# STOS Newsletter

Issue 8 • STOS User Club

Welcome to what is possibly the first ever STOS Club Newsletter to be released on schedule! (Gasps of shock and horror!) The flow of membership applications has steadied out now and I can get down to editing the newsletter a bit more. However we need more articles! I could probably write enough articles for each issue but that wouldn't be too great as this newsletter is meant to show YOUR ideas and skills as much as mine! So get writing...

Quite a few new products have happened since the last issue, so there is now a full list of all STOS-related titles in this issue plus a few others that are handy for STOS programmers, such as art-packages and so on. Also in this issue we start a series on programming STOS extensions, so it's going to be interesting seeing what new commands you can come up with. More Absolute Beginners this issue and a full review of Stephen Hill's new STOS book which you can order through the STOS club in our special exclusive offer. There's also information on the new STOS Musician program – a major improvement over the built-in STOS music routines!

## *STOS TOME*

Quite a lot of interest has been shown in the TOtal Map Editor I have been selling through the club. At the moment I am only just able to keep up with demand, so there may be a slight delay in getting your copy. A lot of people have asked me for more details of TOME and if it has been reviewed anywhere. Hopefully we will be running a review in the Newsletter, but I won't be writing it as that wouldn't be fair (I might be just a little bit biased!), but in the meantime New Atari User have reviewed the package in their March/April issue, and have given it a higher rating than STOS Maestro. Hopefully TOME Version 2.1 will be available soon in new packaging and better supply. There will of course be an upgrade for everyone who has got Version 2.0. In fact they will benefit as I am changing the demo game for TOME again so they will end up with two demo games!

## *Macintosh sounds!*

Chris Payne at Mandarin has bought a CD-ROM player for his Macintosh, and a 550Mb CD containing tons of public domain programs. The benefit to you STOS users is that Richard Vanner has ported loads of sampled sound files to the ST and edited out the header information using STOS Maestro – so you'll soon be seeing some more sampled sound compilations for use in your games and demos. Richard is also looking at porting Macintosh screens to the ST so he can grab the massed of high quality digitised pictures and animation files. For more information contact Chris or Richard at Mandarin.

## *Inside this issue:*

# ABSOLUTE BEGINNERS

## ON DISPLAY

*Number 2 in a series of 1,000,000*

The machine-code freaks get their wish in this issue: STOS extensions. So here's the next in the series of articles for those of you who aren't yet up to machine coding new instructions, and those of you who don't want to grow glasses and a long white coat (a common malady amongst chronic machine code sufferers).

This issue I will concentrate on displaying things on the screen, and the various ways of doing this. As with the previous article in this series, I am trying to write the article for those of you with only a little experience of Basic programming. If you find it too difficult or too easy (I'm trying hard not to patronise anybody) please tell me so I can adjust the difficulty level accordingly. All information gained will be put into the final development of the Icon STOS Basic which will be a teaching system to enable you to learn STOS programming in a very simple and friendly way.

Okay, for this article we will need some sprites to play around with, so load **any** bank of sprites you want. I am going to be demonstrating ways of displaying the sprites, so it doesn't matter really what sprite it is. Sprites banks are automatically loaded into bank 1 by STOS when you enter the following line:

```
Load "mysprite.MBK"
```

The "mysprite" bit would of course be replaced by whatever sprite bank name you want to load (for instance "ANIMALS1.MBK"). We will be working in low resolution so you'll need to enter: MODE 0 You might find it useful to clear the screen from time to time, so we'll use the CLW command (CLear Window).

```
CLW
```

To display the first sprite enter:

```
sprite 1,160,100,1
```

...and hopefully a sprite will appear somewhere in the middle of the screen. Try entering this line again, using different values for the 2nd & 3rd parameters. These are the X and Y co-ordinates of the sprite, so changing these values will move the sprite around the screen. The fourth parameter is the sprite's **frame** number, changing this will change the sprite **frame** that is displayed. The first parameter is the sprite number.

You have 15 sprites to play around with, numbered (for once logically) 1-15. All 15 can be displayed at once, and each can use any of the frames defined in the sprite bank. Okay, things are about to get slightly technical now, in an attempt to explain how the sprite system works. Hopefully this will make life easier later on!

The sprites work on an **interrupt** system. This means that every 1/50th of a second the Atari ST stops whatever it's doing and goes to the sprite control routine in STOS. This routine then redraws whatever sprites are visible on the screen by copying an area from the BACKground screen where the sprite used to be, to the LOGICal screen. This erases the sprite from the LOGICal screen. Then the routine re-draws the sprite on the LOGICal screen. So, if your sprite was covering a picture of a tree on your screen, the area behind the sprite will be re-drawn by copying it from the BACKground screen to the LOGICal screen, then the sprite will be re-drawn in its new position. This system is normally transparent to the user and causes little problem. However it can sometime cause hassles when doing scrolling – more on this problem later.

