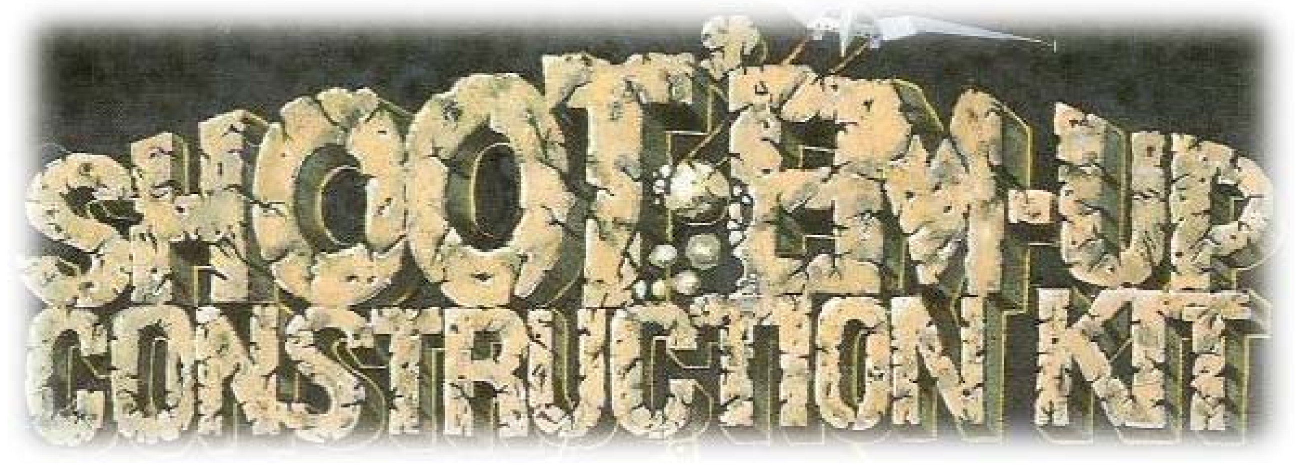


SHOOT-EM-UP CONSTRUCTION KIT

In 1987 the programmers of PARALLAX and WIZBALL decided to give away their trade secrets. It was something special.
"The Shoot 'Em Up Construction Kit" revealed the secrets of professional games programmers and gave you the power to produce games to rival the biggest chart-topping blasters. You didn't need any training or programming experience to use the selection of utilities designed by Sensible Software. Using your joystick you could build your own Spacecraft and Weaponry, design your own Landscapes, Sound Effects, Levels and Attack Waves. You could start from scratch or redesign one of the 4 FREE games included in the pack. SHOOT 'EM UP CONSTRUCTION KIT was an essential part of every Commodore 64 owner's collection, with the Amiga and Atari ST versions following soon after.

THE MENU-STRUCTURE

MAIN MENU
- EDIT SPRITES - S
- EDIT OBJECTS - O
- EDIT BACKGROUND - B
- EDIT SFX - F
- EDIT PLAYER LIMITATIONS - P
- EDIT ATTACK WAVES - A
- EDIT LEVELS - L
- EDIT FRONT END - G
- TEST GAME - T
- STORAGE - D

EDIT SPRITES MENU
- SELECT SPRITE - F1
- EDIT SPRITE - F3
- EDIT COLOUR - F5
- SELECT COLOUR - F7
- SLIDE SPRITE - S
- MIRROR SPRITE - M
- COPY SPRITE - C
- ERASE SPRITE - E
- EXIT - X

EDIT OBJECTS MENU
- SELECT OBJECT - F1
- TEST OBJECT - F3
- EDIT COLOUR - F5
- SELECT SPRITE AND PLACE - F7
- EDIT ANIM SPEED - S
- EDIT ANIM TYPE - T
- EDIT ENEMY BITS - E
- COPY OBJECT - C
- EXIT - X

EDIT BACKGROUND MENU
- SELECT CHAR - F1
- EDIT CHAR - F3
- EDIT COLOUR - F5
- SELECT COLOUR - F7
- SELECT BLOCK - S
- EDIT BLOCK - E
- EDIT MAP - M
- PAINT BLOCK - P
- COPY BLOCK - B
- COPY CHAR - C
- EXIT - X

EDIT SFX MENU
- SELECT SFX - F1
- EDIT SFX - F3
- COPY SFX - C
- EXIT - X

EDIT PLAYER LIMITATIONS M
- PLAYER 1 - F1
- PLAYER 2 - F3
- EXIT - X

EDIT ATTACK WAVES MENU
- INSERT ENEMY - F1
- JOIN ENEMY - F3
- DELETE ENEMY - F5
- XXXX UNITS FREE
- EXIT - X

EDIT LEVELS MENU
- EDIT LEVEL PARAMETERS - F1
- EDIT LEVEL MAP - F3
- EXIT - X

EDIT FRONT END MENU
- EDIT CHARACTER SET - F1
- EDIT MESSAGE - F3
- EDIT MESSAGE FX - F5
- EXIT - X

TEST GAME MENU
- PROPER TEST - F1
- CHEAT MODE - F3
- EXIT - X

STORAGE TO DISK/CASSETTE M
- LOAD
- SAVE SPRITES
- SAVE BACKGROUND
- SAVE OBJECTS
- SAVE SFX
- SAVE ATTACK WAVES
- SAVE LEVELS
- SAVE CHARACTER SET
- SAVE ALL DATA
- SAVE FINISHED GAME
- CHANGE DEVICE - C
- EXIT - X

THE MENUS

EDIT SPRITES MENU

SPRITES are the little beasties you see running about in all arcade games. The hero, the ship, the aliens, the bullets, the explosions - these are all SPRITES. Everything that moves is a SPRITE. Everything that is a SPRITE will move.
The SPRITE EDITOR will allow you to colour, twist, turn and generally manipulate all of the 127 sprites you can use in your game however you wish.

SELECT SPRITE
Use the joystick to select which sprite you want to abuse next.

EDIT SPRITE
Now you can move the flashing square all over the grid and fill in the PIXELS with your present colour by pressing fire.

EDIT COLOUR
Modify your present colour using the palette at the bottom of the screen.

SELECT COLOUR
Choose which colour to paint with next using this option. You are allowed 3 colours (plus the background colour used for erasing). You'll be lumbered with two of them for all of your sprites so make sure you really want them. The third can be different on every sprite you design.

SLIDE SPRITE
This allows you to slide your sprite around in its little box.

MIRROR SPRITE
And this allows you to reflect it in different directions.

COPY SPRITE
Use this option to COPY a SPRITE you have designed to another place. First it will ask you which SPRITE you want to copy from. Then it will want to know where to copy it to. It's as simple as that and incredibly useful when you want to do some animation.

ERASE SPRITE
This function comes in handy when you have designed something so appalling that there is no hope for it.

EXIT TO MENU

EDIT OBJECTS MENU

Once you have designed a sprite you must give it a role within the game. It may be the hero of your game or an enemy or perhaps a missile or explosion sequence. Each one of these GAME OBJECTS is given a number, for convenience. You will find that the numbers are dedicated to particular OBJECTS in the game (e.g. 00 is always player 1, 01 is always player 1 bullet etc.).

SELECT OBJECT
Select one of the 58 GAME OBJECTS available for editing and modification.

TEST OBJECT
This is really for when you're getting near the end of editing an OBJECT. It shows you how the OBJECT will animate in the game.

EDIT COLOUR
As we told you before, one of the sprite colours (colour 4) is changeable and can be different for each OBJECT. This option allows you to re-define it to match your wallpaper.

SELECT SPRITE AND PLACE
This option allows you to decide how your OBJECT will animate. Select your SPRITES, one at a time, from the SPRITE STOCK (top right of screen) and arrange them in order in the middle. Now look at ACTUAL OBJECT and Hey Presto! A superbly animating whatchermakallit.

EDIT ANIM SPEED
Decide how quickly you want your OBJECT to animate using this option.

EDIT ANIM TYPE
This option allows you to decide whether your OBJECT will animate in a SEQUENCE of up to 18 FRAMES or whether its animation will depend on its DIRECTION of movement. (DIRECTIONAL HOLD will make your OBJECT stay in its last position when the joystick is released).

EDIT ENEMY BITS
If the OBJECT you are defining is an ENEMY you will want to tell it how to behave in the game. Select this option and choose the characteristic you want to change. By holding down fire and moving the joystick you can alter the values which will determine: -
- SPEED (of movement)
- POINTS (when zapped)
- HITS TO KILL
- FIRE TYPE - arrows show direction (R = random, D = directional)
- FIRE RATE
- BULLET SPEED
- EXPLOSION SFX (linked to SFX EDITOR)
- BULLET SFX (linked to SFX EDITOR)
- EXPLOSION OBJECT (linked to ENEMY DEATH OBJECT)
- BULLET OBJECT (linked to ENEMY BULLET OBJECT)
- Result of COLLISION - ENEMY TO SHIP and ENEMY TO BULLET

COPY OBJECT
This time-saving option allows you to copy OBJECT INFORMATION.

EXIT TO MENU

EDIT BACKGROUND MENU

The BACKGROUND of your game is made up of little 4 PIXEL by 8 PIXEL CHAR(ACTER)S which are bolted together to make 5 CHAR(ACTER) by 5 CHAR(ACTER) BLOCKS. These are slotted into the MAP to create your very own landscape.

SELECT CHAR
Select one of the 254 CHARS from the CHAR stock.

EDIT CHAR
Paint the CHAR you have chosen in exactly the same way as you painted your SPRITES.

EDIT COLOUR
Modify present colour using the palette at the bottom of the screen.

SELECT COLOUR
Choose which colour to paint with next using this option. You are allowed 4 colours. This time you're lumbered with 3 for your whole MAP (colours 1, 2 & 3). The fourth (block colour) can be different on every BLOCK you design. The bad news is that the block colour can only be one of the first 8 in the palette.

SELECT BLOCK
Choose a BLOCK from the BLOCK STOCK for either EDITING or slotting into the MAP.

EDIT BLOCK
Deposit your chosen CHAR into your chosen BLOCK at your chosen position.

EDIT MAP
Insert your BLOCK into the MAP wherever you desire.

PAINT BLOCK
This useful option shows a blow-up of your BLOCK and may be painted freely using the palette displayed. This allows you to do any fine tuning you think is necessary.

COPY BLOCK
Select this option if you want to copy a BLOCK. The border will turn GREEN. Press fire when you've found the BLOCK you want to copy. The border will turn RED and the BLOCK will GLOW. Press fire again when you have found the BLOCK you want to copy to, and before you can say SHOOT 'EM UP CONSTRUCTION KIT your BLOCK will have reproduced itself.

COPY CHAR
If you want to copy a CHAR follow the same procedure as for copying a BLOCK.

EXIT TO MENU

EDIT SFX MENU

Whenever an ENEMY fires or explodes and whenever a player does something interesting a SOUND EFFECT is triggered. This comprehensive sound mixing module allows you to tailor these sounds to your requirements.

SELECT SFX
This option lets you run through all 24 sounds and select the one you want to change.

EDIT SFX
Fiddle around with the slider controls until the desired SOUND EFFECT is achieved. The abbreviations are: -

WVE	-	WAVE FORM
ATK	-	ATTACK
DEC	-	DECAY
PIT	-	PITCH
SPD	-	SPEED
TIM	-	TIME

COPY SFX
If you accidentally create a SOUND EFFECT that you are particularly proud of you will undoubtedly find that it's in the completely the wrong place. Copy it across using this function.

EXIT TO MENU

EDIT PLAYER LIMITATIONS MENU

This user-friendly option allows you to determine how Players 1 & 2 will react to every gut-wrenching situation.

PLAYER 1
Use the technique you have previously perfected in EDIT ENEMY BITS to alter the values that determine: -

- PLAYER ENABLED - indicates whether Player 1 will be taking part
- NUMBER OF LIVES
- SPEED - of Player 1's movement
- NUMBER OF BULLETS
- DIRECTIONAL FIRE
- BULLET DURATION - determines how far each bullet will travel
- COLLISION WITH CHAR - To stop Player 1 travelling over impassable obstructions, set this to the right number
- AND PLAYER WIL - If Player 1 hits an impassable object choose the outcome - STOP OR DIE
- EXTRA MAN AT 1000
- EDIT PLAYER AREA
  Select this option and you will be faced with a critical decision - how far Player 1 can move.
  Move the joystick to select which pair of cross hairs to move.
  Press fire and move them to the desired position.
- EDIT START POSITION - Move player 1 around the screen to find a suitable position for him to start each game.

PLAYER 2
Repeat for Player 2

EXIT TO MENU

EDIT ATTACK WAVES MENU

At last the time is right for putting your carefully designed nasties into the game. You must decide where they will go and what paths they will follow. Keep an eye on the memory UNITS FREE and remember that complicated flight paths just gobble up that memory and can turn your full price shoot 'em up into a budget reject.

INSERT ENEMY

This is how you put your OBJECTS into the game as ENEMIES.

- First select the ENEMY required
- Press fire to summon up the MAP (the word ROUGH will be printed below)
- Move around the map and indicate roughly, by pressing fire, where you want the varmint to begin its movement.
- You will now see the word FINE printed below
- Scroll the screen gently downwards to find exactly the right place for your ENEMY
- Press fire to make your ENEMY appear and move it into START POSITION
- Press fire yet again to let it go
- Move the joystick around to show the route you would like the ENEMY to take
- Press F7 to store the information
- It may take a short time for the computer to process the data

JOIN ENEMY

You may want to join some ENEMY OBJECTS together to make a BIGGER ENEMY.

- Insert the first section into the ATTACK WAVE sub-menu using the above method
- Select JOIN ENEMIES on the ATTACK WAVE sub-menu
- Choose the next ENEMY section to be joined
- Find the first section using the ROUGH/FINE method and press fire when it appears
- The words LINK TO WHAT will be printed at the bottom of the screen
- Position the cross hairs over the first section and press fire
- The word OFFSET will be printed at the bottom of the screen
- Locate the new section in the desired position and press fire

DELETE ENEMY

To remove an unwanted ENEMY simply select this function and use the ROUGH/FINE method to find the redundant ENEMY. On pressing fire you will see a cross-hairs indicator. Place it over the ENEMY, scream "Eat Laser Pie Alien Filth" and press fire for instant removal.

EXIT TO MENU

EDIT LEVEL MENU

This option allows you to determine up to 22 different levels for your fun-packed, supa soaraway shoot 'em up. Each level may be STILL, AUTOMATICALLY SCROLLING or PUSH SCROLLING and may span any area of the MAP you have defined.

EDIT LEVEL PARAMETERS

Move around the table displayed to choose the PARAMETER you want to change. By holding down fire and moving the joystick you can alter the values which will determine: -

- TYPE - STILL, SCROLL or PUSH
- SPEED - of SCROLLING
- DURATION - of STILL SCREEN
- END OF LEVEL - what happens next... REDRAW, CONTINUE or LOOP

EDIT LEVEL MAP

Set the beginning and end markers for each LEVEL by selecting this option. Push joystick down to view beginning marker and up to view end marker. To reposition the markers for each LEVEL, hold fire and move the joystick.

EXIT TO MENU

EDIT FRONT END MENU

SEUCK comes with these three options for defining and making the "front end" - the title screen for your game. Players will see this before they press fire to start a game.

EDIT CHARACTER SET
This is where the hard work ends and the fun really begins! With this little beauty you can completely change the style of the letters used in your beginning of game message and score indicators. Can hardly contain your excitement huh? Well just plot those pixels and create those chars.

EDIT MESSAGE
Use the joystick to position the flashing square and type in your opening message and credits.

EDIT MESSAGE FX
Now move the joystick to select one of the 23 vibrant colour combinations to bring your opening message to life.

EXIT TO MENU

TEST GAME MENU

You can test your game at any stage of development by selecting TEST GAME. To pause press RUN/STOP. To quit press Q while paused.

PROPER TEST
This is the real thing, what the punters will see. Now just sit back and wait for the critical acclaim to come pouring in.

CHEAT MODE
This enables you to test out the game from the last level edited (in the EDIT LEVELS MENU) with an infinite lives POKE.

EXIT TO MENU

STORAGE TO CASSETTE/DISK MENU

SAVE or LOAD part or all of your data to tape OR disk using this facility. Select the required option and follow the prompts.

SAVE FINISHED GAME
Saves out a copy that will load as a separate game, but will not reload into the editor.
To load from BASIC, simply type LOAD"NAME",8,1

CHANGE DEVICE
Toggle switches between tape and disk access.

EXIT TO MENU

GLOSSARY

Here you'll find some of the definitions for terms used in the Shoot 'Em Up Construction Kit, and in this online manual.

ANIMATION
A sequence of SPRITES run in a set order to give the impression of movement.

BLOCK
An arrangement of 25 CHARACTERS into a square.

CHARACTER
A square of 4 x 8 PIXELS. The basic unit of currency in computer graphics.
A common abbreviation of CHARACTER is "CHAR".

CHARACTER SET
All the alphanumeric digits, each designed within a single CHARACTER.

EDITOR
A utility to help you design a particular aspect of your game.

FRAME
A single stage of an ANIMATION sequence.

FRONT END
Everything that takes place before the zapping begins.

MAP
A selection of BLOCKS arranged to make a game BACKGROUND.

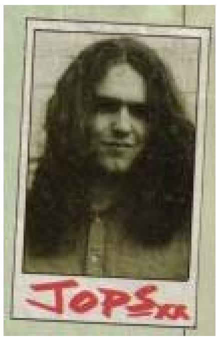
OBJECT
In SEUCK, every moving thing is defined by a series of rules. These are defined in the OBJECTS menu.

PIXEL
A single unit of colour which combines with others to make any graphic image.

SCROLLING
A technique that moves the image on the screen in a way that gives the impression of a moving landscape.

SPRITE
Sprites can move across the screen without erasing the background. You can also detect a collision between two different sprites, or between a sprite and the background. Sprites in SEUCK are used for all the moving objects - the player, the bullets, the explosions and the enemies.

THE SENSIBLE GUIDE TO CONSTRUCTION – Part 1



Hands up all who've bought the Shoot 'em Up Construction Kit, but don't quite know what to do with it. Sensible Software's Jonathan 'Jops' Hare is here to explain how to get to grips with the program, and have fun with a little Slap 'n' Tickle in the process.

The Shoot 'em Up Construction Kit is like a game of chess: it doesn't take long to grasp the fundamentals of what to do and how to do it, but it could take a lifetime to master the infinite subtleties of its machinery.

It's for this reason that I've decided to put together an extensive document of tasty SEUCK hints, tips and info to help everyone, including ourselves, to fully understand just how unbelievably fantastic the latest Sensigame really is.

It's possible that your first reaction to SEUCK is: 'Good God! Where the Hell do I start?'. However, after a soothing rub-down from a Swedish masseur and a refreshing cup of piping hot water, it's advisable that you sit down and re-adjust your perception of the whole Cosmos. SEUCK is not the sort of creature that is laid into straight away. Don't sharpen your knives and prepare for the kill - you'd be best off taking at least three picoseconds of your precious time to think out, at least roughly, what you're trying to achieve. What are you trying to prove, you filthy, ugly nobody in the world of software (not yet, anyway). Ha! Ha!

Enough of that, now for the sensible stuff (double Ha! Ha!). OK, as you know, SEUCK is a very versatile tool - it can offer you still-screen games such as Robotron 2084 (remember that) and Gauntlet, or classic arcade scrolling games like Slap Fight, 1942, Astro Wars and that perennial classic, Kill the Evil Alien Bar Stewards.

If required, game types can skilfully be combined: a bit of scrolling, a bit of pushing and a bit of still-screen, to give you a truly awesome genius of a gem - the software of the 90's perhaps.

