROLLING BLANK EDITOR:
http://gamesplaygames.co.uk/seuck/seuck-sideways-l.d64

BREAKTHROUGH EXAMPLE SEUCK EDITOR:
http://gamesplaygames.co.uk/seuck/breakthrough-demo-seuck-l.d64

If you create a game with the new Sideways SEUCK that you'd like to be made available from this website, feel free to email it to me and I'll upload it for everybody to enjoy.