To remove a sprite, you can do three things:

1) Do an OFF command. Unfortunately this command also switches off all the other sprites and any music.

2) Change the sprite position to somewhere off the screen by typing something like:

```
sprite 1,999,1,1
```

or

```
sprite 1,999,1
```

Notice that the second version does not have a frame number. This is optional as the sprite command will redraw the sprite with the same frame as it had before.

3) Do the following set of commands, substituting your sprite number for the value n:

```
dreg(0)=0:dreg(1)=n:dreg(2)=0:trap 5
```

This is the fastest of the commands, and also has the result of making your friends think you're using machine code! The next display command we will use is the SCREEN COPY command, so remove the sprite from the screen using one of the above methods and type in the following program which will load a screen into memory so that we can play around with it.

```
10 mode 0 : key off : curs off : hide on
20 reserve as screen 5
30 load "\STOS\PIC.PI1",5
```

Line 10 resets the screen to low resolution (MODE 0), gets rid of the function key block at the top of the screen (key off), gets rid of the cursor (curs off) and hides the mouse (hide on). If you were impatient and have typed RUN already, you will have noticed that the text and mouse cursors have disappeared. Type CURS ON and SHOW ON to get them back – and stop being impatient!

Now enter the following lines:

```
40 fade 6 to 5
50 screen copy 5 to back
100 curs on:show on:end
```

...and run the program.

Line 40 Fades the colours of the current screen to that of the screen in memory. The first parameter is the time the fade will take (in 50ths of a second) and the second parameter is the screen bank number to fade to.

Line 50 copies the picture from bank 5 to the BACKground screen.

Line 100 restores the cursor and the mouse cursor and ends the program. You will notice now that if you move the mouse cursor, areas of the picture will appear on the screen. This is because of the way STOS handles sprites (as described above). Because the picture was copied to the background screen and not the logical screen, when the sprite (mouse pointer) is erased by copying the area of the background screen to the logical screen, it changes the logical screen. The way to stop this is to make sure both the screens are the same before you do any sprites. So add the line:

```
60 screen copy 5 to logic
```

...and run the program. That's solved the problem with the background. Now we'll try something clever. Enter the following lines:

```
50 screen copy 5 to back : screen copy 5 to logic
55 X=0
60 def scroll 1,0,0 to 320,200,-16,0
70 scroll 1
80 screen copy 5,X,0,X+16,200 to logic,304,0
90 X=X+16:X=X mod 320
95 goto 70
```

...and run the program. You will notice that the screen is scrolling to the left. I will describe what the new lines are doing and then we'll try and add a sprite.

Line 50 does the same as the previous lines 50 and 60: It copies the picture to both the background and logical screens.

Line 55 sets the variable X to 0.

Line 60 defines a scrolling zone. This one has been set up to scroll the whole screen 16 pixels left

every time a SCROLL 1 command is done.
Line 70 activates scroll number 1 once.
Line 80 copies an area of the picture to the right of the logical screen. By adding 16 to X each turn, the next 16 pixel width column of the picture is copied.

The X=X mod 320 on line 90 is to make sure that X doesn't go higher than 319. The MOD function (MODULUS) gives the remainder of A divided by B, so 15 mod 12 would equal 3. It is incredibly useful for making numbers go around in a loop, so when X gets to 320, X mod 320 equals zero, making the program go back to copying the left of the picture again.
Line 90 makes the program go back to line 70 after waiting for the next Vertical BLank (so the screen doesn't flicker). After about 30 seconds you will probably find this effect quite boring, so press Control-C to break into the program. Here's where life normally gets complicated – adding sprites to a scrolling screen. Add the line:

```
75 sprite 1,160,100,1
```

...and run the program. You will notice that the sprite is corrupting an area around itself. This is because of (again) the way STOS works its sprites. The scroll command only affects the logical screen, whereas sprites work on the logical screen AFTER copying the background to their old location, thus the old (unscrolled) background is being copied over the (scrolled) logical screen before the sprite is drawn. End result: Garbage around the sprite. A trail is also being drawn behind the sprite – this is caused by the sprite still being on the logical screen when the scroll command is done. We'll fix the garbage around the sprite first though. So how do we fix it? Simple: Apply a little logic to the situation. Over to you Mr Speck...

### *Speck's Laws of Programming #1*

Whenever you draw a sprite and the background and logical screens are set up as different screens (as they normally are), you must make sure that the background screen is the same as the logical screen, otherwise garbage will occur around the sprite. Or:
Background screen <=> Logical screen where <=> is the programming symbol for "had better damn equal"

So in the case of our 'scroll and do a sprite' routine we should copy the logical screen to the background screen with something like:

```
72 screen copy logic to back
```

Enter this and run the program. But the sprite is still leaving a trail behind it! This is because the sprite is on the logical screen when you copy it to the background screen. End result: Lots of garbage on the screen.