It's advisable to master each of SEUCK's separate utilities - Edit Sprites, Edit Background, Edit SFX, Edit Objects and so on - before you embark upon your first chart-busting wonder. For this reason I suggest that you start your induction course by loading one of the four free games in the module and then breaking into the editing menu. By pressing the space bar, you can proceed to destroy all my hard work. For instance, load Slap 'n' Tickle, and from the main menu, select EDIT SPRITES and find the main ship's sprite. Now the fun begins. Press 'E' to erase the sprite - of course you're sure - and deface the empty sprite with a portrait of your favourite Page 3 Pin-up!

When you're quite happy with your artwork, return to the main menu and TEST GAME. Bingo, by jingo! Now you've got your favourite shoot 'em up starring your favourite foxy chick. What's that you're saying? When you turn sideways it goes all funny? That's because your player/ship is a directional object, and when you bank sideways, the sprite tilts through the air, just like a real alien spaceship does.

To alter this horrendous ugly mess, simply EDIT OBJECTS and select Player 1 ship. Press F7 and find your lovely lady in the top right hand box by running through the sprite stock. When she appears, press fire and you can plonk her down until she fills all 18 places in the OBJECT. Just keep on pushing fire and moving the joystick - you'll get the hang of it.

Now go back to the game and you will see that Miss Big 'Uns is a perfect specimen, as flawless on the screen as she is in the flesh. Change the SFX for Player 1 Fire and you can make her howl with pleasure every time she fires.

There, wasn't that just too fast and simple for words...

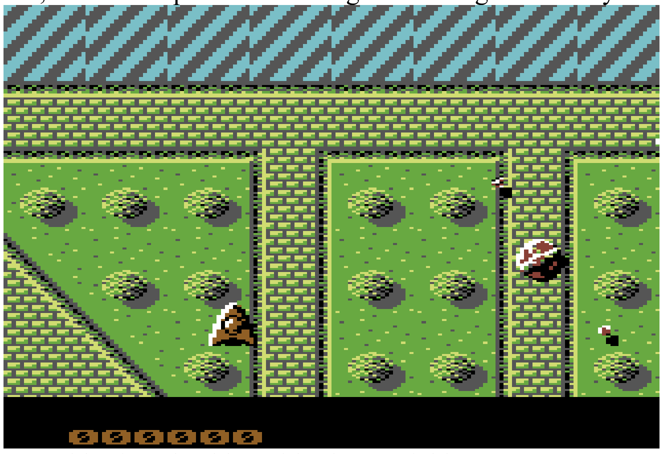
Obviously I could go on forever talking about little alterations you could make to the games included, but it seems to me that the easiest and most informative way to cram these pages full of hints and tips is to give you a detailed rundown on exactly how I went about creating all of the fab free games that come with SEUCK. So, here we go with a utility-by-utility account of the creation and manipulation of...

SLAP 'N' TICKLE

The basic idea of Slap 'n' Tickle is to show that SEUCK is capable of creating a straightforward scrolling shoot 'em up, well up to and above the standard of the average C64 blasters available at present. I wanted the game to be free flowing and easy to get into, so it's done in the classic 'full-screen scrolling with a mothership at the end of each level' style.
Although every game is constructed by constantly changing from utility to utility in order to make the whole thing fully operative, I've decided to divide this piece up into separate utilities in order that I may explain myself a bit more clearly. If you know what I mean, Harold.

EDIT SPRITES AND OBJECTS
Here we are talking about graphics and graphic techniques. The first thing one must decide when designing is from which angle the action is being viewed - in Slap's case, from above. After that comes the direction of the light source (if there is one). This is essential to determine which direction shadows might fall, and in bas-relief graphics to determine which edges of an object are highlighted - that is defined, with a light colour - and which edges are lowlighted or defined with a darker colour. This technique is used in order to give a more 3D look to graphic images. For Slap 'n' Tickle, I've imagined the light source coming from the top left hand corner of the screen - hence all shadows fall towards the bottom right.
Once these things are sorted out, you then have a minor dilemma on your hands. You see, Commodore multicolour sprites - the format of all SEUCK sprites - are created from three basic colours (the background colour is transparent). Two of the sprite colours, once selected, have to be used in every single sprite in the entire game, so you have to be damn sure where you want them! Choose wisely, or you could regret it later.
With Slap 'n' Tickle, I plumped for black and white as they're nicely contrasting for a sharp light/shadow effect, and they're also the strongest colour on the C64, really bringing the sprites to life and making them stand out from the relatively pastel background.

PLAYER 1 / 2 SHIP
This is basically straightforward, being a rounded triangular shape with a dome in the middle. Note the shading and light source I was prattling on about. Once I was happy with the design, I proceeded to copy the ship into the next vacant sprite using Chris Yates' 'Oh such bliss to use' sprite copier and proceeded to tilt the ship slightly on its side and adjust the shading accordingly.
This is really a process of trial and error as to what looks best. To give yourself an idea of how you're doing, press F1 to select the sprite and go back and forth between the two sprites you're animating. If it looks good enough, thumbs up - if not, try again. This process isn't as hard as it first seems with a bit of practice.
Anyway, I took my titled ship and copied it to the next vacant sprite, mirrored it so that the ship was tilting the other way and adjusted the shading accordingly. Simple eh! The next task was to select EDIT OBJECTS and put the Player 1 'ship on an even keel' into the forward, backward and neutral directional slots. I then put the 'ship banks to the left' in the left, and the 'ship banks to the right' in the right - it's easy when you know how.


More blasting chuckles with Slap 'n Tickle

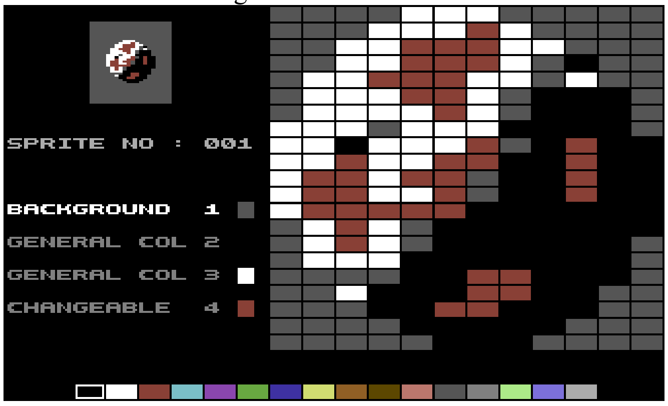
TURRETS
The turrets are custom designed to sit on top of the towers - it's not just a fluke. The main thing with these was to create a long, smooth animation sequence that didn't use up too many sprites. The closing frames are the opening frames running backwards, and the spinning crossguns are merely three frames of a cross at different angles to give an appearance of rotation when animated. The turret is also held closed for a few frames to make the object seem more solid.
Incidentally, the Copta aliens use a similar technique of a sequence of frames repeated to give the impression of rotation, but the Copta has a four frame sequence.

MOTHERSHIPS

The Dragon and other Motherships are literally pieced together using JOIN ENEMIES. Each sprite was designed with careful attention to create a clean join between neighbouring sprites in the finished alien. Again, for the Dragon's wings I simply copied and mirrored the sprites and adjusted the shading. Take note - this is a very useful time-saving technique.
The Strobes are just a circle made from different sized rings of the sprite colours, with colours switching positions to different rings, and animated to give the impression of rings constantly appearing, expanding and disappearing.
The Chessboard bullets use another simple technique of shrinking a shape accordingly. Also note the way the shadow animates with the bullet.
All the sprites except the Player Ship have shadows, and they're there in order to lift the sprite from the background. Obviously, the Player Ship is that much further from the ground, and the shadow would normally be cast that much further away from the vessel. This isn't possible to show with SEUCK, so a shadow was omitted.
For a game with many geometrically based enemies like SEUCK, a simple, logical mathematical approach is often the best way to produce good animation sequences, but don't let an obsession for geometric accuracy overshadow your artistic eye.
And now - yes, yes, yes. It's...

THE BACKGROUND

The background has three colours that must be used in every block, and one that may be varied. I wanted the For Slap 'n' Tickle background to be quite soft, but still have some pronounced shading, and also some space scenes - so I chose black for space, and yellow and dark grey to create a soft, but still contrasting solid backdrop.
I constructed bits of the map with pathways, so that the tanks could run up and down them. To save tediously piecing together each block, I used COPY BLOCK mode and just edited the minor changes for pathways to each block afterwards. This technique is used a lot when defining blocks for a specific area of the map, such as the Chessboard.
Running out of blocks can be a problem, but things like outer space and the motorway are real savers - repetition can work with a little imagination.


SPRITE NO : 001
BACKGROUND 1
GENERAL COL 2
GENERAL COL 3
CHANGEABLE 4

The superb sprite designer, editor, animator...

OBJECTS

Enter the EDIT OBJECTS menu, press 'E', and you can really get down to fine detailing the little nasties. There are many things of interest I could point out, but here are the juiciest bits of gossip I can think of.
By making the fire rate of an object equal to the animation speed, multiplied by the amount of frames in the animation sequence, it's possible to synchronise the firing of the object to the first frame of the animation sequence. What a mouthful! Examples of this are the turrets and the motorbike and armoured lorry on the motorway.
By giving the Mothership sections different firing directions and bullet objects, spectacular and hard-to-avoid firing effects are created painlessly. Try playing around with these.
Little touches can be added, such as the chess pieces requiring as many hits to kill as they are worth in pawns in a real game of chess, and creating turrets of different strengths so that the tough ones fire faster bullets more often and require more hits to kill. Oh - the list is endless...

ATTACK WAVES

This is the make or break of the game - but watch out in case you run out of units. Economy is the key word - there are sections of the map in Slap 'n' Tickle which aren't used, largely because we had no attack wave space left to put aliens in these areas.
The ATTACK WAVE EDITOR is unbelievably easy to use. Paths were created for the tanks, and using the Attack Wave Editor I easily got them to follow them like the stupid sprites they are.
The Coptas on later levels take off from one starport and land on another, and the chess pieces move just like real chess pieces do. God, it's so good.

One technique to note with the Attack Wave Editor is to start the enemy's movement pattern beyond the perimeters of the screen, so it actually appears to scroll on with the landscape. However, the Nemesis-like space spinners are sometimes deliberately made to appear in the middle of the screen for an added element of surprise.
The Motherships are joined aliens. Simply take the first part of the ship and move it in the desired path. Watch you don't run out of flight path - I did many times. You can now construct the rest of the Motherships using JOIN ENEMY, and there is no need to put in a flight path - SEUCK will work it out automatically. Simple eh!!!
Basically, the secret of good attack waves is a lot of selecting and re-entering until you get it just right. A lot of game testing is advised at this point, and you may need to play around with the alien's firing rates and what have you.

EDIT SFX
So now you've got a fantastic game with no noise! The only real sound of any note in Slap 'n' Tickle is the whirring at the beginning, which is in fact the continual sound of tanks firing. This sound effect has a very long decay, so it ends up looping into the next fire. The motorway noise is based on a similar principle, made up from the firing of all the vehicles plus the player start noise, which I am really proud of. When both players start at once you hear a chord. The same sound is used for both start sounds, but one note is five semitones higher than the other. Musical - that's what we are here at Sensisoft!

EDIT LEVELS
The start and finish of levels are quite straightforward, but when you've come to the end of a level and wish to have a still screen, you have to redraw in order for the aliens to appear. Note the way that pieces of the map can be missed out and other pieces, like the still screen with the freaky border and the Mothership, can be re-used without any hassle. Just slot it in wherever you fancy, and don't forget to loop at the end of your sequence of levels.
Also, check the timing of any still screen levels. I had to spend a little time adjusting the timing so that the screen moved onto the next level just after the Mothership had disappeared, as opposed to miles after it, or whilst it was still battling away.

PLAYER LIMITATIONS
Once you've finished your game, the final task is to decide how much help to give the player. How many bullets at one time, and how much of the screen can be navigated? I decided to be generous and give Players 1 and 2 loads of bullets and virtually the whole screen to play with. It's more fun that way after all.

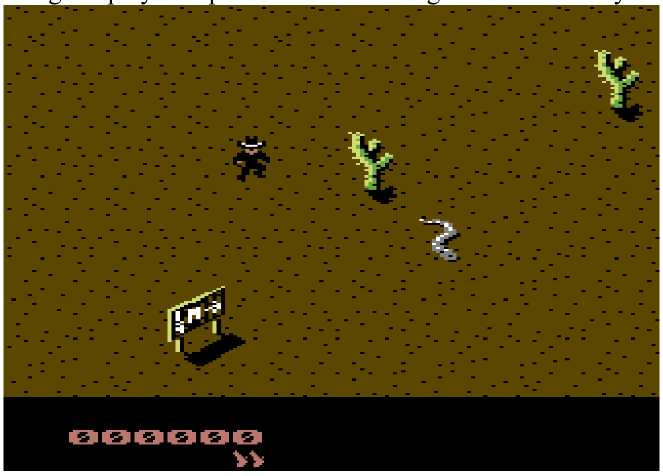
FRONT END
Yes! The really fun bit. I knocked up a naff character set, cracked loads of personal jokes, and bingo - a wild 'n' wacky front end. Who needs a scrolling message?
Remember, you can approach the SEUCK utilities in any order you want (within reason). There are no rules as to the perfect method, but a modicum of planning is needed or you could have a fabbo attack wave on a section of background that you've forgotten to actually include in the game. Got my drift? What a laugh eh! What a ruddy laugh! Ha! Ha! Ha! What a SEUCKER you are mate! Ha! He! Ho! Ho! Cor Blimey Guv'nor - what a Corker!!
So that's Slap 'n' Tickle out of the way, but don't despair, Jops will be back from the psychiatrist in time for the next chilling installment of: The Sensible Guide to Construction. Stay there and scroll down - you know it makes Sense...

THE SENSIBLE GUIDE TO CONSTRUCTION – Part 2

Still having trouble creating a smash-hit mega-game with the Shoot 'em Up Construction Kit? Never fear - Sensible Software's Jonathan 'Jops' Hare is here to give hints and tips on how to squeeze the most out of SEUCK.
Yes, yes, yes! Back with a bang! It's Jovial Jops, the people's favourite with the second and final installment of the Sensible Guide to Construction! So let your hair down a bit and become a hippy - yes, yes, yes, no, no, no, don't ever become a grimy egg-smelling hippy - oh no sirree! Become a computer bore - yes! A fabulously dull chip basher. But wait! Now, thanks to Shoot 'em Up Construction Kit, you can write your own full-price chart-topping blasters and be the God of Software. Just buy one or more copies of SEUCK (no full-time programmer ever works without back-up files) and you can spend your days playing with your joystick in the corner, creating the most earth-shattering computer game of all time. Just imagine, for £14.99 you could be more famous than Tony Crowther. Ok, enough of the advert. Sorry kids. Now for the hard core stuff.
In Part 1 I divulged a bit about Slap 'n' Tickle amongst other things, and in this part the show must go on with the three other devastating pieces of original game design: Outlaw, Transputer Man and Celebrity Squares, all children of the big SEUCK parent in the sky, except Celebrity Sqaures, which is the illegitimate child of Gary Liddon and Gary Penn with able assistance from other parties.

OUTLAW

At last we are where our hearts really lie, deep inside the C64. What do we see, but miles and miles of desert. Fear not, for this is Outlaw, and we have just been taken back to Arizona to the turn of the 20th Century for the greatest Spaghetti Western in History.
The main objective was to show the way in which SEUCK could be used to create Commando style combat games by using the player to push the screen along. This is basically the 'push' selection in the LEVELS EDITOR.


Wild West action with Outlaw

SPRITES - OBJECTS
The sprites were a tricky bunch, the main problem being that it was a task way beyond me to manage to get a shadow on all the sprites. As I am of late quite obsessed with shadows, (you should see my collection of classic Scandinavian shadow books under my bed) this caused me quite a dilemma.
The basic Commodore sprite is just too small to create a detailed enough image of a person or body and still have room left to squeeze in the shadow essential with most sprites viewed from overhead. So, after much beating around the bush, I decided that I would have to lose the shadow on all human sprites and hope it didn't show too much - luckily I don't think the end result was that revolting! In fact, there are some Outlaw sprites I am quite proud of.
I had quickly decided on my favourite black and white as the two permanent sprite colours. When I came to do Player 2, the buxom young Texas wench that wiggles as she walks, I thought 'Oh! What to do here'. But once I got her wiggling, Chris could hardly pull himself back to his disk loader he was so excited, and I thought, "Hey! This is no ordinary sprite, this is a living, breathing, sexy young lady", and promptly asked Matthew Tims, the charming boss at Outlaw, if he wanted my voluptuous sprite to dress up in something skimpy and pose for the front cover of Shoot 'em Up Construction Kit.
Actually, both Players are good examples of multi-directional Objects, although you may find the two-frame animation a bit hard to cope with at first. (Watch the way our hero stumbles sideways). Also, the injun with the daggers and the young whippersnapper who drops the bombs show how the same multi-directional features can be applied to an enemy as well as to a Player. Note that it's also possible to define your own frames of a sprite moving diagonally and put them in the appropriate boxes in the directional object! I must confess I cheated and used the forward and backward sprites for the diagonals of all the multi-directional objects in this game, but it does conserve a lot of free sprites, so it's not cheating really.
Both the kneeling injuns (I used copy and mirror to get them facing both ways) and the sheriff use different animation frames more than once in their animation sequence. This is another useful and effective sprite-conserving technique. As I explained badly in part 1, the Object firing rate can be synchronised so that it fires every time the Object enters the first frame of animation. This is done by multiplying the animation speed value by the number of frames of animation, and then making this new number the Object's firing rate. Try it out for yourself, it really does work.
The last word I have to say about sprites is to do with the varmint, the white haired old boy - notice the way he is designed with half his body missing, in anticipation of how he will relate to the background in the finished game.

BACKGROUND
Just like the Wild West itself, the majority of the Background is desert. Shoot games have one or two repeated blocks in the background that keep occurring time and time again, and Outlaw is no exception. Just two blocks made from about eight or nine chars form, at a guess, about 50% of the entire background. The secret when trying to create a natural terrain is to make the whole thing look as irregular as possible. Well, I tried.
The most striking thing about Outlaw is the contrasting perspective between the towns and the desert, and how this actually works without looking wrong. This is because the human brain warps its perception in order to see things in a nice standard way, so that it can figure out what's going on. If you think that the towns in Outlaw are bad, or that I'm rambling on about the most boring thing ever, you're probably a very perceptive perkin, but you're not wanted any

more, so go and buy Commodore User, (sorry, I don't mean it - honestly). So, back to the point. Pay attention you lot! Pay attention everyone!
The mazes in the desert scenes are made from rocks and crosses and things, and are done by making all the chars beyond a certain point solid, so that your man cannot pass through them - cunning, eh? Note the way all of the shadows of the objects are also non-detect chars, just to make the 3D even more realistic, not only visually, but actually being there as well!
The towns on later levels have mazes of sorts in the form of buildings. By linking these mazes with clever Attack Waves, you can actually create puzzles whereby certain aliens have to be killed from a certain place; so you always have to think about where you are on the screen, and how far you can scroll up it without altogether losing a necessary escape route from the carnage.

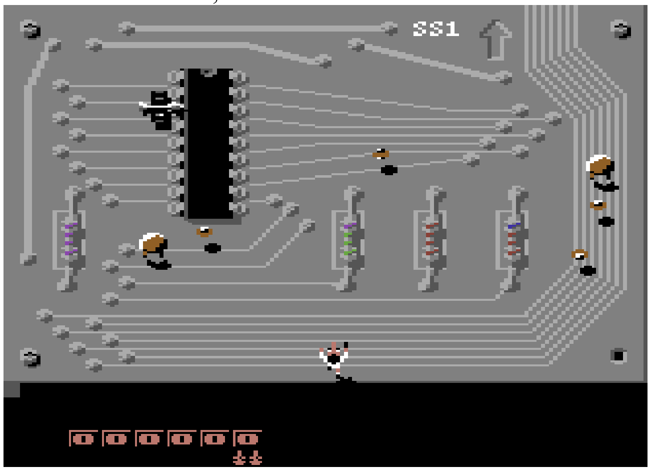
LEVELS
Basically, Outlaw is a 'push' Commando style game that uses a still screen at the very end of each level. This breaks up the action and makes the player feel as if he has reached a definite milestone in the game - non-stop pushing is very soul destroying. Also, as happened with Slap 'n' Tickle, some parts of the map had to be missed out due to lack of ATTACK WAVE memory. Two pieces of map were stuck together using the CONTINUE option between the two sections to cover up the gap and make it appear to the punter as one totally smooth and whole piece of background. Ingenious, eh?

ATTACK WAVES - OBJECTS II
The key to the ATTACK WAVES is all to do with where the enemies appear on the screen. Because it is a pushing game, one knows exactly how high up the screen the player is going to be when the enemy does appear. So, if you have the enemy come on and fire directly at the point where the player is standing, the game obviously becomes a lot harder. This is why a lot of the injuns with bows and arrows slide on from the top half of either side of the screen, thus directly harassing our poor defenceless here in the middle. The varmints do the same thing on later stages.
A lot of the enemies deliberately make life difficult for you - the boys drop bomb boxes invariably in the very place you were hoping to go, the snakes slither down the pathways in the desert mazes, and the sheriffs don't always pick the most convenient places to carry out their shooting trials. Also, you should observe the ENEMY BITS, the way that identical looking aliens are in fact made more vicious and spiteful for later levels, especially the varmints. The worms act purely as shootable and oh-so-loveable bonus points, and the quicksand actually blends into the background by staying still and scrolling on with the map as if it were part of it. The fires also do this, but they fire also (get it?). The use of animated sprites in the background is especially apparent in the fiery injun village Matthewsville.

PLAYER LIMITATIONS
Really, the whole crux of Outlaw is exactly where the pushing level is in relation to the screen. This is done on the PLAYER LIMITATIONS editor, and wasn't finalised until the very end. Short-range shots were also used in order to force the player to have to move around the screen more. So folks, in a nutshell, that's Outlaw - perhaps the most fun-packed game ever to make its way onto your C64. Oh no, not quite. Sorry, I forgot about...

TRANSPUTER MAN

A new concept in computer gaming technology, Transputer Man was intended to be a still-screen, loads-of-aliens game like Robotron 2084, and in the end mutated into a total freak of its species.



Deep inside your computer, Transputer Man starts debugging...

SPRITES
The sprites are viewed at an angle, slightly from one side, but mainly from above, so you get a little bit of body and a big bit of head with the men, (recognise that fellow - old heroes never die). The shadow also adds depth to all the

sprites, except the ghost - that is immaterial and therefore casts no shadow. Also notice the way the hole from which the ghost appears is part of the sprite, even though it blends in with the background.
The flying rocket's wings, when fully spread, come in only by a pixel a time for the first few frames, then at a faster rate until the wings meet in the middle. This is because for the first few frames the wings are moving faster vertically than horizontally, which is less discernible from above than horizontal, hence it appears to move less. Hooray! Hooray! A science lesson a day keeps the doctor at bay! Discuss, arguing points for and against in not more than five words.

BACKGROUND
The background is designed in greys so as to avoid black and white, the harshest of the 64's colours. The shapes are simple and clean for that extra computer-like feel. Again, detecting chars are used to block your path, and on later levels the gaps between the boards are also used to make the screen less easily negotiable. I had to make sure that all of the map's out of bounds areas were detecting chars, otherwise a dead transputer man may well have been revitalised in a totally useless part of the map. For my transputer, I filled a blank block with 25 unused characters, and used the delightful PAINT BLOCK feature to design the Transputer. Always use the PAINT BLOCK for defining large detailed shapes on the landscape - it's a Godsend. The blocks went very quickly with this game because there's so much detail to each screen.

LEVELS
The reason the screen is repeated five or six times on the map is because each apparent level is really five or six still-screen levels joined together, with the same picture redrawn over the top. It's a different part of the map, so it's got different Attack Waves - clever, huh? It did take a while to set the length of each level however, as it does with any still-screen level in a SEUCK game. Just keep on trying out different amounts of time until it all flows well and smoothly onto the next phase of the game.

ATTACK WAVES
There's nothing really special to say about the attack waves apart from the bonuses, which show how it's possible to have pick-up bonuses in games, such as the keys in Gauntlet.
Also, take a good peak at those gorgeous dancing girls - the hideous little tunes they dance to are actually created by their firing noises. The bullets they fire are invisible and move very fast so as not to clog the screen with sprites. They all fire the same basic tone of sound effect. In fact, the sound effects for the girls are all exactly identical, except for the pitch of the note. The black girls play normally, the Middle Eastern girls' bass notes are a third above that, and the whites play the fifth of the root-note.
By making the firing rates of the girls all mathematically divisible - 20, 40 and 60 - and making sure all the girls appear at the same time on-screen, you get an almost-in-time little ditty, however hideous it is. The best tunes are meant to be a bit off-time - it's jazz man, know what I mean? It's feel, baby.

SOUND FX
Chris Yates did most of the SFX for Transputer Man, as well as coding all the SEUCK utilities. What a busy man, eh, and what lovely SFX they are too - all echoey and burbly. Don't ask me how he did 'em, 'cause I don't know, and I can't ask him 'cause he's asleep at the moment, and anyway, he's nearly thirty miles away, so I don't think I'd hear him too well.

PLAYER LIMITS
See how the full-screen player area makes the pushing levels look just a bit suspect? Don't say we didn't warn you!

CELEBRITY SQUARES
Well, there it is, all the SEUCK games in a nutshell. Oh sorry, I forgot about Celebrity Disaster - shame on me. Shame, shame! Mumble, mumble, mumble! Fetch me my straitjacket!
Well, Celebrity Squares adequately demonstrates the use of detectachars that kill you, and also demonstrates that dark grey is the best colour to lowlight sprites when they're travelling over a totally black background - space for those of you with an over vivid imagination. And that's it, there's nothing else I have to say about Celebrity Squares, if you want to know more about it, go out and buy it and see for yourself. C'mon, give us your money punters, we love you all!

TAPE VERSION

In a way of rounding off, I'd like to enlighten the few people who haven't understood the way the SEUCK tape is laid out. On side one there's an introduction screen followed by SEUCK with Slap 'n' Tickle and SEUCK with Outlaw. Side two starts with a blank SEUCK (ready and waiting for your latest sizzler) followed by SEUCK with Transputer Man and SEUCK with Celebrity Squares.
To load each program you must roughly locate the starting position of each program and press SHIFT / RUN STOP as usual, then press play on tape. Once you've found the starting point of each program, we suggest you take a note of the tape counter, so that the next time you want to load it, you'll know exactly where it is without having to guess. I hope that is all clear to those people concerned.

THE END (IS NIGH)

As way of signing off this fun-packed document, I'd like to tell you that one software company received a demo of a game written on SEUCK shortly after the utility was released, although it was rumoured to be a little reminiscent of Slap 'n' Tickle.
So come on you lot, get writing those games, and one day, you could be so famous you'll be burning your money because your oil refinery broke down.
Anyone can write games on SEUCK. I wrote Transputer Man, Outlaw and Slap 'n' Tickle all by myself, and I can't even format my own disks! So for once, us ones with all the ideas and no technical knowledge can knock those smart-assed programmers on the head. Now we can be equal or even better. We can also be Butter, Margarine, Coleslaw, Lemonade, Hamwiches, Bernard Matthews, Ted Moult, John Noakes, Blue Peter, Play School, Humpty, Hamble...
Man Servant! Summon me my splendiferous technicolour dreams straightjacket. My name is Rubberwinkle, Gibber, Gibber!!

The future's so bright, Jops has to wear shades...

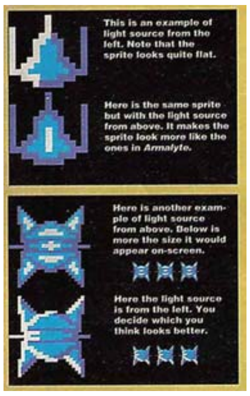
THE SECRET OF SEUCKCESS - Part 1
By Andrew Smith

This article first appeared in Commodore Format issue 30.
If you can't beat 'em, join 'em. And with Shoot 'Em-Up Construction Kit, you can. When it comes to writing games, it does all the work for you... Well, not all of the work. You still have to come up with the ideas, and that's where Andrew Smith, author of Aquablasta and Daedalus, can help you out.
Writing games with Shoot 'Em-Up Construction Kit is a doddle. Well, yes and no. SEUCK (as it's known to its friends) does do all the hard work for you - all the number crunching and routines-stuff - so it's easy enough to write a game that works. Writing one that is fun to play is another matter altogether, and requires a lot of thought and creative input. While most of that has to come from you, there are some general rules to follow if you want to make your game as playable as possible. Wanna know what they are? Then you're in the right place.

THE INSPIRATION

The first thing to do is to decide on a scenario. Not all shoot-'em-ups have to be set in space, so be more original, let your imagination go wild. One of the most memorable SEUCK games the CF crew has seen was called Spoony Lad which featured pitched battles between various bits of cutlery - pretty daft, but it caught our attention. Why not try a Wild West shoot-'em-up or one set inside the human body, with white cells battling viruses. If you do want to set your game in space, try to think up a new twist.
When you've decided on a scenario note down your ideas. Think of a brief plot, something simple that will get the ideas flowing as you write your game. Don't write a complicated plot; leave that until you are creating the game. Writing a brief storyline helps you decide how the scenery and aliens should look. This is important because you don't want a mish-mash of sprites - that'll just look a mess.

CREATING SPRITES

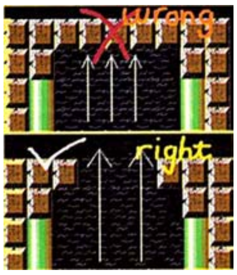

This is an example of light source from the left. Note that the sprite looks quite flat.
Here is the same sprite but with the light source from above. It makes the sprite look more like the ones in *Armalyte*.
Here is another example of light source from above. Below is more the size it would appear on-screen.
Here the light source is from the left. You decide which you think looks better.

Load up your copy of SEUCK, go into the sprite editor and have a go at drawing the player's sprite. Do a few and choose which looks best. Draw your first ship in sprite number 0, your next effort in number one and so on.
Take note of the colours you're using; you're limited to three colours plus the background colour. Don't draw your ship in a solid block of colour because that tends to make it look flat and two-dimensional. Use a white to suggest that light is shining on the sprite and a dark grey to indicate shade. If you want to use this style shading for the enemy sprites as well don't change the colours you use for the light and shade. You can only change the third colour, so alter that if you want the enemies to be a different colour (which is a good idea).
The best method for suggesting a 3D sprite is to use what's called light sourcing. Imagine there's a lightbulb somewhere around your sprite; use your lighter colour on the parts of the sprite where the light would fall (yes, if the bulb was on, pedant). On the opposite side of the sprite use your shading colour.
How to lightsource a sprite from different angles

BACKGROUNDS

This is one of the hardest parts of writing a game. You're limited to a few colours, which makes it difficult to draw decent scenery. The best colours to use for shading and light are brown and white respectively (the technique I used in Aquablasta). This leaves you with just the first eight colours the C64 has to offer which does cause problems. Try to avoid the colours red and dark blue because they clash with the brown and look awful. Now you're down to six colours - not a lot really.
Start off your scenery simple then gradually add detail; it's much easier this way. The best advice here is feel free to experiment.

THE SCENERY MAP



This is a part of SEUCK that allows you to take the blocks you drew in the background editor and paste them onto the screen forming a map through which your sprite has to navigate. This is where the game starts to take shape, so spend a lot of time making the graphics look good. A word of warning - don't make your map too complicated. Nothing will put someone off playing your game as much as the Game Over message popping up every few seconds.
Also it can be very easy to get carried away with pasting blocks everywhere and without realising it join up both sides of the screen with scenery. This means that the player won't be able to get past, he'll get annoyed and very probably won't want to play again (which is fair enough). There are a few hints about what works in our handy diagram (the one called "Handy Diagram") but again, experiment to see what works best.
Blocked path means no way through. Make sure you don't get carried away and block the route up the scenery.

GETTING ANIMATED

This is quite hard to do well and needs a bit of practice. Say, for example, you wanted to rotate an enemy sprite along its y-axis. What you do is draw the un-rotated sprite then copy it into the next sprite area. Alter the copied sprite a tad, just enough to see that it has actually moved. When you've finished that frame of the animation copy it into the next area and alter it slightly again. Continue doing this until the sprite is fully rotated. If you can do that then you are well on your way to creating decent animations. What you have to remember is to keep all the sprites you used in the animation the same colours.

THE ENEMIES

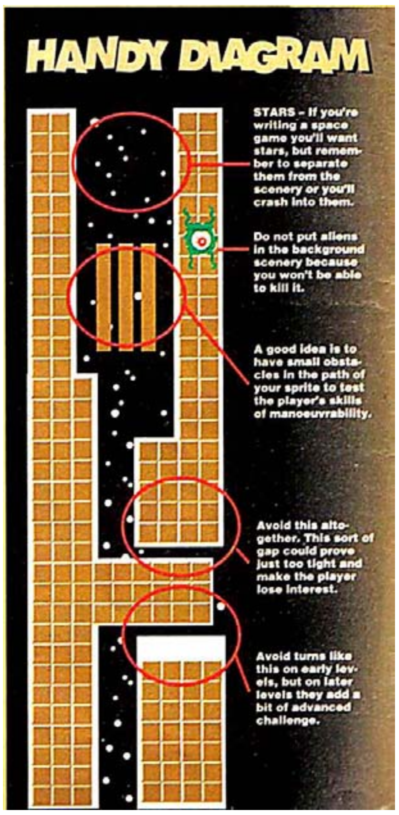
This is the most exciting part of creating your game because when you've dotted a few enemies around your map you can test the game out by blowing up your creations. Don't be tempted to put too many enemies on-screen at once; not only does it make the game more difficult, but also slows it down and makes the sprites flicker.
The best way to add challenge to the game is to make each individual enemy more deadly by, for example, making the enemy bullets faster or making an alien fire in all directions.
As you put the enemies into the game, invite a mate or two around to test the game so far and to discover their opinion on whether it's too hard. Trust me, your game will be much better with a little help from your friends (the Beatles once sang a song about it, and the home computer hadn't even been invented then, they were that clever).

GUARDIANS

Massive guardians, we've all seen 'em. It's a tried and trusted formula - large, Steven Seagal-like (erm, I think he means hard to kill - Ed) aliens at the end of every level. Guardians are traditionally made up from several sprites so it's best to draw them on paper before trying to create them on the computer. Because of the size of the guardians it is best to use simple onscreen movements as it saves memory. Don't under any circumstances have your guardian zipping about the whole screen at warp factor nine - this wastes a lot of memory.
Another 'tradition' with guardians is that they take multiple shots to kill, so make sure that all the sprites that make up your guardian need the same amount of shots to be killed. This sort of detail can be edited using the Edit Enemy Bits option. Also try not to have all the sprites firing bullets - it'll slow the game down a lot, lead to flickering and make the guardian next to impossible to kill (that'll be Steve Seagal's next film, then? - Ed).

PLAYER LIMITATIONS

This is where the player's attributes are set, including the speed at which it travels, the number of bullets it can fire, the area onscreen it can move through, etc. If, while you're testing the game, you feel your sprite moves too slowly you can speed it up using the player limitations. You don't want it to move too fast, though, or else it'll be too hard to control.
The amount and speed of bullets that the player can fire is also set here. When I write games I have three bullets onscreen at once and have the bullets move at maximum speed. This ensures the game will not slow down too much, by moving the bullets off screen as soon as possible.
A handy diagram


HANDY DIAGRAM
STARS - If you're writing a space game you'll want stars, but remember to separate them from the scenery or you'll crash into them.
Do not put aliens in the background scenery because you won't be able to kill it.
A good idea is to have small obstacles in the path of your sprite to test the player's skills of manoeuvrability.
Avoid this altogether. This sort of gap could prove just too tight and make the player lose interest.
Avoid turns like this on early levels, but on later levels they add a bit of advanced challenge.

SOUND EFFECTS

There are no set rules for sound effects, but here are a few tips. If you're using lots of different types of enemy onscreen, use a different effect for each. Saw sound waves give a crashing sound, while sine sound waves create what is known in the trade as "bleeps". So you could use sine sound waves for an explosion caused by the player, and saw sound waves for when the enemy scores a hit. A combination of bleeps for firing bullets and crashes for hits works well.
The guardians should have a menacing sound, so use the sine waves at a low pitch and make their explosions loud and long. Listen to Turrican 2 or Armalyte - they'll give you loads of inspiration.

THE FRONT END

This basically means your title screen. Make it as attractive as possible, as it will influence people to play the game. If you want, use the colour cycling techniques for your title screen. Oh, and don't forget the author's name - you.

THE DIFFICULTY CURVE

When you've finished your game it has to be play-tested extensively. A difficulty curve has to be taken into consideration, i.e., the game should get gradually harder as you play. You don't want a very easy first level then an extremely difficult second one, so get as much play-testing in as possible. If you have had constructive criticism all the way through your game's development, then your game has the potential to be quite a corker.

WHAT TO DO AND NOT TO DO

DO make the first levels relatively easy, to make the game instantly playable. After each level increase the difficulty to keep the game fresh.
DO spend a lot of time on the graphics. Decent graphics will make the player more likely to want to see the rest of the game.
DON'T make the map too complicated or it'll be impossible to navigate through.
DON'T place enemies in the scenery because it makes them invulnerable.
DON'T have too many sprites onscreen at once as they slow the game down.
DO sprinkle obstacles in the path of the player. It adds variety to the game.
DO try to keep colour clashes to a minimum - not only do they look ugly, but also they make it difficult to see what's going on. Particularly avoid red against brown.
DO use a variety of sound effects - different ones for each enemy.
DO include end-of-level guardians in the game. They allow the player to rest their trigger finger while the scrolling comes to a standstill, add variety and challenge, and let you show off your sprite-drawing abilities.
DO make backups of your game regularly. Accidents will happen!
DO get mates to playtest the game as it develops.
DO beee dooo beee dooo dum dum.

FACT FILE

- Shoot-'Em-Up Construction Kit was re-released by GBH at budget price, £5.99 disk and £4.99 cassette
- Daedalus by Andrew Smith was on Commodore Format Power Pack 30
- Aquablasta by Andrew Smith was on Commodore Format Power Pack 19

THE SECRET OF SEUCKCESS - Part 2
By Jon Wells

This article first appeared in Commodore Format issue 31.
Offering more tips than a particularly huge economy size pack of PG, Jon Wells concludes our two-part SEUCK masterclass. If you've got Shoot-'Em-Up Construction Kit then this is ESSENTIAL reading; and if you haven't, well, 'erm, it's still darned smart!

THE PLOT THICKENS

That first, inspired idea is one of the hardest - and most important - parts of any game's development. If it's to capture the imagination of players then it must be as original as possible. Your game may be a shoot-'em-up - after all you're using the Shoot-'Em-Up Construction Kit (SEUCK) - but that doesn't mean you have to limit yourself to space, guns and laser death.
Why not write a game where you're climbing up a building which is on fire? The plot would be to rescue people from the windows, your 'weapon' being a water cannon that douses fires as well as blasts enemies.
If you're doing a two-player game, why not introduce a female character to complement the male? (Oi, this sounds very sexist - Clur, aka Clare Hodgson.) The game Outlaw (see FIGURE 1), which is included with SEUCK, is an excellent example of this. Remember shoot-'em-up is a title that should describe your game and not limit it.