### *Speck's Law #2*

Always avoid copying from or scrolling anything on the logical screen unless ALL sprites and mouse pointers are removed from the screen. So the solution is to scroll the background screen, copy it to the logical screen, and then add the sprite. We could achieve this with:

```
70 logic=back : scroll 1
72 logic=physic: screen copy back to
logic
75 sprite 1,160,100,1
80 screen copy 5,x,0,x+16,200 to
Back,304,0
```

The logic=back command sets the pointer for the logical screen (normally set to the physical screen which is the one you see) to point to the background screen, so that it will be scrolled by the scroll command. The logic=physic on line 72 sets it back to normal for drawing the sprite. Line 80 has been changed to take into account the fact that we are now scrolling the background screen instead of the logical screen. However, as the sprite is on interrupt, it might draw itself when STOS is on line 70 and logic=back, with the result that it would be drawn onto the background screen, so we must make the sprite update only when we want it to. Try this:

```
69 Update off
75 sprite 1,160,100,1 : update
```

The Update off command switches off the automatic sprite updating so that the sprites (all of them, including the mouse pointer) will only be drawn when you do an UPDATE command. The Update On command switches these updates back on. So now you should have a program that scrolls the screen, with a sprite overlaid on top of it. You will notice the fact that the line numbers aren't exactly evenly spaced – this leads to:

*Speck's Law #3*

Making your program look like the internals of a telephone exchange is a good way of getting programming credibility as other programmers cannot tell what you are doing, and thus think you are an amazing programmer.

P.S REM stands for Rapidly Evaporating Memory! That's enough for this issue – I've got to go and feed the machine-code freaks now ! ■

## Fun School 3

Database/Mandarin need one or more programmers to work on the Over 7s version of Fun School 3 to be written in STOS. If you have time on your hands, ideas for programs, or know anybody interested in coverting to 8-bit micros like the Spectrum, please ring Chris Payne on

## Upgrade now!

If you are producing PRG versions of your games or demos, it is essential that you upgrade to the latest version of STOS otherwise they won't work on the newest Ataris. Version 2.5 works with TOS 1.6 and the Atari STE which is now on sale.

Simply send £2, or £1 plus a disc to Sandra Sharkey at the STOS Public Domain Library.

## *Exclusive Offer to STOS Club Members!*

The STOS Club has been provided with a disc with all the listings for Stephen Hill's book "The Game Maker's Manual" on a 3.5" disk. So that we can exclusively offer you the book and listings disk for the normal RRP of the book of £11.95. That way you get all the programs without wearing out your typing finger – and there are some hefty listings in this book! This offer is exclusive to STOS Club members only and is subject to availability. We expect to have enough for the first 50 or so orders, but demand for the book is high – so get your orders in now!

If you already have the book you can obtain the disc on its own by sending £2. Make cheques payable to Aaron Fothergill (STOS Club) for £11.95 at the usual STOS Club address. Your book and disc will be despatched within 24 hours by first class post in a protective jiffy bag.

# REVIEW

*The Game Maker's Manual Atari ST & STOS Basic Stephen Hill. Sigma Press 11.95 280 pages.*
*Review by Aaron Fothergill.*

This book has been at the "should be available next week" stage for a few months now, so I was rather pleased to get a review copy the other morning! I was also pleased that it's one of those books that you can either start on page one and read all the way through, or just dive in at certain sections to pull out useful information.

With technical books I tend to do the latter, and because each chapter seems to have been written to be self contained, this book definitely appeals to the part of every programmer and computer user that wants to bash something in without doing too much reading. There is a lot in this book – the list of contents alone takes up six pages! The lengthy contents section is handy though as it means you can quite easily find a section in a hurry when you are working on a program.

There are 11 chapters in the book, all of which deal with a different section of games programming: From the first steps in program design to programming STOS extensions in machine code. There are also chapters on writing shoot-em-ups, Rebound games, Simulations, Role Playing Games, Adventure Games, 3D techniques, Animation techniques, Sampled Sound and Scrolling Techniques – so it basically covers almost every type of game design going!

Being pleasant to read can make all the difference to a technical book as boring books lead to bored programmers. Although this book isn't the sort of book you would want to take on holiday with you to read on sun-drenched beaches, it isn't going to get put on that incredibly dusty shelf that you really must get around to tidying one day. The text is easy to read, has interesting points about **why** certain programming techniques are used and what other ways you might want to try programming a routine. There are also some interesting historical points for all you nostalgia buffs, including a complete mini Kingdom game!