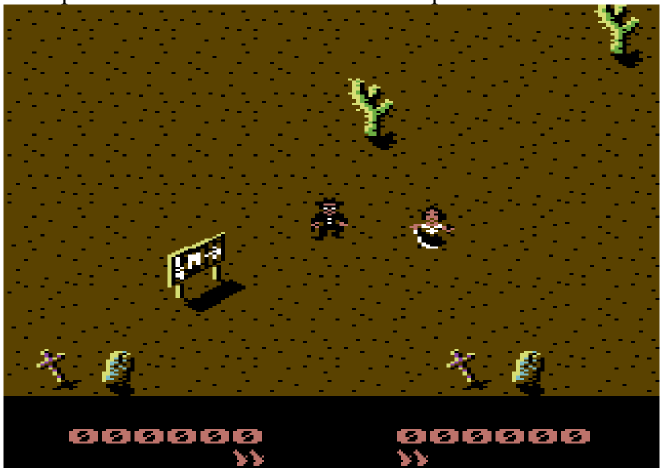


FIGURE 1 - Outlaw by Sensible Software

SWAP SHOP

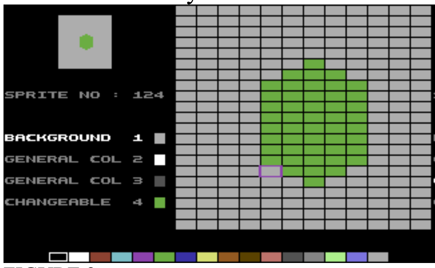
When you're using the Sprite Editor in SEUCK your sprites can be drawn in three colours: two multi-colours and one changeable sprite colour. For the changeable sprite colour you can use any one of the 16 colours that the C64 has to offer.

It's important to remember the restrictions of working with multi-colours. In this mode the C64 will only use two multi-colours for ALL of the sprites. What this means is that the two multi-colours have to stay the same for each and every sprite. (This also applies to BACKGROUND character graphics). So for maximum choice and effect, ensure that the multi-colours you choose contrast: so that, for example, one is dark while the other is light.

ALONE IN THE DARK

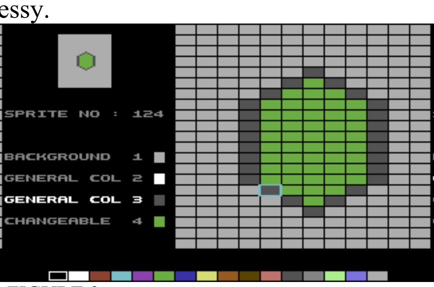
Always draw your sprites with a dark surrounding, as this makes the sprite look less blocky. As an experiment select dark green as your changeable colour and draw a solid circle to represent a sprite. (See FIGURE 2). At this stage the sprite will look blocky. Now draw a grey colour around the outside of the sprite. (See FIGURE 3). Notice the difference? Now, try drawing a small circle of light grey or white in the top left-hand corner to simulate light. (See FIGURE 4).

As a general rule, try to keep your sprites simple and don't be tempted to use high levels of colour and shading. Too much detail always makes them look messy.


SPRITE NO : 124

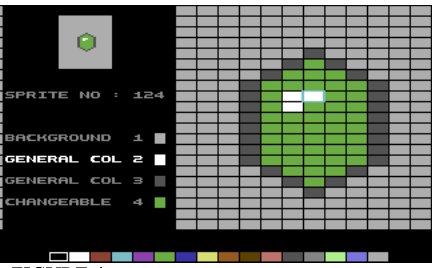
BACKGROUND 1
GENERAL COL 2
GENERAL COL 3
CHANGEABLE 4

FIGURE 2
Green blocky sprite


SPRITE NO : 124

BACKGROUND 1
GENERAL COL 2
GENERAL COL 3
CHANGEABLE 4

FIGURE 3
Grey outline smooths edges


SPRITE NO : 124

BACKGROUND 1
GENERAL COL 2
GENERAL COL 3
CHANGEABLE 4

FIGURE 4
Highlight makes the sprite more rounded

WHAT'S THE OBJECT?

The Object Editor is used to make all the "characters" in your game. Not every character has to be out for your blood or form part of an attack wave. You can use the Object Editor to make either pick-ups or background animations (torches, smoke from houses, etc).

Using the Object Editor, it's possible to simulate sideways scrolling. To do this you can treat background objects as if they were enemy bullets. For example by turning a tree into an enemy bullet we can make it scroll across the landscape. First create a tree. Then create "Enemy One" as a filled-in sprite. Select the "fire type" as left, and then for the Bullet Object select the tree. Set the Bullet Speed to one and copy the Enemy Object to "Enemy Two". Now change the Bullet Speed for "Enemy Two" to two.

Enter the Attack Wave editor and store Enemy One at the far right of the screen. Now do the same for Enemy Two and place it just below Enemy One. Blank the enemy sprite graphics so you can't see them onscreen, and run the game with your level on STILL.

The tree bullets will move from right to left at different speeds, simulating a sideways-scrolling effect. (You may have to alter the fire rate at which the bullets appear to get the effect spot on.) This method was used in the Power Pack game Monster Mash for the jumping fish (See FIGURE 5) and was first invented by Alf Yngve, so he should get all the credit.

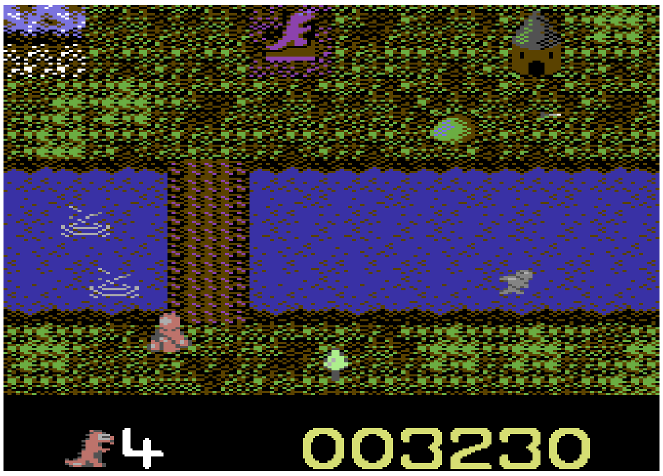


FIGURE 5 - Avoid the jumping fish

IN THE BACKGROUND

These should be designed with great care, as there is an option in the Player Limitation Screen to either kill or stop your sprite on contact with various backgrounds. So sloppy background design can make your game unplayable.
The first colour checked by the C64 when drawing a screen is the background colour (the dark blue that you first see when you restart your machine is the default background colour) and this is considered to be the first colour in the palette. Next in line are the two multi-colour shades, and like the sprites they must stay the same throughout the entire game.
The "changeable" colour is stored fourth, but for backgrounds it can only be selected from the first eight colours of the palette, and not all 16. This is because of the way multicolour graphics are drawn, as all 16 are only available in high-resolution (single colour) mode. SEUCK is designed to only use the multicolour mode.
It's wise to choose your two multi-colours from the "other" eight colours in the palette. For example, it would be a waste to choose white as the "light" multi-colour because you can select white as a changeable colour anyway. You'd be better off with light grey or green instead.

SPACED OUT

There's a specific trick that works with the black backgrounds commonly used for space games. Select the changeable colour as black throughout the game. This allows you to use the background as a "real" colour, and it can be any of the 16 in the palette. This does mean, though, that you'll be limited to the same three colours throughout the game!

SOUNDS GOOD

The accompanying blasts, blips and bleeps of a SEUCK game are easily made with "The Kit" - once you know what the abbreviations mean and what functions they describe. (See FIGURE 6, below).

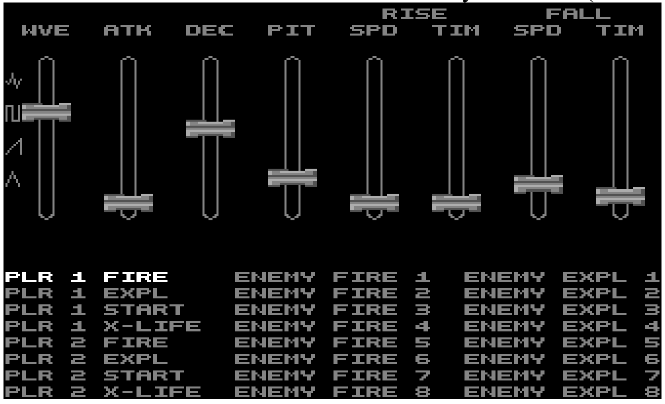


FIGURE 6 - Sound Effects Editor

- WVE - the type of waveform you are using, these are from top down: RANDOM NOISE for explosion effects, PULSE and SAWTOOTH for siren effects, TRIANGLE for whistle and flute effects.
- ATK - determines how fast the effect rises to its peak, and DEC(AY) determines how fast the effect falls from this peak.
- PIT - determines the frequency, i.e. how low or high the note sounds, of the effect
- RISE - is the effect where a sound starts at a selected PITCH and then rises until it reaches a high point. Then it returns to the starting pitch and repeats the cycle. This is reversed for the FALL option.
- SPD - these two determine the SPEED at which the RISE or FALL option happens.
- TIM - determines how long the effect runs before moving to the opposite effect, such as jumping from RISE to FALL and back again.

It's easier to get your head around with an example, so here goes. To simulate a police siren you'd select the PULSE waveform, then move the ATTACK and DECAY sliders to the top. The PITCH should be in the middle and both SPEEDS must be set to slow (about 1cm from the bottom). Move both TIME settings to an equal point on the scale. The combination of these settings should create a wailing sound that rises and falls like a siren.

UNDER ATTACK

When planning how to kill your player it's worth remembering that if you position too many enemy sprites onscreen at one time, this will cause sprite flicker. This happens because the VIC chip can only position a maximum of eight different sprites onscreen at once. "So how come I've seen SEUCK games with 32 sprites onscreen at once then?" I hear you cry.
Normally you can only have eight sprites at once, but by using the IRQ Raster Interrupt the actual limitation is for eight sprites on any horizontal raster line. This type of routine is called a multiplexor and SEUCK has one built in.
SEUCK's multiplexor can handle about 32 sprites onscreen (four lots of eight). If there are nine sprites or more on any horizontal raster line then flicker will occur. Try to space your enemies out more and avoid too many enemies crossing each other's paths horizontally. Remember that the bullets are sprites too.

ON THE LEVEL

In a highly eco-friendly fashion, each level can actually be used more than once. Parts of them could be recycled to form another, later level. Imagine you had six levels and wanted to make another four. You could use parts of the first six levels to construct these other four instead of using up yet more map memory. The important thing to bear in mind is that the attack waves will be repeated as well as the map area.

AT THE SHARP END

The intro screen should be the last part of your game to be developed. It's here that you credit friends, relatives (the director, the producer, thanks luvvies... - Oscar). It's handy to note that your 'credits' may not use all the character set. This means there may be a little room in the character set for a picture to represent the game. For example you only need three numbers for the date (2,0 and 5) leaving seven other numbers for your modification. Using these and any spare letter characters you should be more than able to create a picture that can brighten up your title screen. (FIGURE 7 below shows an enhanced title screen from the game TWIN TIGERS, not possible in SEUCK without extra machine code).

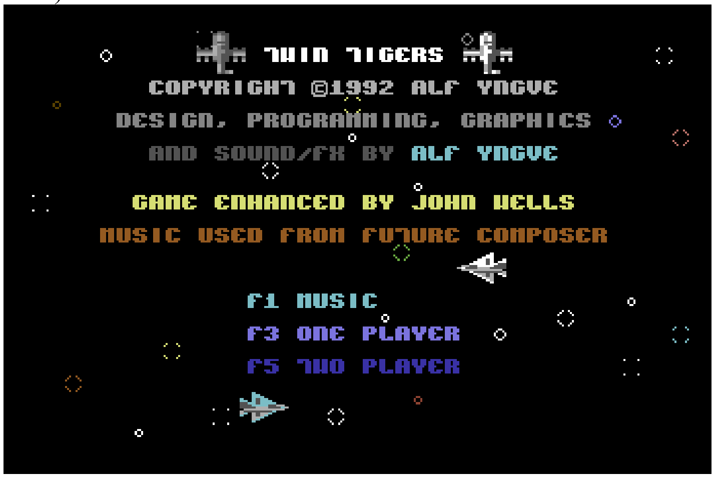

TWIN TIGERS
COPYRIGHT ©1992 ALF YNGVE
DESIGN, PROGRAMMING, GRAPHICS
AND SOUND/FX BY ALF YNGVE
GAME ENHANCED BY JOHN WELLS
MUSIC USED FROM FUTURE COMPOSER
F1 MUSIC
F3 ONE PLAYER
F5 TWO PLAYER

FIGURE 7 - Twin Tigers by Alf Yngve and Jon Wells

PLAYER LIMITATIONS

I say, I say, I say. When is a shoot-'em-up not a shoot-'em-up? When you've fiddled with the player limitations! (Erm, is that a joke, I don't understand - Ed.)

You can turn your blaster into a slasher just by tweaking the Bullet Duration. With this feature you can make your player 'appear' to use a short-range weapon - e.g. a sword, whip, dagger, club or fire (as in Monster Mash), rather than the traditional 'long-range' gun (see FIGURE 8). The lower the number of the Bullet Duration, the shorter distance your weapon (the bullet) will travel.

As an experiment, create a male/female sprite and then create a 'sword' sprite that will be the Player's Bullet Object. Set the Bullet Speed to two and the Duration to about seven in Player Limitations. Then, when you play the game, your player will appear to be swiping with the sword instead of shooting it. You may have to alter the speed or duration to get the best effect but the principle remains the same.

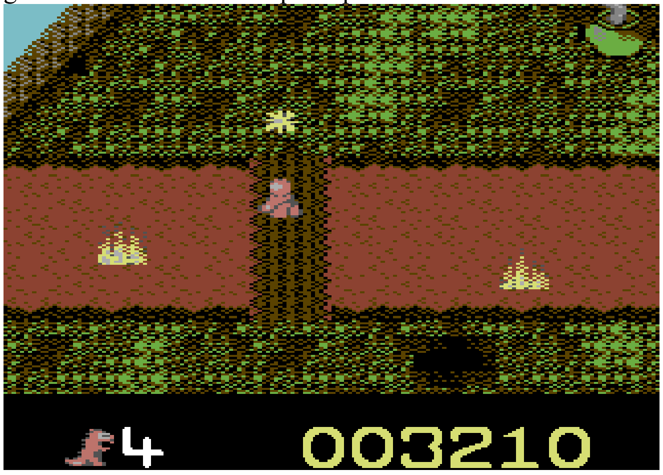


FIGURE 8 - Pink monster breathes fire

A FEW EXTRA TIPS

- Work out what's so good about your favourite games and try to include these elements in yours - but don't resort to copying, 'Cos that's just too sad.
- Ensure that at least one new enemy appears on every level.
  If all the enemies are on the first level there's little incentive to play on.
- Make each level look different or players will soon lose interest.
- Make backgrounds as colourful as possible. This may mean purple trees and blue cliffs, which theoretically should look silly, but they often look rather neat.
- Design your graphics on paper first.

FACT FILE

- Shoot-'Em-Up Construction Kit was released on budget by GBH
- Monster Mash is an enhanced SEUCK game by Jon Wells, released on Commodore Format Power Pack 31
- Twin Tigers by Alf Yngve and enhanced by Jon Wells, was released on Power Pack 25

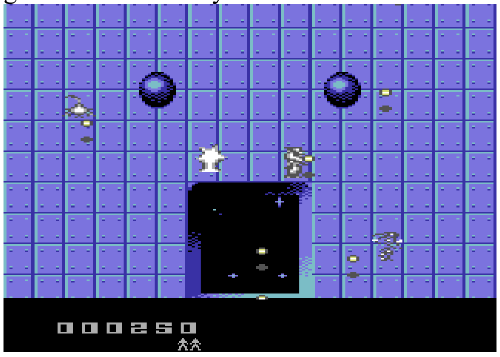
TOP 10 STEPS TO SEUCKCESS
By Simon Forrester

This article first appeared in Commodore Format issue 45.
Aliens have captured CF's regular SEUCKster, Jon Wells, this month, so Simon Forrester wrote his feature, while Dave Golder rushed off to conquer the alien race and rescue him.
Loads of people write SEUCK games - I know because I get buried daily in a pile of Jiffy bags full of covertape submissions. It looks as if quite a few people have got the idea of writing good SEUCK games from the previous Secret of SEUCKCESS features, but there are still a few left to reach. Coming up, then, are ten things you have to remember when writing your masterpiece:

1. SCENARIO
Decide on a scenario. Standard alien-bashing games are never going to capture the imagination of either the player or the reviewer (if your game gets that far), but a nice, novel plot does wonders. Check out some of the more recent re-hashed covertape game plots if you want a good example.

2. MAP
Keep your map simple - it's very difficult for the player to navigate, steer and shoot things at the same time. If you plan to give the player a choice of routes, make sure both are possible (just make one more difficult than the other), as they won't thank you for letting them play for five minutes only to come to a dead end.

3. SCROLL
The screen doesn't always have to scroll. If you want to keep the player interested, have them stop occasionally to fight a larger enemy or an end-of-level guardian. There's no point having nice big aliens floating around if you don't give the player the chance to stop, savour them, and kill them properly and with style, instead pushing them forward past the guardian before they've had a chance to let loose the ammo.

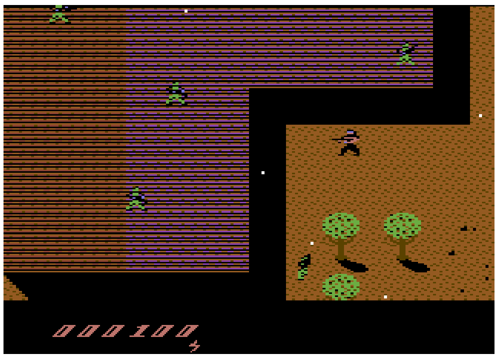


Alien Smash by Martin Edwards
I thought Alien Smash was a reference to those robots that used to advertise mash potatoes.

4. ENEMY SOUND
Try to give each different sort of enemy its own particular sound effect - it's always nice to have different things making different sounds, and it'll keep the player interested for a hell of a lot longer if you give them variation when they kill or collect something.

5. DIFFICULTY
Don't make your game too difficult to begin with. If the player gets killed twice within the first five seconds but then has to spend several minutes flying past a dull, featureless landscape, they're not going to thank you. Face it - they're not going to like you either. Build the difficulty level up slowly and let the player learn from their mistakes (it's called a sensible learning curve).

6. PLAYTESTING
Get someone else to playtest your game - if you've written a game then you're obviously going to be very good at it, but if you don't get a second opinion, you'll never know how good it is, or which bits need to be changed.



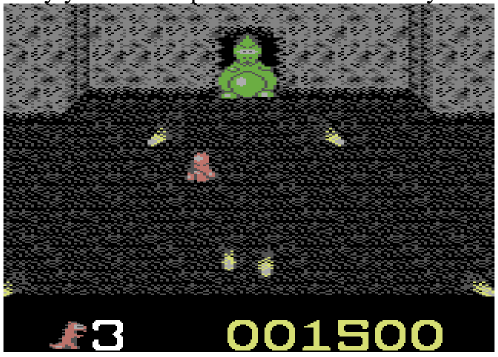
Chaos Road by Martin Edwards
It's complete Chaos... or is it? You should think carefully about the positioning of your enemies.

7. SPRITES

Always take time to draw your sprites - they are one of the most important elements in the game when it comes to impressing or pleasing the player, and a variety of interesting enemies keeps them interested for longer. It also helps if you draw your sprites on a black background - it looks better.

8. ATTACK PATTERNS

Vary your attack patterns. If each enemy attacks in the same way, your player will fall asleep from sheer boredom.



Monster Mash by Jon Wells
Decent end-of-level guardians are a must.

9. OBJECTS

Use the object editor. Moving objects that are essentially irrelevant show the player that you've spent a lot of time worrying about attention to detail. For example, if the player gets to fly past a crowd cheering them on, they'll know you've spent time on this game.

10. HUMOUR

Finally, try to use a sense of humour. Computer games are supposed to be fun, and if you recognise this, so much the better. It's easier for an enemy to kill a player if the enemy happens to be a tin of pickled yams flying at them screaming 'banzai' or something.

ASK THE PEOPLE

I stood outside Sainsbury's in Bath and asked a few people what they thought the problem was with most SEUCK games these days. Here's what they had to say:

- Mrs Queegee Bananahoe, 67. "I don't like the way them shooty games don't let you take a breather. It'd be nice to stop moving forward and spend a few minutes making a cuppa, or shooting the giblets out of some gigantic mutant space turkey."
- Lionel Blair, popular entertainer. "Some people always use the standard alien attack patterns, and it's boring (does a little tap dance). What am I famous for, anyway?
- Cyril Bristlecocker, nine. "I don't like the way the kernel engine is programmed. Imagine not creating a form of index register using a second stack? The whole thing must run so many nanoseconds slower than it could you know. Words fail me..."
- Roger Obviouslymadeupname, 35. "I think it's appalling. It should be banned, as I said in my letter to my MP, the Queen, and Philip Schofield. Am I going to be on the telly then? What exactly is a SEUCK game, by the way?"

SECRET OF SEUCKCESS by Jon Wells - Part 1

This article first appeared in Commodore Format issue 39.
Want to learn how to shoot-'em-up? JON WELLS has been tinkering with the Shoot-'Em-Up Construction Kit and he has this fabulous report...
This feature is dedicated solely to the Shoot-'Em-Up Construction Kit. In the three parts we'll be looking at endless possibilities to enhance your SEUCKed games. We're going to be looking at incorporating BASIC for the title screens, high-score tables and end sequences.
In this particular introductory feature we'll be covering the basics, just to get you started. Next follows two listings that insert a reset option in your finished game (saved from SEUCK). The first is for cassette users, the second for disk.

TAPE LISTING (PRG file)

```
1 REM ** SEUCK TAPE RESET HACK **
2 C=0:FORA=49152TO49273:READB
3 POKEA,B:C=C+B:NEXTA:PRINTCHR$(5);CHR$(147)
4 IFC<>14152 THEN PRINT"DATA ERROR!":GOTO 7
5 PRINT"SAVE HACK FOR FUTURE USE"
6 PRINT"TYPE SYS49152 TO START.."
7 PRINTCHR$(154):END
10 DATA 162,0,189,16,192,157,0,1,232
11 DATA 224,105,208,245,76,0,1,32,86
12 DATA 245,169,16,141,87,4,169,1,141
13 DATA 88,4,76,252,3,169,76,141,217,68
14 DATA 169,16,141,218,68,169,96,141
15 DATA 219,68,141,99,66,169,41,141,98
16 DATA 66,169,53,133,1,141,156,68,162
17 DATA 40,189,66,1,157,16,96,189,48
18 DATA 253,157,224,231,202,16,241,76
19 DATA 72,64,169,1,208,3,76,6,69,120
20 DATA 169,55,133,1,169,0,141,32,208
21 DATA 169,11,141,17,208,76,226,252
22 DATA 162,32,189,224,231,157,48
23 DATA 253,202,16,247,76,58,68,189
```

DISK LISTING (PRG file)

```
1 REM ** SEUCK DISK RESET HACK **
2 C=0:FORA=49152TO49281:READB
3 POKEA,B:C=C+B:NEXTA:PRINTCHR$(5);
4 IFC<>14836 THEN PRINT"DATA ERROR!":GOTO8
5 PRINT"SAVE HACK FOR FUTURE USE":PRINT
6 PRINT"TYPE SYS49152 TO START.."
7 PRINT"THEN LOAD GAME AS NORMAL"
8 PRINTCHR$(154):END
10 DATA 162,0,189,16,192,157,0,1,232
11 DATA 224,113,208,245,76,0,1,169,11
12 DATA 141,40,3,169,1,141,41,3,96,169
13 DATA 24,141,77,4,169,1,141,78,4,76
14 DATA 237,246,169,76,141,217,68,169
15 DATA 16,141,218,68,169,96,141,219,68
16 DATA 141,99,66,169,41,141,98,66,169
17 DATA 53,133,1,141,156,68,162,40,189
18 DATA 74,1,157,16,96,189,48,253,157
19 DATA 224,231,202,16,241,76,72,64,169
20 DATA 1,208,3,76,6,69,120,169,55,133
21 DATA 1,169,0,141,32,208,169,11,141
22 DATA 17,208,76,226,252,162,32,189
23 DATA 224,231,157,48,253,202,16
24 DATA 247,76,58,68,189
```

Once you've entered the relevant hack, it would be wise to SAVE it so you don't have to type it in again. The tape version is easy to operate - simply enter SYS 49152, press Return and press PLAY on tape.

The disk version however is a little more long-winded. First enter SYS 49152 - this installs an interrupt to access the disk. Now load the finished game with the usual command, e.g. LOAD "MONSTER MASH",8,1 - under no circumstances should you reset your machine, because if you do the hack will be wiped from memory. You must remember that these hacks only work with finished SEUCK games - those that are saved with the construction kit's loader. The listings won't work on modified games, like the ones from the Commodore Format Power Packs.

START TWEAKING

Once the game has loaded and the title screen appears, you can return to BASIC by hitting the RESTORE key. Don't worry, all the game's code will still be in memory. To prove this, type SYS 16595, which will re-enter the game.

Right, now you know how to restart the game, we can start doing some little bits of code. Hit RESTORE again to enter BASIC, and let's change the colour of your player. This is in location 11411 ($2C93 hexadecimal) for player 1, and 11471 ($2CCF) for player 2. To change this we POKE (or store) the memory location with the relevant numbers.

To change the colour of player 1, use the following POKE:
POKE 11411,(PEEK(11411)AND240)+COLOUR
Where COLOUR is in the range 0-15.

Re-enter the game with SYS 16595 and hey presto! Play the game and you'll find that the player 1 sprite is a new colour... Well, I know it's nothing outstanding, but it's a start and it can only get better.

To gain infinite lives, POKE 19308, 173 for player 1 or POKE 20092,173 for player 2. (That's hex $4B6C for player 1 and $4E7C for player 2). To re-enable lives, change the 173s in both POKEs to 141 instead.

If you want to have a real laugh then change the player's speed. POKE 16550,8 for player 1 and POKE 16569,8 for player 2. (Hex locations $40A6 and $40B9). The POKE speed doesn't have to be 8, you could choose a higher or lower number as well.

The amount of lives is held at 16557 ($40AD) for player 1, and 16575 ($40C0) for player 2. The highest these locations can hold is 9 and no more.

The player collision values (deciding which character stops/kills the player) are held at 16556 ($40AC) and 16575 ($40BF). POKE these locations with 255 ($FF) to enable the player to move over all background characters.

(More locations are in the MEMORY MAP in part 3)

PICK A COLOUR

Some of the in-game colours can also be easily changed.

To change the SPRITE MULTI-COLOUR #1:
POKE 16513, colour (hex $4081)
Remembering that the colours range from 0 to 15 only.

Other locations are:
16515 ($4083) - SPRITE MULTI-COLOUR #2
17427 ($4410) - BORDER COLOUR
16517 ($4085) - BACKGROUND COLOUR #1
16518 ($4086) - BACKGROUND COLOUR #2
16519 ($4087) - BACKGROUND COLOUR #3

Next follow a couple of small BASIC programs, but first a word of warning. Don't attempt to include large listings of your own. This is because the start of BASIC is at 2049 ($0801) and the game code starts at 2304 ($0900) - this gives us only 254 bytes of BASIC to play with at the moment, and a larger program would overwrite valuable code.

Anyway, now let's fiddle about with the NMI (Non-Maskable Interrupt). We'll do a simple "flashing colour" program, which will give you an idea of the extra routines that can be incorporated into a SEUCK game.

Type in and RUN this BASIC program:

```
1 FOR A=24624TO24629:READB:POKEA,B:NEXTA
2 POKE17668,48:POKE17669,96
3 DATA238,32,208,76,148,92
```

Enter the usual SYS 16595 to restart, and when you start the game the BORDER will flash repeatedly while you play. Now press RESTORE and POKE 24625,33 - this time BACKGROUND COLOUR #1 will flash.

The number codes to POKE into this routine are:
32 - BORDER
33 - BACKGROUND COLOUR #1
34 - BACKGROUND COLOUR #2
35 - BACKGROUND COLOUR #3
37 - SPRITE MULTI-COLOUR #1
38 - SPRITE MULTI-COLOUR #2

To disable the flashing routine, POKE 17668,148:POKE 17669,32. (You must enter both POKEs).

GREAT MEMORIES

So how's it done? Well, the sound effects routine is jumped to from location 17668-17669 ($5403-$5404), and we've changed this with our listing. So instead of playing the sound effects, it jumps to a new routine at 24624 ($6030). This first flashes the colour and finishes by jumping to the actual sound effects routine in location 23700 ($5C94).

And so we'll move onto a more advanced routine. Type in and RUN this BASIC listing:

```
1 FOR A=24640TO24657:READB:POKEA,B:NEXTA
2 POKE16829,64:POKE16830,96
3 DATA32,148,92,206,71,96,169,2,208
4 DATA5,169,0,32,51,92,76,56,65
```

Right, so what does it do?
Well, turn up the volume on your TV and restart the game (with SYS 16595). Now listen. If you've typed it all in correctly you should hear sound effects on your title screen. You can play any of the 24 sound effects that are available in the kit. So, to test this, reset by pressing RESTORE and enter POKE 24651,0-23 (number of the sound to play). Use a value of 24 to play no sound.

The number formation goes like this:
0-7 are the player SFX
8-15 are the enemy SFX
16-23 are the enemy explosion SFX

It would be a good idea to make your "title effect" in enemy explosion 8 (sound effect 23). You could make a warble or low thud sound - this would give you an eerie effect on the title screen.

One last thing - to disable the RESTORE option, POKE 24593,0. Be very careful though, because once you restart the game you'll have no way to reset again.

SECRET OF SEUCKCESS
by Jon Wells - Part 2

This article was first published in Commodore Format issue 41,
and reprinted in issue 43.
In part 1 we covered the basics of getting into our SEUCKed games. Now we'll go one step further - incorporating BASIC into the finished games. This will enable us to use the 64's BASIC programming language to display better title screens, add high score tables and end sequences or even incorporate intro sequences, get ready and game over parts. Before we start though, you will need the hack listings from part 1.

BEFORE WE START

First of all, use the Hack listing and load in your SEUCKed game. If you remember, hitting RESTORE will exit you to BASIC. At this point we will now install the BASIC driver.

When a finished SEUCK game is saved, the editor is saved with it, but the finished program does not use it. The SEUCK editor is over 20K from locations $6500 to $8680. What a waste of memory. If there was only some way to use this memory for extra presentation... Well, that's exactly what these next few listings do.

Now enter the following fab listing:

Before we type them in though, enter POKE 21745,87. This POKE moves the random fire from a routine in the editor to a part of the main code.

LISTING 2A (PRG file)

```
5 FOR A=25856 TO 25873:READ B:POKE A,B:NEXT A
10 DATA 162,0,189,0,9,157,0,105,189,0
11 DATA 10,157,0,106,232,208,241,96
```

SAVE it for future use, RUN it and enter SYS 25856. Type NEW and this gives us a bit more BASIC space to enter a bigger listing as follows:

LISTING 2B (PRG file)

```
5 C=0:FOR A=25856 TO 25983:READ B:POKE A,B
6 C=C+B:NEXT A
7 IFC<>14459 THEN PRINT"DATA ERROR!":END
10 DATA169,0,141,90,94,141,16,72,141
11 DATA131,72,141,51,73,169,128,141,21
12 DATA72,141,126,72,141,46,73,164,224
13 DATA141,91,94,141,17,72,141,22,72
14 DATA141,127,72,141,132,72,141,47,73
15 DATA141,52,73,169,121,141,27,72,141
16 DATA122,72,141,212,73,169,122,141
17 DATA89,94,169,105,141,215,72,141,227
18 DATA71,141,187,73,160,11,162,0,140
19 DATA94,101,140,102,101,152,24,105,96
20 DATA141,97,101,189,0,37,157,0,133
21 DATA169,0,157,0,37,232,208,242,200
22 DATA192,38,208,222,189,0,38,157,0
23 DATA134,169,0,157,0,38,232,224,128
24 DATA208,240,96
```

Remember to SAVE this listing as well. RUN it and type SYS 25856 again. This amazing listing has just relocated all of the map, colour and block data over the editor, and also changed the pointers in memory to read this. One more listing to go and we're nearly there. Enter NEW again and type in this listing:

LISTING 2C (PRG file)

```
5 C=0:FOR A=25856 TO 26031:READ B:POKE A,B
6 C=C+B:NEXT A
7 IFC<>20475 THEN PRINT"DATA ERROR!":END
10 DATA162,0,189,0,0,157,0,225,232
11 DATA208,247,169,0,141,32,208,169,51
12 DATA141,94,66,169,101,141,95,66,169
13 DATA96,141,0,65,32,211,64,32,3,96
14 DATA32,41,66,32,233,67,169,208,141
15 DATA231,67,76,96,66,32,211,64,169
16 DATA129,141,13,220,169,240,141,26,208
17 DATA169,127,141,13,221,169,151,141
```

```
18 DATA0,221,169,63,141,2,221,169,4,141
19 DATA136,2,169,21,141,24,208,169
20 DATA8,141,22,208,162,2,189,0,225,157
21 DATA0,0,232,208,247,169,127,141,0
22 DATA220,169,255,141,2,220,141,7,220
23 DATA141,4,221,141,5,221,141,6,221
24 DATA141,7,221,169,8,141,14,221,141
25 DATA15,221,169,55,141,18,208,169
26 DATA113,141,25,208,169,27,141,17,208
27 DATA169,55,133,1,88,96,169,127,133
28 DATA51,133,53,133,55,169,38,133
29 DATA52,133,54,133,56,96
```

Once that's all typed in, SAVE it and RUN it. Then you must enter SYS26015, which allocates the original map locations into BASIC locations and installs BASIC into SEUCK.

STARTING THE GAME

From now on you must enter SYS 25856 to run your SEUCK game without the normal title screen. If you quit or die, the SEUCK game will return back to BASIC from where it left off.

If you are playing about with one of the SEUCK demo games, enter POKE 16964,0 to stop the game from returning to the editor when the Space bar is pressed.

And from now on you won't need the reset hack, so enter POKE 24593,0 to disable it.

If you are using a two-player game, then POKE 16578,0 or 1 starts a one or two player game respectively. Also, in one player mode, you may notice that pressing the fire button on player two's joystick results in no second player appearing.

As we no longer need the data listing, type NEW to clear the program. Now let's check to see how much BASIC memory we've got. To do this type the following:

```
PRINT FRE(0)-(FRE(0)<0)*65536
```

If all goes well you should have about 7800 bytes, which is about 8K of BASIC RAM; more than enough for what we want. There's still plenty of memory left from the editor - 5.5K from $A000-$B6B0 which can be used for music, and 7K from $6600-$6900/$8680-$A000 which we can use for add-on machine code.

A BASIC DEMONSTRATION

Now for a small BASIC program to demonstrate how you can add to the game. Type this in and RUN it.
TITLE SCREEN LISTING (Click to download the PRG file)

```
1 REM ** TITLE AND GAME OVER DEMO **
2 PRINTCHR$(147);CHR$(19);CHR$(5);
3 FORA=0TO200:NEXTA
4 PRINTTAB(2);"MONSTER MASH COPYRIGHT 1993 C-FORMAT"
5 GOSUB10
6 PRINTCHR$(147);CHR$(152):SYS25856
7 PRINT"{down}{down}{down}{down}{down}{down}{down}{down}{down}"
8 PRINTTAB(16);"GAME OVER"
9 GOSUB10:GOTO2
10 PRINT:PRINT:PRINTCHR$(150);
11 PRINTTAB(9);"PRESS SPACE TO CONTINUE"
12 GET A$:IFA$<>CHR$(32) THEN GOTO12
13 RETURN
```

(Replace the {down} in line 7 with CRSR downs; there should be 9 of them.)

Nothing special I know, but it should give you a rough idea of how it all works.

In part 3 we'll look at the memory map and adding a high score table.

SECRET OF SEUCKCESS
by Jon Wells - Part 3

This article was first published in Commodore Format issue 45.
If you've got the Shoot-'Em-Up Construction Kit you've got a programming tool that makes writing vertically scrolling blasters a doddle. But getting it to produce really good vertically scrolling blasters takes more skill. Stick with Jon Wells, though, and you'll develop the skill.

Blimey, I bet you're thinking, look at all those figures. Let's skip this. But hold on, because these tables are your key to complete control when it comes to setting up in-game features in SEUCK games. This time we're going to look at the memory map which features the locations of some large chunks of game code, as well as the all-important in-game values you'll find within SEUCK. For instance, if you wanted to change the amount of lives player one has, you'd just POKE the new value into the memory address you'll find in the following tables.

MEMORY MAP 1 - MAJOR SECTIONS

DESCRIPTION	Decimal location (BASIC)	Hex location (machine code)
Spare memory	1024-2303	$0400-$08FF
Map background scroll data	2304-6399	$0900-$18FF
Block colours (128 bytes)	6400-6527	$1900-$197F
Spare memory	6528-6655	$1980-$19FF
Block data	6656-9727	$1A00-$267F
Scrolling machine code	9728-11391	$2680-$2C7F
Object pointers	11392-12803	$2C80-$3203
SEUCK title screen	12804-13283	$3204-$33DF
Attack wave patterns	13284-16383	$33E0-$3FFF
Main game machine code	16384-25855	$4000-$657F
SEUCK editor machine code	25856-46719	$6580-$B67F
Sound FX data	46720-49151	$B680-$BFFF
Sprite graphicsv	49152-57343	$C000-$DFFF
SEUCK editor text	57344-59391	$E000-$E7FF
Game screen page 1	59392-60415	$E800-$EBFF
Game screen page 2	60416-61439	$EC00-$EFFF
Scroll and score panel code	61440-62463	$F000-$F3FF
SEUCK editor charsets	62464-63487	$F400-$F7FF
Game background charset	63488-65527	$F800-$FFF7

There are also some very important locations dealing with the controls of the two players:

MEMORY MAP 2 - PLAYER OPTIONS

DESCRIPTION	Player 1	Player 2
Player enabled	16559 / $40AF	16578 / $40C2
Amount of lives	16557 / $40AD	16575 / $40C0
Speed of player	16550 / $40A6	16569 / $40B9
Player main colour	11411 / $2C93	11471 / 2$CCF
Amount of bullets	16551 / $40A7	16570 / $40BA
Bullet speed	16553 / $40A9	16572 / $40BC
Directional fire	16552 / $40A8	16571 / $40BB
Bullet duration	16554 / $40AA	16573 / $40BB
Character collision	16556 / $40AC	16575 / $40BF
Die or stop (0/1)	16555 / $40AB	16574 / $40BE
Extra life at 10,000	16558 / $40AE	16577 / $40C1
Horizontal start position	16547 / $40A3	16566 / $40B6
Horizontal 256 bit point	16548 / $40A4	16567 / $40B7
Vertical start position	16549 / $40A5	16568 / $40B8
Play area up stop	16543 / $409F	16562 / $40B2
Play area down stop	16546 / $40A2	16565 / $40B5
Play area left position	16541 / $409D	16560 / $40B0
Left 256 bit point	16542 / $409E	16561 / $40B1
Play area right position	16544 / $40A0	16563 / $40B3
Right 256 bit point	16545 / $40A1	16564 / $40B4

Extra important SEUCK addresses include:

MEMORY MAP 3 - SCORES AND ROUTINES

DESCRIPTION	Decimal (BASIC)	Hex (machine code)
Score player 1	24227-24232	$5EA3-$5EA8
Score player 2	24233-24238	$5EA9-$5EAE
Score clear	17446-17466	$4426-$443A
Score sprite plot	23077-23253	$5A25-$5AD5
Player 1 lives plot	23253-23357	$5AD5-$5B3D
Player 2 lives plot	23357-23461	$5B3D-$5BA5
Score add routine	23461-23561	$5BA5-$5C09
Clear and set SID chip	23561-23603	$5C09-$5C33
SFX initialise	23603-23700	$5C33-$5C94
SFX play	23700-23906	$5C94-$5D62
Player 1 char collision	19083-19192	$4A8B-$4AF8
Player 2 char collision	19866-19975	$4D9A-$4E07

I LIKE TO SCORE

The player's score pointers are stored in a 6-byte series; from 24227-24232 decimal ($5EA3-$5EA8) for player 1, and 24233-24238 ($5EA9-$5EAE) for player 2.

To convert this into BASIC you can use the following routine:

HIGH-SCORE LISTING 1 (PRG file)

```
1 S1=0:S2=0:FOR A=24227 TO 24232
2 READ B:S1=S1+(PEEK(A)*B)
3 S2=S2+(PEEK(A+6)*B):NEXTA
4 PRINT"PLAYER 1 = ";S1:PRINT"PLAYER 2 = ";S2
5 DATA 100000,10000,1000,100,10,1
```

Note: in its current form, this listing will work once. In theory it needs a RESTORE command added, so that the DATA can be READ again. I will provide an updated version in due course - Andrew.

If you are using a two-player game, then POKE 16578,0 or 1 starts a one or two player game respectively. Also, in one player mode, you may notice that pressing the fire button on player two's joystick results in no second player appearing.

When using this program, S1 and S2 refer to the two players' scores. You could use these figures to calculate a high-score table if you wanted to, using something a little bit like this:

HIGH-SCORE LISTING 2 (PRG file)

```
10 B=9:DIM H1(10),N1$(10): REM FOR 1 PLAYER
11 IF S1>>H1(B) THEN B=B-1:GOTO10
12 B=B+1:FOR A=9TOBSTEP-1:H1(A)=H1(A+1)
13 N1$(A)=N1$(A+1):NEXTA
14 H1(B)=S1:N1$(B)=NAME$
```

This might be a little difficult to follow, so let's go through the program line by line:

10 - Here we set up an array to hold 10 different names (N1$) and scores (H1)
11 - Next we compare our new score (S1) with the old ones (H1), to find out whether the next score in the table
is higher than our new score (in Dave Golder's case, it's usually lower).
12 -
13 - These lines then move the scores and names down one position in the list,
to make room for our new score.
14 - Finally we store the new name (NAME$) and score (S1) into the array.
This name would be obtained by asking the user to enter it (as with just about any other game ever -
what do you expect them to do, communicate with the game telepathically?).

The original scores and names would be set up first with DATA statements. The DIM in line 10 simply tells the computer to allocate memory to hold the values and names. If that all seems complicated, then here is an example of setting up the table:

HIGH-SCORE LISTING 3 (PRG file)

```
20 FORB=1TO10
30 READN$,S
40 H1(B)=S:N1$(B)=N$
50 NEXT B
55 FORC=1TO10:PRINTN1$(C),H1(C):NEXT
60 DATA "JON",50000
70 DATA "DAVE",40000
80 DATA "ANDREW",30000
90 DATA "WAYNE",20000
100 DATA "RICHARD",10000
110 DATA "ALLAN",5000
120 DATA "SHAUN",2500
130 DATA "NIGEL",1250
140 DATA "SIMON",675
150 DATA "PLAYER 1",0
```

Remember to put your names inside quotes so any spaces can be handled.
(This is an extra listing added by Andrew.)

That completes the three parts of the article originally published in Commodore Format, updated and edited here to remove some printing mistakes, and to place the debugged listings from part 3 into part 1.

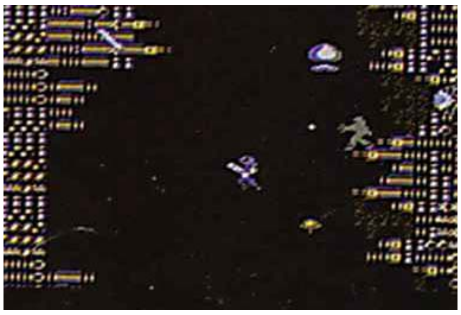
You find the Listings in the File **“Secret of Seuckcess - Listings.d64”**

HOW TO BE A MEGASTAR
By Ian Osborne

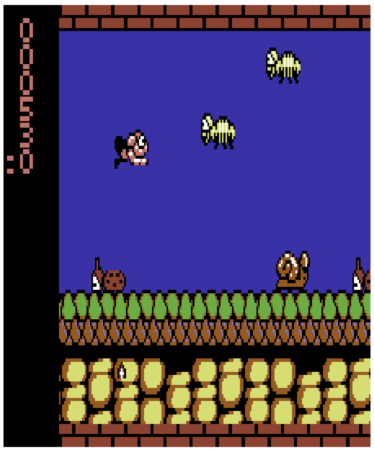
This article first appeared in ZZAP! issue 87.
Had your latest entry for the ZZAP! Megatape rejected? Feel like throwing yourself off a tall building in despair? Don't do it - find out where you went wrong with IAN 'KNOW-IT-ALL' OSBOURNE.
Every morning's the same here at ZZAP! Towers. The postman brings the mail in a huge, back-breaking sack packed with well-wild game tips, snazzy letters, crazy compo entries and requests for offers that finished years ago (don't laugh - we do get them!).
If we're really lucky we might get a reader's game submitted for the Megatape - or should that read 'unlucky'? As you've probably guessed, only rarely do reader games come up to Megatape standard, so to aid your careful coding we're doing a round-up on games that didn't quite make the grade - so pay attention... I might be asking questions!
Unsurprisingly most reader games are written on utilities such as the Shoot-'Em-Up Construction Kit. We're happy to assess utility games but remember, SEUCK sold by the sackload and if you can write a blaster on it, so can everyone else - to make your game stand out from the crowd you must do something original!

ORIGINAL SIN

Most SEUCK games are vertical scrollers, but don't forget you can also do flick-screen or push scrolling. Even if you go for a straightforward auto-scroll, it needn't be limited to forward firing. Asif A Bhagwandin's The Absolute Beginning featured eight-way firing that put Forgotten Worlds to shame, and the result was one of the best SEUCK games we've ever seen! If we had more room on the Megatape he'd have probably found fame and fortune as a ZZAP! Megastar, but as it stands his game's just a little too samey for publication. Great stuff though!

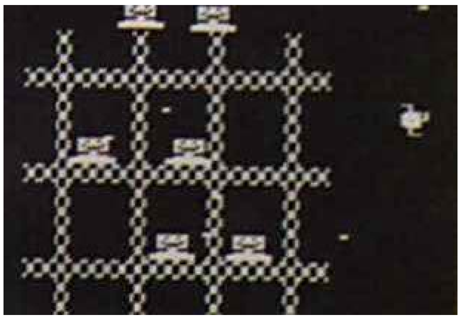

T.A.B. - Atmospheric, playable, interesting... everything a SEUCK game should be!

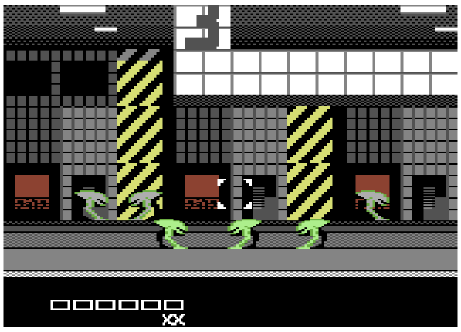
One of the most original SEUCK games I've seen was Shaun Pearson's Fly Pig, which was played as a horizontal scroller by turning your TV on its side! (DO NOT TRY THIS AT HOME!) Ten out of ten for free thinking, but it didn't really work as a game - because a TV screen is wider than it is tall, when on its side the playing area was just too narrow.


Shaun Pearson's strange sideways shooter

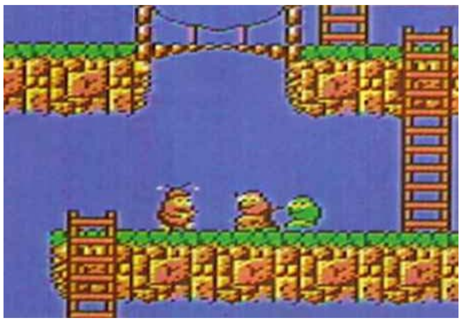
Up, up and... along? For all its originality, Fly Pig doesn't work as a game.
(What about as a bacon sarnie - Phil King).
Picture rotated 90 degrees to resemble original screenshot.

Klaboom by James Sweeney was another brave attempt. Played on a static screen dominated by an impenetrable backdrop, you had to move around it and blast away at 'computer sprite' baddies. Unfortunately the game as a whole was let down by blocky monochrome graphics and very dated presentation. Also worth a mention is Alf Yngve's Alienator 92, an Op-Wolf clone.


Klaboom - A potentially interesting game marred by dated presentation.


Alienator 92, Alf Yngve's brilliant Op-Wolf clone

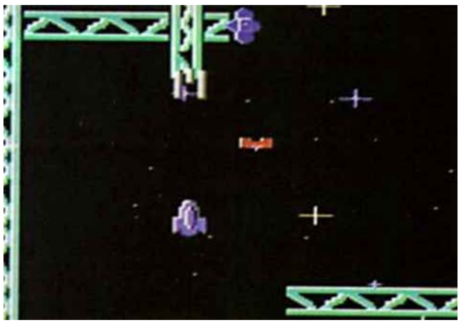
Although blasters lend themselves most easily to starship-and-aliens or Rambo-esque games, it doesn't have to be like this. Michael Williams produced a great little cutesy platformer in Cyber Squidgies, but effectively slit his own throat by 'borrowing' his sprites from Creatures (more on this later). And if you want original, what about Insects in Space? That featured a busty angel as the main sprite, and bees for baddies! When writing a game you should resist the temptation to just load the utility and tap away - a few minutes' thinking time is time well spent!


Cyber Squidgies - Creatures revisited? Nice game, shame about the sprites...

YA GOTTA HAVE STYLE

Have a think about your favourite Commodore games. What is it that makes 'em so fabby? They'll all have one thing in common, I'll be bound - FLAVOUR. So make sure your game has it too! When you've chosen your game theme, STICK TO IT! If you've gone for a spaceships-and-aliens approach, don't throw in a backdrop that looks like the Chelsea Flower Show, or put vicious-looking enemy sprites in a cutesy game. They'll just look out of place and ruin the atmosphere.

Without apologising for stating the obvious, the most important element in any game is the gameplay. It's also the iceberg that has sunk many a good SEUCKer - take Richard Thompson's Zanatron: a competent if undistinguished blaster with blimmin' gorgeous explosion sequences. Trouble is the enemy bullet sprites look exactly the same as the explosions, making the whole thing hellishly confusing!


Zanatron - Great explosions, but they interfere with the game!

Brian Hughes' Toys was colourful and fast, but the enemies had the habit of appearing onscreen unannounced, usually on top of the player - can you imagine how annoying that is? If you want sprites that make sudden appearances on the playing area, 'fade' them in, giving the player time to shift before they become active.

A tricky game made harder by the elaborate backgrounds
Toys - fast, colourful, but basically flawed.

When you've finished your game, let your friends have a blast on it and listen to their constructive criticisms. There's nothing worse than a game that's too easy or ridiculously hard, so pay special attention to difficulty - I've seen so many games that are so damned tricky you'll be hard pressed to get off the first screen!

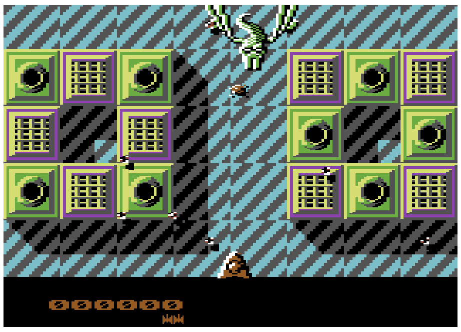
Letting your friends playtest your games will also help identify niggly strategic quirks too, such as the one in SEUCK game Tiger Hell. Superb graphics and outstanding backgrounds, but no gameplay --90% of the enemies fired forward at an incredible rate, making it impossible to nip between shots and fill 'em full of lead, but easy-peasy to fly round them and let them scroll off the screen. You could cruise through most of the game without ever firing a shot!

Tiger Hell - enhanced with a title screen, but let down by a lack of play-testing
Triad's good-looking but difficult helicopter game

If strategic quirks are annoying, programming bugs are bleedin' well inexcusable. If your game is bugged, don't release it until it's fixed! Equally annoying was the SEUCKed push-scrolling blaster set in a maze. The game lets you walk down a dead-end path that ended off the screen, and because the scrolling wouldn't let you retrace your steps you were trapped! This is a ridiculous, game-destroying bug that would've been exposed by the most rudimentary play-testing.

Less spectacular but all-too-common are those twin banes of shoot-'em-ups: bad sprite masking and the disappearing bullet syndrome. Just 'coz certain commercial releases think they can get away with this don't think you can - not if you want to make the Megatape anyway!
'ELLO, 'ELLO, 'ELLO!

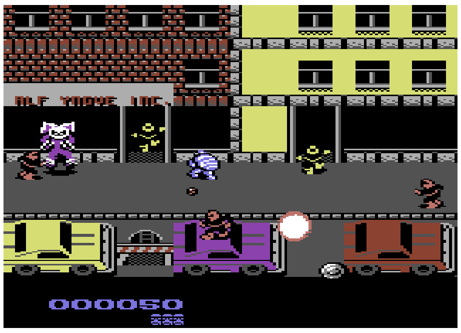
When you create a game, with or without a utility, it automatically becomes your copyright - which is why we ask you to fill in that Megatape form when sending it in. What some people fail to realise is that commercial games are also copyrighted! One individual (who shall remain nameless) sent in a game labelled 'Blow 'Em Away - the game they couldn't stop.' Sensible Software's lawyers could've stopped it - it was just a lightly tweaked rip-off of SEUCK demo Slap 'n' Tickle! (It wasn't the only one we've received either.)


Inpsired by the arcade game Slap Fight...
Don't even THINK about ripping off SEUCK demos! (Especially 'Slap 'n' Tickle'...)

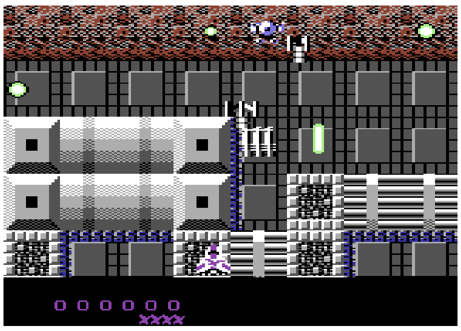
Not only is this sort of thing dishonest, it's also ILLEGAL. Coding your own 'tributes' to commercial games is also a no-no. A few months ago we received a great little game that we couldn't use 'coz it's gameplay was pinched from SWIV. If there's any hassles arising from this sort of thing it's the PROGRAMMER who's in trouble, not ZZAP!

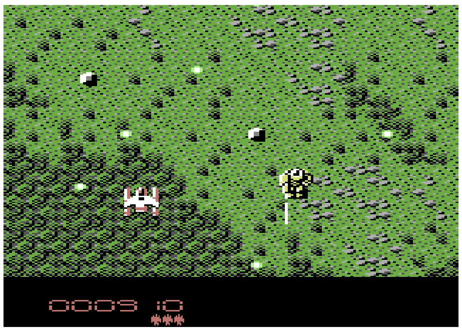
Arguably less dishonest but equally illegal is 'borrowing' a registered trademark as a game theme. One such offender is Alf Yngve's A.M.N.T. (Adolescent Mutated Ninjitsu Terrapins) which he himself describes as 'one of the funniest Mutant Turtle rip-offs ever.' Maybe, but companies like Konami spent a fortune on Turtles licenses, and they certainly wouldn't be laughing! It's a pity really as Alf's games (like Tau Omega and Tau Zero) are some the best SEUCK blasters ever. With just a little more playability they'd be tailor-made for the covertape.

Alf Yngve Inc. releases another great SEUCK game...



A.M.N.T. - What a game! Zzap! Towers were more impressed than the Turtles license holders would be.
(Alf changed the Terrapins to Blueberries for its PD release).


Tau Zero - I've seen commercial releases that aren't as fabby as this!


Which came first, Zero or Omega?
Tau Omega - Dig those crazy sprites!

<u>WELL, WHADYAWANT?</u>

There's nothing worse than a game that's ALMOST brilliant, as the faults that stand between it and perfection stick out like Phil King's gut. Like all utilities, SEUCK has its limitations, so to reach covertape standard a SEUCK game has to really stand out from the crowd.

- The graphics and animation must be first-rate, with crystal-clear sprites set against devastatingly picturesque backgrounds.The game as a whole must have heaps of atmosphere really drawing the player into the action.

- The gameplay must be well thought-out, not too tricky but not too easy. It's up to you to decide how steep the difficulty curve is, but make sure it has one.

- If the gamer is to load it more than once, it'll have to ooze variety - the further you get the more you should see, forcing you to have another go to see what comes next It's hard work I know, but well worth the effort in the end.

- The game as a whole needs that spark of originality to set it off against the other three million reader games we get sent. This is especially true if a utility like SEUCK is used.

- Finally, the game must be bug-free.We're not going to publish a game that doesn't work properly, so don't bother sending one in!

I know we're asking a lot, but it's your covertape too - do you really want to see it full of also-rans? Stick to these rules and you stand every chance of getting your game published! Cheerio, and happy SEUCKing.

THE SHOOT'EM UP CONSTRUCTION KIT EXTENSION by Ian Goffe

Published in Commodore Disk User volume 4 number 5, March 1991

The Shoot 'Em Up Construction Kit was highly acclaimed when it was launched in 1987. The kit contains, in my opinion, one of the best GRAPHICS creation tools available for designing SPRITES and BACKDROPS. I was wondering how many readers own a copy of this excellent tool, and seeing that a few readers' games have been developed with the utility - namely - PHANTOM, B-RAID and ATLANTIS (There are others), then many of you have obtained a copy methinks.

A LITTLE BACKGROUND

One of the main features of the kit is the background editor, in which a level is built up by simply placing predefined blocks of characters into a level map - the jigsaw piece technique - no less! The main principle for this article is to allow the user to load in a standard S.E.U.CK. background file from disk, in which you have created the necessary characters and blocks, that can then be slotted into place on anyone of 256 screens by using my editor. These full colour screens (well, one colour per block as in S.E.U.CK. of course) can then be used in your own programs by using the source file that I have included. It can be assembled by the MIKRO assembler, 6510+, and my very own 6510 MACRO ASSEMBLER (as published in 'YOUR COMMODORE').

BLOCK COMPRESSION TECHNIQUE

Allan Webb touched on this subject in his 'HIGH SPEED GRAPHICS' series that was also run in 'YOUR COMMODORE'. I intend to go one step further by presenting a comprehensive editor that complements the S.E.U.CK. The kit uses this compression technique, which is widely used for storing many screens worth of data in a compacted form. Each block that is created within S.E.U.CK. is made from a 5 by 5 matrix of characters, and each level is further made up of a set of blocks placed together, giving the overall effect of a full screen image. This technique of expanding blocks of characters onto the main screen has wide applications in games programs, where, on most occasions memory is at a premium. Each of my screens is 40 by 20 characters, giving a total of 800 bytes per screen. Therefore, normally for 256 screens this would need to use up 204,800 bytes of memory. (Amiga please!) But, by using this compression technique, we can store 256 screens worth of data in only 8192 bytes, now THAT'S SOME SAVING.

THE EDITOR IN USE

The editor is operated simply by a joystick in Port 2, and additional key-presses. A large BLOCK CURSOR, a la S.E.U.CK. is used to guide your way around the screens. The idea behind the Editor is that it allows you to flip between the 256 screens, and insert a relevant block onto that screen at the block cursor position. Before you can actually see anything though you will need some of your previously designed blocks of characters that you have originally created with S.E.U.CK. The S.E.U.C.K. background files (saved from S.E.U.CK. from the SAVE BACKGROUNDS option within the STORAGE menu - I'm sure I don't have to tell you that!) are loaded into the editor by pressing the 'L' key. You will be asked for the filename at this stage. The filenames from S.E.U.CK. are suffixed with a file type, in this case .B for backgrounds.

This can, however, be omitted from the filename that you input; only key in the main part of the name, e.g. GRAPHICS .B can be entered as GRAPHICS. This is achieved as the editor automatically adds an asterisk to the filename, therefore loading in any file that STARTS with the filename that you input. This is the same when loading in any screen data that you have created. (Worth mentioning, don't you think?). If you just want to get used to the editor than load up a demonstration editor, containing some blocks ready for use - the Blue Peter style, hey?

THE STATUS WINDOW

Below the main graphics area is the STATUS WINDOW, where the user communicates with the editor, and certain information is displayed. The current screen number that is being edited is shown, alongside the block number that is to be used for inserting onto the current screen. The value given alongside "HOLDING" refers to the block that currently lies underneath the block cursor.

KEY/JOYSTICK FUNCTIONS

The block number can be changed in two ways:
NOTE: The '/\' symbol equals the UP ARROW key.
1. Using the '/\' and '*' keys to add to or decrease the current block number.
2. Using the 'G' key to GRAB a block that is under the cursor - in this case the BLOCK NUMBER will be given the value contained in "HOLDING".

The screen number can be changed by using the '+' and '-' keys to move to the next screen in either direction. There are also key-presses available to dump a screen into spare and then, if needed, copy it to another screen. This function is operated by the 'F' and 'T' keys to copy 'FROM' and 'TO'. Press the 'F' key when on the source screen, and then move to the screen that you wish to copy to, at which point you press 'T'. This function can also be used as an 'UNDO' option. After storing the screen with 'F', you may make a complete mess of it, at which time pressing 'T' when on the same screen will bring back the original. (Better to be safe than sorry!). If you wish to clear out all screens by filling with a particular block then get the BLOCK NUMBER holding the desired block that you wish to fill with, then press the 'E' key to erase all screens. You will be asked for confirmation of this command, pressing any key other than 'Y' at this point will allow you to breathe easily once more - that's if you pressed by mistake, of course! Individual items can be saved out; these can then be loaded back for use in your own programs. The commands you require are:

S	Save screen data
A	Save attributes (Block colours)
B	Save block definitions
C	Save character set

Screens that you have designed within the editor can be loaded back by pressing the 'K' key. The colour can be changed using the FUNCTION keys, while the '@' key allows you to send disk commands, and view the disk directory - a DOS support is a good feature of any utility program, I believe.

KEY SUMMARY TABLE

+	Screen number + 1
-	Screen number - 1
/\	Block number + 1
*	Block number - 1
G	Grab block
F	Dump screen into spare
T	Copy spare to screen
E	Erase all screens with block number
A	Save block colours
B	Save block definitions
C	Save character set
S	Save screens
K	Load screens
L	Load S.E.U.C.K. background file
@	DOS support
F1	Increase background colour by 1
F3	Increase multi-colour 0 by 1
F5	Increase multi-colour 1 by 1
F7	Increase border by 1

HOW S.E.U.C.K. HANDLES BLOCK DATA

The main components of anyone block in S.E.U.C.K. are the 25 characters that make up the block - the BLOCK DEFINITION - and the individual block colours.
S.E.U.CK. stores the block definitions in a 3200 byte table (128 blocks * 25 characters). The first 25 bytes in this table hold the characters used to make up block 0, the next 25 bytes for block 1, etc.

Therefore to get to the base address of any one particular block definition, we can find the offset in the definition table easily by: -

OFFSET =(BLOCK NUMBER * 25).

The 25 characters used in a block are then stored from (BLOCK DEFINITION TABLE)+OFFSET.

S.E.U.CK. initially stores this 3200 bytes from $lAOO-$2680. The colour for each block is stored in a 128-byte table, so as any one block colour can be easily indexed with an offset of its actual block number. The colour data is stored from $1900-$1980 in S.E.U.C.K.

The other important part of the blocks is the actual character set, this being stored in S.E.U.C.K. under the Kernal ROM, from $F800-$FFFF. However, these absolute values are irrelevant as far as we are concerned, as a standard S.E.U.CK. background file is saved as a single file, containing block definitions, block colour, character set and level data.

HOW THE EXTENSION EDITOR HANDLES DATA

When the extension editor loads in a S.E.U.C.K. background file, it gives a relocated load address of $1000. As was stated, contained in this file is the data for the level map, we won't be using this, as our 256 screens are to be stored in our own format. As the background file is now located at $1000, the relevant parts of memory that we require are;

$2000-$2080 Block colours
$2100-$2D80 Block definitions
$2D80-$3580 Character set

(A copy of the character set is dumped to $FOOO for use by my program).

SCREEN DATA CREATED BY THE EXTENSION EDITOR

Each of the screens created by the editor is 32 bytes long, each byte referring to a particular block number running from left to right - top to bottom when expanded onto the screen. The screen data runs from $4000 to $6000.

USING THE DATA IN YOUR OWN PROGRAMS

The files that you will need to view the screens from within your own programs are:

1. Block colours
2. Block definitions
3. Character set
4. Screen data

Each file is saved from the editor; see the key function summary for actual key presses. I have supplied a source file, (VIEWSCREENS.ASM), to allow you to print any of the 256 screens. The file is well documented and should be self explanatory, if you read through it thoroughly. It is initially set up to run from address $COOO (49152 decimal); once assembled load in the following set of data that you have created within the editor.

BLOCK COLOURS (attributes) to $2000
BLOCK DEFINITION to $2100
CHARACTER SET to $3000
SCREEN DATA to $4000

The labels in the source file are set up to receive the data from the above addresses, note that all of the addresses are the default (can be loaded back with LOAD"filename",8,1), except the CHARACTER SET which must be relocated to $3000. These values can be easily changed in the source file to suit your own requirements. As is stated in the source file, to print a screen use:

```
LDA #screen number
JSR EXPASCR
```

This is all that is needed. By the way, how do you like my title screen? The music accompanying it was found embedded deep inside the COMPUNET front sequence from the DISKONOMISER program from the NOVEMBER '89 issue of CDU. That just about sums it all up. So come on all you graphics guys, and gals, out there. How about a bit of full screen animation by recycling the screens in quick succession or a mega-multi screen game? Good luck, and have fun.

ED'S NOTE :- LATER IN THE MAG YOU WILL FIND ANOTHER UTILITY THAT COMPLIMENTS THIS PROGRAM: S.E.U.C.K.WITH MUSIC. IT'S GOOD TO SEE PEOPLE MAKING THE MOST OF S.E.U.C.K.

The disk containing the original files is **"SEUCK-Extension.d64"**

SEUCK WITH MUSIC
By David Dewar

Adding music to your S.E.U.C.K. games is easier than you think
Published in Commodore Disk User volume 4 number 5, March 1991

Ever since the C64 was released upon us some years ago, people have dreamed of having power at their fingertips. Power to create various arcade games of their own, without the need to spend ages learning machine code. Numerous so-called GAME CREATORS have been released over the years to try and fulfil this creative need for non-programmers. Unfortunately, all these greatly restricted us to have simple, small and repetitive playing areas.

Thankfully, to the relief of most of us, the masterful duo that brought us WIZBALL and PARALLAX let loose their talents on a games writing utility known to us all as "THE SHOOT'EM UP CONSTRUCTION KIT" (S.E.U.C.K. for short). This program enabled thousands of C64 enthusiasts to create exciting arcade style games all of their own, with the minimum of effort. Impressive fast scrolling screens, complete with masses of whizzing enemies were now possible with the aid of S.E.U.C.K. Gameplay could easily be varied according to the users own needs and desires. Virtually every type of game could be made. Ranging from a sophisticated shoot 'em up through to a souped-up version of COMMANDO. The choice was only limited by the users own imagination.

S.E.U.C.K. AND MUSIC

S.E.U.C.K. might well be a great program to use and help show off your game designing skills, but one aspect that it sadly lacks is the option to include music with your finished production. Do not fear, all this can now be changed with the aid of a BACK-UP CARTRIDGE (with a built in RESET BUTTON), some MUSIC, a SMALL DRIVER routine and by reading on!

GETTING STARTED

From the CDU menu select the program "SEUCK WITH MUSIC". This is a demo giving you more useful information. After viewing this program load the program called 'SEUCK PLAYER'. On running this file you will be presented with a menu showing the available MUSIC which can be used. Press keys 1-6 to select a piece to load and listen to. Once you have decided which piece of music you wish to use, turn the computer OFF, then on again. Now load in your own or someone else's finished S.E.U.C.K. game. Reset the computer then load in the previously chosen piece of music, follow this by loading in the file name 'SEUCK-INIT'. Now you can enter SYS28681. A few important instructions will be displayed, all of which should be carefully carried out. Once copying is complete, reset and load in your new improved game and watch those raster bars bounce!

TAKE NOTE

On some rare occasions you might find that once you've pressed fire to start the game, you find that you're on another level. It does this because when you saved out the finished game from within the S.E.U.C.K. utility, you did not return the level parameter to the beginning. Pressing RESTORE will restart the game to level one. It is still best to go back to the S.E.U.C.K. utility, alter and resave. The chosen piece of music should not be loaded in from 'SEUCK PLAYER'. All music is played from $7506, so any other music you wish to use must occupy the same part of memory.

FINAL NOTE

I hope you enjoy S.E.U.C.K. games even more now that they have music with them. In the near future I hope to bring you a cruncher that will save out your S.E.U.C.K. games into the smallest file possible. Also, I hope to bring you an intro writer that will help smarten your S.E.U.C.K. creations by providing a nice little intro. Very special thanks must go to the MANIACS OF NOISE for supplying the brilliant music used in this program. Thank you Charles!

EXTRA HELP

The disk containing the original files is **"SEUCK with Music - Files.d64"**
There are 6 demo tunes included in the program, but if you want to create your own, here's some useful information.
1. Save the tune from the music editor to $7500
2. The initialise address should be $7500
3. The IRQ play address should be $7506
If this sounds too complicated, e-mail me for help.

QUICK TIPS

TIP #1 - INFINITE LIVES

Source: Commodore Disk User

Simply load a game that has been saved from SEUCK (using the 'Save finished game' option) and reset the computer.
Type POKE 17629,0: SYS 24576
You should be back in the Kit menus with all the game data intact.

Note: this will only work on games that have not been altered or had the Kit removed from memory.

TIP #2 - SEUCK WITH MUSIC

Source: Richard Bayliss
This is an alternative method to David Dewar's SEUCK WITH MUSIC

First of all, you will need to decide where to put the tunes in memory for the SEUCK game. I mainly put music between $6600 and $9FFF, providing that the code at $8000-$8100 does not get disturbed else enemies will shoot sideways, instead of random.

NOTE: A possible fix for this problem. POKE 21745,87 moves the random fire from a routine in the editor to a part of the main code.

So what to do then? Well, for a start, use a tool such as the All Round relocator, or the I-Relocator and load a tune which is at $1000 and see if the tune can be relocated. If the player can be recognized and relocated, then change the address to where you wish to put your music (must be between $6600 and $B67F). Once you have done this, save your tune on to a formatted disk. Let us have title relocated music at $6600 and in game music relocated at $9000.

Now load in your SEUCK game, and load up your 2 relocated tunes to their positions. Now we want to change some routines to initialize the music. Enter the Action Replay cartridge M/C monitor. Then enter the following assembly code

```
> 40E6 LDA #$00
> 40E8 JSR $6600
```

This will initialize the music in the title screen, but we will want the music to play, so now let's go to where the main title screen's loop jump addy is ($41BC JMP $4138) and change this to

```
> 41BC JMP $8100
```

What's $8100 going to be for? Well, this is going to be our own simple subroutine to play the music, then jump back to the title screen's main loop. So at $8100 type in the following assembly routine:

```
> 8100 JSR $6603
> 8103 JMP $4138
```

Now, test the game title screen using G 4245 in the m/c monitor or exit the monitor and type SYS 16965. The title screen will appear with some music playing in the background. Easy stuff huh?

If you feel that title screen music is not the same, and wish to have some in game music as well as title music, then this next part will show you how to play the in game music in your SEUCK game. At $440F enter the following assembly code:

```
> 440F LDA #$00
> 4411 JSR $9000
```

The in game music will initialize. Now we will want to play the music. So let us assemble at $4503 and enter the following:

```
> 4503 JSR $9003
```

The music will play. But if you enter the game, you will find that the music occasionally gets interrupted, because of the sound effects routines still being read. We will want to turn this off so do as follows:

```
> 5C0D BIT $D404
> 5C10 BIT $D40B
> 5C13 BIT $D412
> 5C24 BIT $D403
> 5C27 BIT $D40A
> 5C2A BIT $D411
> 5C2F BIT $D418
> 5C52 BIT $D405
> 5C55 BIT $D406
> 5C58 BIT $D404
```

Now save your final game with music from $0801-$FFFF, then execute the game (SYS 16965 or G 4245).

TIP #3 - LEVEL RESET

Source: David Dewar

Before saving your SEUCK game, make sure you reset the level counter. Go to Edit Levels, press F1 and make sure the cursor is on level 1 (at the left). Always do this before you save out the finished game.

TIP #4 - COMPRESSION

Source: Richard Bayliss

When you write a SEUCK game and want to make your game into a 1-file executable, you will find that your SEUCK game can be large in file size, even after using a compression tool like PuCrunch or Exomizer. There is a good solution to this problem though. All you need to do is zero some of the memory. So with a M/C monitor on your Action Replay, you would need to enter this command:

```
F 65B0 8000 00
then
F 8100 B67F 00
```

This cleans all memory of the SEUCK editor. After you have done this, use the Action Replay M/C monitor to save your game like:

```
S "MY GAME",8,0801,FFFF
```

Then use a cruncher on the file - start address $4245 / 16965 and set $01 to $37 / 55.

TIP #5 - SYNCHRONISED SHOOTING

Source: Jon Hare

To synchronise an enemy's firing with the first frame of its animation, go to Edit Objects for that enemy. Multiply the number of frames by the animation speed, and put that value into the Fire Rate in "Edit Enemy Bits" (press E). For a good example of this happening, check out the bad guys (Indian archers and sheriffs) in "Outlaw", one of the demo games that comes with the Kit.

Read more of Jon's wisdom in THE SENSIBLE GUIDE TO CONSTRUCTION.

JON WELLS TIPS

C64 SEUCK Unused Memory Map for Enhancements

When your SEUCK creation has been completed and you have saved a stand-alone version of your game it is possible to remove the editor and start adding in your enhancements.

Any enhancements you make will require some knowledge of Assembly Language and a suitable hacking ROM cartridge, I suggest using the Action Replay.

Enhancements can be done either the original way with an actual 64 or by using the CCS64 emulator on a PC with the Action Replay ROM bolted into it. I used this method when enhancing Synetic.

The following areas of memory are held by the editor and not used in the final stand-alone SEUCK'ed game:

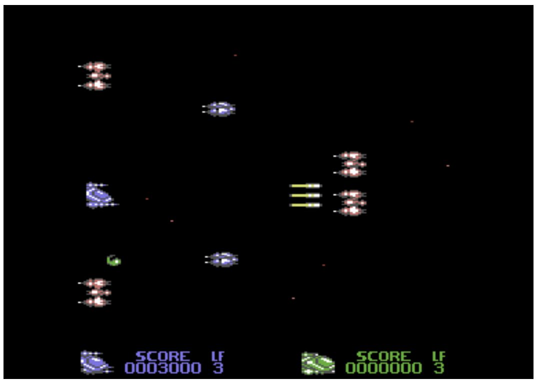
$0340	-	$0900	unused area
$3204	-	$340B	this area holds the games title screen text
$650C	-	$B6C0	main editor code and data
$E000	-	$E800	gfx and font for editor
$F280	-	$F500	?
$F5B0	-	$F800	?

Important
1) You must fill the $F3F8 - $F3FF area with $C1 's to ensure the borders remain blank.

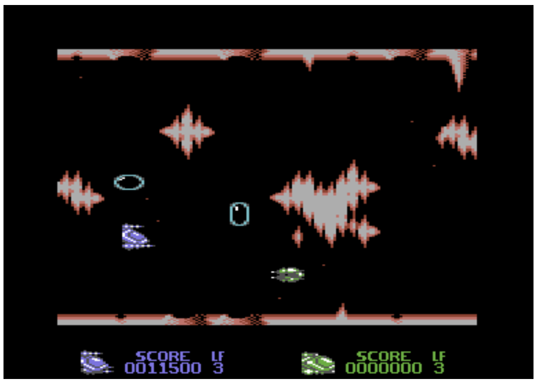
2) You must also change the command LDA $8000,y in location $54EF to LDA $5000,y as this points to memory and holds a basic random number sequence for the enemies fire.

BREAKTHROUGH

BREAKTHROUGH was to have been a unique achievement - a SEUCK game that scrolled horizontally!



Fast-moving enemies reminiscent of Armalyte



Dodge through the bubbles and background

With amazing music by Paul 'Feekzoid' Hannay, all that is left is the preview that was published on the Commodore Format covertape. You can also read more about it at in an interview with Jon at::
Games That Weren't. **http://gtw64.retro-net.de/**

REVIEWS & TESTS

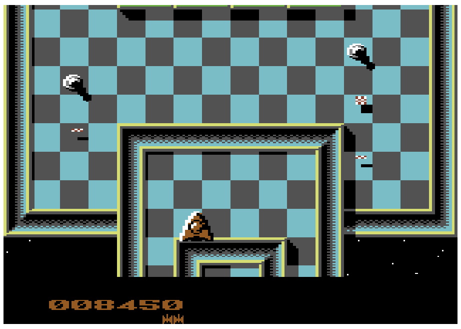
ZZAP! GOLD MEDAL REVIEW

SHOOT 'EM UP CONSTRUCTION KIT
Outlaw Productions, £14.95 cass, £19.95 disk, joystick or keys
Create your own sizzlers with Sensible Software's SEUCK!
Reviewed in ZZAP! issue 33, December 1987

Sensible Software, whose previous successes include Parallax and Wizball, are set to take the vertically scrolling shoot 'em up market by storm with their Shoot 'em up Construction Kit, a utility which allows a person with absolutely no knowledge of BASIC or machine code to create their own stationary or vertically 'scrolling one or two player head-to-head blasting games.

The SEUCK allows all the variables of a shoot 'em up to be defined to suit the user's requirements. Sprites may be designed and animated, backdrops drawn, alien attack patterns set up and even scoreline and bonuses fixed. A completed game may also be saved and played independently of the designer program.

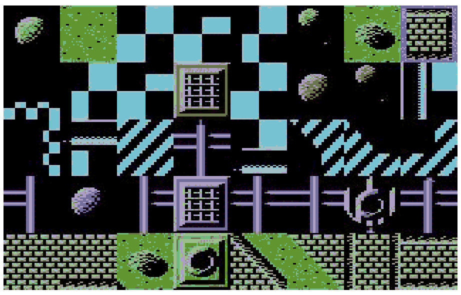
Designing a game starts with the main menu. From here the user selects one of nine editable features: sprites, backgrounds, objects, sound effects, player limitations, attack waves and levels to the character set and front end. Choosing one of these results in-a sub-menu appearing, which gives further options for the editor in operation.
The chess pieces move like real chess pieces!



The SEUCK even allows two players to do battle on screen at the same time

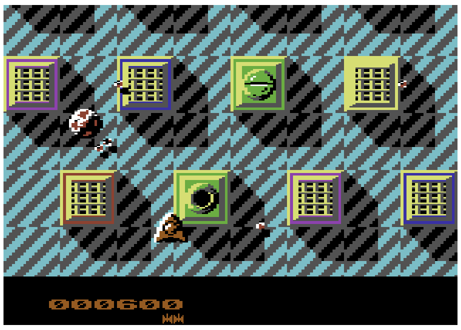
For example, when 'the 'Edit Sprite' option is chosen, the screen prompts: select, edit, edit colour, slide, mirror, copy and erase. This comprehensive series of options allow up to 128 12x21 pixel sprites to be designed, coloured and animated. These may be saved out, or used immediately in the 'Edit Objects' mode which allows the animation to be defined and polished as required.

Backgrounds are designed and drawn in a similar fashion, with an extensive sub-menu allowing a char(acter) to be selected, edited and coloured, and all the chars to be put into blocks and used to create a very large scrolling map.

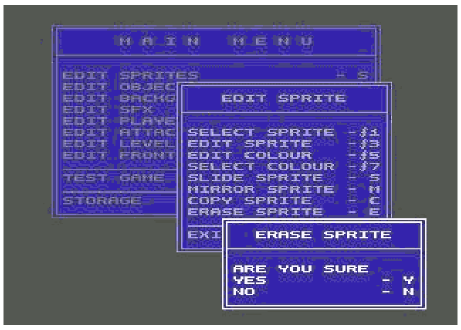

The 'select block' mode, displaying the separate blocks available to create your scrolling map

Once the backdrops and sprites have been prepared, the limitations of the player and enemy sprites are set. Here, the user defines the enemy attack patterns, the level of their aggression and sets the players' movement.

The game may be split up into separate levels, the duration and type of which can be specified, including a static bonus screen at the end of each, if required.



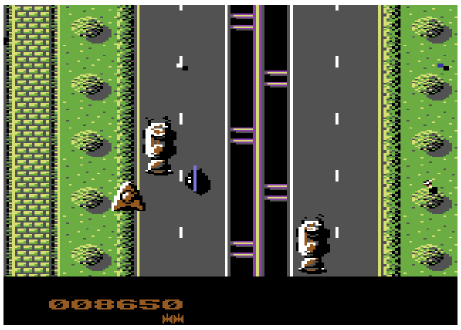
To add to the atmosphere, some suitable spot effects may be created using the 'Edit Sound Effects' option. This sound generator is akin to a miniature mixing deck and allows up to 24 different sound effects to be generated by adjusting the on-screen slide controls. The wave, attack, pitch, plus speed and time of the rise and fall can be altered to create a wide variety of sounds, from soft gongs to raucous white noise blasts.



Use a joystick or keys to move around quickly In space no-one can hear you shoot... no sound in a vacuum
Alien attack patterns are easily defined and stored in the program

For the finishing touch, a title screen is also written and 'colour cycled' by selecting one of the 23 predetermined patterns. The character set may also be redesigned to give the game that final polished appearance.

As the creation process progresses and new features are implemented, the game may be play-tested and tweaked as necessary. Finished games are then saved to tape or disk using the storage facility, and run as an independent program.
You've got a (slap) fight on your hands



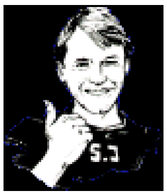
More blasting chuckles with Slap 'n' Tickle

To show some of its capabilities, and give the user something to practice on, the package comes with four demo games: the cheekily named Slap 'n' Tickle, a variant on an Imagine licence, Outlaw, a Gunsmoke clone, Transputer Man, a Robotron game and Celebrity Shoot 'Em Up, a blaster containing sprites designed by many famous names within the industry.

PAUL SUMNER:

This has to be one of the greatest packages ever released on the 64. The games that come with it would be worth the money alone, but this gives you the ability to make your own. You've got an excellent sprite designer and graphics utility with features, such as slide and mirror, that are hard to find on some dedicated art packages! There's also a brilliant sound effects mixer thrown in for good measure. All this is so compelling, I defy anyone not to become totally engrossed as soon as they start using it. Its beauty is that it enables virtually ANYONE to produce high quality bashes of virtually infinite variation. The incredible feature of the package though, is that just when you think you've exhausted its resources, you find some other little trick it can do - such as the independent file saving. With people able to create high quality stand-alone games quickly and easily, it shouldn't be too long before we see the first SEUCK Sizzler or, dare I say it, Gold Medal?

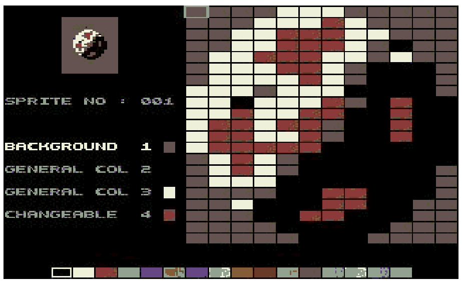
STEVE JARRETT:



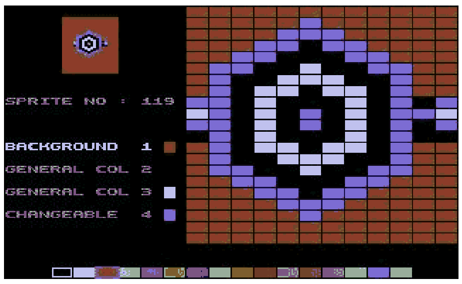
What a great idea! OK, it isn't the first, but it's certainly the best executed and most comprehensive (non-adventure) game designer to date. I couldn't believe how quickly and easily a simple game could be created. The whole package is brilliantly designed and so pleasant to use. The sprite designer alone is great fun and aspiring Disneys will appreciate the animation facility. I won't say that SEUCK will be everyone's cup of tea: some people have neither the time, imagination or inclination to sit down and design their own games. And don't kid yourself - it does take time to create a really good game - but if you fancy designing your own Sizzler, then SEUCK is just what you've been looking for.

JULIAN RIGNALL:

Sensible Software have certainly come up with something special this time! If you've got the time and patience, you can use the SEUCK to create some really incredible games. Earlier attempts at game designers have failed due to the lack of freedom. SEUCK gives the user an incredible amount of space in which to develop his ideas - it'll take a long time before someone manages to push it to its limits! The thought that has gone into its workings is absolutely phenomenal, so much so that you can sit down and start designing without reading the instructions - we did! You do have to refer to them eventually, and when you do they're helpful and informative. The package comes complete with a quartet of games, all of which are extremely good - in fact they're better than some full-priced games! The Shoot 'Em Up Construction Kit is a milestone in computing history. It mightn't appeal to all 64 owners, due to the amount of attention it requires, but if you've ever wished you could design your own game, this is a dream come true.
Edit sprites on the big grid... ...and see what they look like top left



SPRITE NO : 001
BACKGROUND 1
GENERAL COL 2
GENERAL COL 3
CHANGEABLE 4



SPRITE NO : 119
BACKGROUND 1
GENERAL COL 2
GENERAL COL 3
CHANGEABLE 4

The highlight of the construction kit: the superb sprite designer, editor, animator...

There's no doubt that The Shoot 'Em Up Construction Kit is an exceptional piece of software - the Gold Medal award is thoroughly deserved. However, since it's in a class of its own we feel that it's unfair to compare it with other software on the market, and therefore there are no ratings. We have offered three individual opinions as a guideline, but ultimately it's up to you to decide whether or not you'd like to try and design your own games.

Screenshots in this review have been re-created in the VICE emulator

ZZAP! RE-RELEASE REVIEW

SHOOT-'EM-UP CONSTRUCTION KIT
Gremlin, £4.99 cassette, £5.99 disk
Reviewed in ZZAP! issue 84, May 1992

10 PRINT, erm, what's next? Maybe, er, 20 GOTO erm, oh sod it - IAN OSBORNE just isn't cut out for programming, so we give him the Shoot-'Em-Up Construction Kit.

We all start out with good intentions, don't we? We all say we're not just going to use our C64 as a games machine; no, we'll learn to program it, write a couple of awesome games, make a fast fortune and retire to sunnier climes!

Trouble is it's too damned difficult - most people get as far as writing their names all the way down the screen, maybe scrolling it or making it flash on and off if they're really clever, then decide it's all too much hassle and go back to playing Space Invaders!

Well fret no more, 'cos Gremlin Graphics have re-released Sensible Software's Shoot-'Em-Up Construction Kit at a give-away bargain basement price, so now even the most inept would-be programmers can create exciting and original games... any of which can be sold directly to the public (or a major software house if it's that smeggin' good) without having to pay Gremlin a penny in royalties. Ha!

Cor, look at all the things you can do from the main menu:
you can even use a cheat mode to test your game!

Yah boo, SEUCKs to you!

Before you get down to business (fnarr) your first task is to decide just what kind of game you want to create. The SEUCK offers three different styles of play, all of them, sadly, vertically scrolling. Having a horizontal option would, presumably, have gobbled up too much memory.

So, your masterpiece is going to have to be either a straightforward (upward?) vertically scrolling blast (with a choice of fast or faster still but both supremely smooth), a single flip-screen affair or a 'push-scrolling' jobbie where the screen scrolls in accordance with the main sprite. Thankfully all three approaches can be combined in a single game, adding that all-important commodity we like to call VARIETY!

The program is utilised through a well-set-out menu system. From the main menu you can access any of the nine sub-menus controlling the editable features. Arguably the most interesting is the sprite editor, which allows you to create up to 128 sprites, 12 x 21 pixels large (plenty big enough to satisfy even Miss Whiplash's wanton needs!). Up to three colours can be used, but once chosen you are stuck with two of them for all your sprites so plan your colours carefully. You can also create a mirror image of a sprite, slide it about, (very useful if you go off the edge of the box), and even copy and alter it - animation has never been so framing easy!

Having created some spiffing sprites it's time give them some sort of a setting on which to do battle - that's where the background editor comes in. Each screen is made up of 254 characters, which are painted using exactly the same method as for the sprites. This is a fairly laborious task, but it is made easier by copy options.

SEUCK it to 'em

To define a sprite's role in the game, you use the object editor. Using its changeable colour option you can change the variable colour for each sprite, so if you want a series of similar sprites to make a simultaneous attack, they don't have to look the same. They don't have to move the same either - use the attack wave editor to send them zapping across the screen any which way you like!

So much for the baddies, what about the good guys? The player limitations menu allows you to adjust the number of lives, speed, blasting power, and everything else you need. You can even create a two-player simultaneous option! To make your game really stand out from the crowd, change the character font - now your control panel and in-game messages can look exactly how you want them to. And while you're at it, don't forget to add some appropriate sound effects using the couldn't-be-simpler sonic sub-screen!
The dragon - sprite 048And now at actual sizeThe Sound FX edit screen

With SEUCK, even the most talentless git can create great graphics and sizzling sound FX. Well done, Ian!

Every utility has its drawbacks, and SEUCK's is that it can't handle power-ups. I can't think why this has been left out. They're damned near ubiquitous in shoot-'em-upsville, and wouldn't have been hard to accommodate either - just create a routine whereby collecting an icon or blasting a specific baddie automatically adjusts the player limitation menu. If your game gives an extra life for scoring a certain number of points you can offer a huge score for killing a specific alien, effectively creating a 1UP, but you can never increase your sprite's abilities. Defining icons is a little awkward too - you have to create 'static' aliens that explode on contact, doing no damage whatsoever.

SAMPLE GAMES

Still not convinced? Well Gremlin have included a couple of sample games as part of the package. Created using SEUCK, they're a fine example of what the program can do in the hands of an imaginative programmer.

Slap & Tickle

Oo-er missus - with a title like this, I had trouble believing it was a shoot-'em-up! Not that I was disappointed - it was rather fabby! A vertically scrolling blaster, you control a neat little spacecraft that bearing more than a passing resemblance to the Star Trek logo! Boasting a huge variety of baddies, Slap & Tickle will tax your dodge-and-shoot skills to the full, and it gets harder as it goes! Silky smooth scrolling and excellent joystick response make Slap & Tickle a real winner.

A little bit of Slap & Tickle can be fun:this ready-made game is well worth playing when you're fed up of SEUCKing.

Blood Bullets

Unlike its forward-firing predecessor, Blood Bullets is a multi-directional blaster in the Commando/Ikari Warriors mould. Another polished performer, it's amazing just how good the backgrounds and animation can be. Unfortunately it's another vertical scroller. I can't help feeling that the second demo game should utilise another aspect of the creator, like push scrolling or flip-screen. Still, it's great fun to play, and a marvellous example of what the SEUCK can do.

Can anyone confirm that the GBH/Gremlin re-release on the C64 came with Blood Bullets? I was aware that it was one of the Amiga SEUCK demo games.

The best thing about these demos is that they can be altered in any way you like, using the kit itself - if you get stuck, just load up a demo game and see how the professionals do it! Like all SEUCK games, an automatic infinite lives cheat is included while still in the data base, invaluable when testing it.

So long, SEUCKers!

Despite these gripes, Shoot-'Em-Up Construction Kit is a brilliant program. Never before has there been a utility that allows you to create such fabby games - unlike previous games where they all looked and played the same. SEUCK games are limited only by your imagination. Pandering to all your needs, it's difficult to see how it could be any better - complicated animation routines are catered for, you can define multiple sprite baddies, and best of all when you save out your completed game it can be loaded and played independently of its parent program. At last, a game kit that lets you give copies to your friends, and don't forget to send one for the ZZAP! Megatape! We wouldn't be averse to paying for them either, if they manage to cut the mustard of course!

SEUCK is incredibly simple to use, but is not a magic wand - don't expect to create brilliant games within seconds of loading up. Even so, with a bit of forward planning and intelligent game design, there's no reason why your game shouldn't be good enough to publish. Give it a go - it's fun!

OVERALL STRIKE! 92% (SIZZLER)

COMMODORE DISK USER REVIEW

Shoot-'em-up Construction Kit
Review by Norman Doyle
Published in Commodore Disk User
volume 1 number 2, January/February 1988
The Commodore Disk User logo

How to build a first class zapper the easy way. The Wizball boys bring professional joystick action within your grasp.
Wow! I don't often say it but like, wow! This is crucial stuff, man.
When they said they were doing a construction kit, I thought Chris Yates and Jon Hare were treading a well-trodden path to disaster but what a blast they've come up with. What Chris and Jon have done is to break the shoot-'em-up down into eight separate parts, creating sensible routines and screen layouts to help the user design the game elements which can then be combined to make games that look incredibly different.

EDITING SPRITES

All the little characters in the game are sprites. If it moves, sprite it. Any of 127 sprites can be used, abused and generally mangled or manipulated. The sprite design screen is one of the best and fastest to use that I've ever seen. A really slick multicolour display that can be designed, copied and then flipped, wrap scrolled in four directions, mirrored or modified for animated sequences.

EDIT OBJECTS

This is where you breathe life into your creations. An object is any of the sprites that you've created and up to 58 objects can be chosen, each having up to 18 frames of animation. Obviously for all the sprites to have this number of animation frames you'd need over 1000 sprite definitions. A total of 18 frames is a bit luxurious and most times you won't need more than about six. Some of the objects have specific allocations such as Player 1 Sprite or Player 1 Bullet so care has to be taken to read the allocation at the top of the editing screen.

Even if you don't you can copy the information entered into the correct object page and this facility can also be used if two different objects have almost identical movement patterns. If the object moves it can have a direction-dependent animation instead of the continuously cycling one. That means if a spacecraft moves left, it will automatically select the sprite that tilts in that direction. Finally, the various parameters can be set for the score awarded for a kill, how many hits to make a kill, type of bullet, firing pattern, SFX (bullet and explosion), bullet speed, firing rate, and the result of a collision with the player. What more could you ask for?

EDIT BACKGROUND

This is the wallpaper that scrolls along as you advance through the game. 250 character blocks can be defined, each one consisting of a 5x5 matrix of character squares. Each character can be designed individually with three fixed colours and one that can be selected from the eight principal colours on the number keys of the computer. Once you've used the character editor - a smaller version of the sprite editor - you can build up each character block from these elements. Blocks and characters can be copied easily and the map can be built up block by block using the Edit Map option.

EDITING SOUNDS

There are 24 different sound effects built into the program but this is not as limiting as it sounds. Each effect can be fine-tuned to your own requirements. Waveform, attack, decay, pitch, speed and time can be varied until you hit on the sound you like. If this sound happens to be linked to the wrong category, you can copy across from one definition to another.

EDITING PLAYER LIMITATIONS

Why let the players have it all their own way? Edit Enemy Bits is the meanie panel where you choose the restrictions that will keep them cursing all the way through. The menus for Player One and Two are the same. Whether the player can take part, the number of lives and bullets, speed of player and bullet, range of bullet and direction can be determined. The other consideration is - what happens if an impassable block is reached? The option is between death or just stop. Really mean operators will be pleased to note that the movement of the player can be restricted to a fixed area of the screen, as large or small as you choose. The more generous designer may show magnanimity by giving away extra lives for each 10,000 points scored.

EDITING ATTACK WAVES

This is where the going gets really tough! You select the rough area of the map for each enemy object, fine tune to the exact point and press fire to place it at its start position. The pattern of movement can then be 'drawn' on the screen with the joystick and stored in the game's memory. There is a limited space for these moves so keeping it simple is the key to the hairiest shoot-'em-ups with lots of aliens. Mega-monsters such as you find at the end of each level of Nemesis can be created by joining several sprites together. These can then be animated to follow a flight path. Some of the possible effects are absolutely stunning.

EDITING LEVELS

There is a maximum of 22 levels that you can define. These can be any number of screens long and can be held still for a selected time while the player is mauled by a mega-monster, scrolled at a chosen speed or push-scrolled when the player hits the top of the screen.

TEST GAME

This is where the game plan really comes together and the temptation is to go off and blast a few objects...and why not? After all, the Test function has an infinite lives cheat mode so you can try the game from beginning to bitter end.

STORAGE

Saving can be directed to disk or tape despite the fact that you're using the disk version. There are two modes of saving. The first is saving the whole game in a form that can be loaded independently of the construction program - good enough to sell, in fact. The second save allows each design section to be saved individually for later fine-tuning after play testing, or simply to allow you to switch off in mid game design.

In later issues of Commodore Disk User we'll tell you how the sprites designed with the program can be manipulated for other projects.

NOT SO SILLY

The good news is that you can sell your games without having to pay royalties to Palace Software, the programmers or anyone else. This is not as silly as it may sound. All four of the free example games are worthy of consideration by the budget software houses and there's no reason why yours shouldn't be equally as good.

The bad news is that you can't have a loading screen or fantastic musical accompaniment. This may affect your chances with the budget people but surely your game IS just too good to ignore, isn't it?

The design of the screen can also take a little smoothing out. If too much is happening on the screen at the same time, the system can't cope and the movement slows down and judders. If you like you could argue that some of the best scenes of murder and mayhem at the cinema are performed in slow motion. I doubt if the potential buyers or your game would see it that way.

The three rules all programmers must remember are:

You can only scroll the screen vertically, horizontal is out.
Don't try to do too much on the screen at the same time.
Try to avoid having more than eight sprites in horizontal alignment on the screen or else the extra ones will become invisible enemies!

The Shoot-' em-up Construction Kit may be called SEUCK for short but suck it doesn't! It's a brilliantly constructed and executed utility. The whole package is self-explanatory and does not need a heavy tome full of instruction. All you need is printed on a neat, easy to consult poster that can be pinned on the wall behind or beside your computer.

Standard Slap Fight shoot-'em-ups are possible, but you can also produce Commando games, Gauntlet games and you may even break the mould and produce something that no one has thought of before.

As far as I'm concerned the only limitation to the Shoot-'em-up Construction Kit is the size of the gap between your ears. Are you big enough to take up the challenge?