Chapter 11 does a complete explanation of the STOS extension system, and for those of you who are well into machine code, this is a good enough reason in itself for getting this book. Both Interpreter and Compiler extensions are covered. I might also point out that I got all the information I needed for programming the TOME extension from this chapter, as Stephen kindly lent me a copy of the chapter to work from. The only bad point I can find about this book is really only a problem of its late release. Because it was written several months ago, no mention is made of some of the newer STOS products. It covers using Maestro quite well, and although it refers a lot to the fact that the Compiler is a very useful thing to have for speeding up your programs, there isn't much on using the compiler itself to get the best results. Newer extensions and programs like the STOS Squasher, the Double joystick routines and TOME aren't mentioned. Although to be fair, none of these three were around at the time of writing the book and I would be out of a job if this book explained everything!

## *The Long and the Short of it*

If you are serious or intend to get serious about programming in STOS, or if you want to try out some new programming techniques, then buy this book! My review copy is about to be chained to my computer desk so that they can't take it back! By the time you read this the book should be available either through your local bookshop, or computer shop. Take note also of our exclusive offer for Club members on page 7.

## Useful tip

Free with every copy of Games Galore is STOS Squasher, a powerful file compression routine. Simply insert the routine in your STOS folder, then when you boot up STOS you've got two new commands: SQUASH and UNSQUASH so you'll be able to compress your MBK and PRG files by up to 60%!

If you are stuck for space in your games, STOS Squasher could be just what you're looking for.

# The Complete STOS Programmer

A question I often get asked on the help line is "what other accessories are available for STOS, and what ones do I need?". Quite a few people have bought STOS with a specific project in mind. Most of you have heard already of the Maestro & Compiler accessories for STOS, and quite a few of you have heard of the Sprite 600 package. There are also some new products coming soon, and also some utilities written by other companies. In this article I shall be overviewing some of these accessories and explaining what you might need them for – hopefully without sounding too much like a salesperson!

Also included with the descriptions are the RRP (Recommended Retail Price) of the utility and a typical price that you might expect to pay from a mail order software company.

### *Generally Useful*

**STOS Compiler:** Almost vital if you want to release something commercially. It speeds up your program by an impressive amount, and also saves it as a .PRG file which can be auto-booted. This means that it is much more difficult for someone to work out how you wrote the program and steal your ideas. This should be first on your list if you're serious about your programming. RRP £19.95 Typical price £13.99

**STOS Maestro & Maestro Plus:** Sound sampling software and software with cartridge. Maestro can be used to enhance the sound effects of your games or provide backing music for them (although this takes a lot of memory). It can also be used for basic speech synthesis and the cartridge is usable as an audio input (for things like voice triggers). The software can be used with the Replay cartridges so if you have one of these you don't need the Maestro Plus cartridge. Coming soon will be Maestro 2 with improved sampling quality and some extra editing facilities. RRP £24.95 (Maestro) Typical price £16.99 (Maestro) £69.95(Maestro Plus) £54.99 (Maestro Plus)

**STOS Sprites 600:** 600 pre-drawn sprites. This package is quite handy if you are writing a shoot-em-up and don't have a graphic artist to draw your sprites for you. There are other sprites in this pack also, but they are really only much use if you use them to create the other sprites you need, as they are usually either static, or are only drawn for one direction. RRP £14.95 Typical price £11.95

**STOS Musician:** Originally intended to replace the un-friendly music editor provided with STOS, Musician got slightly out of hand and now includes a complete new music extension with an 8-track sequencer! Musician is easy to use and can have internal chip, sampled and MIDI music tracks in the same song. Also includes 100+ tunes in both the new and old music formats. Including music from the Games Galore package and original music by Shadrax. Well recommended for your game's background music, it is also a potential educational tool. (available April) RRP £14.95 to £19.95 Typical price N.A

**The Game Maker's Manual:** The first STOS-related book to be released (see the review in this issue). This could be called the STOS manual part two! Loads of useful information for all levels of STOS programmers. Highly recommended. RRP £11.95 Typical price N.A

### *Specialist use software*

**TOME (The TOtal Map Editor):** This is a STOS extension that gives you extra commands to handle map-based games. 95% of today's shoot-em-ups and 99.9% of the platform games are written using Tile-based maps for the backgrounds. TOME can also be used to create scrolling demos (see the Hotdog Demo in the P.D Library). Very useful for creating these sorts of games. Has been used to write Yomo on the Games Galore compilation and is being used by professional software authors to design commercial games, as well as being used for Yomo 2389, the "Plus" version of Yomo. RRP £18.95. Typical price £14.95 (to STOS Club Members).

**STOS Vidi:** A video digitizing package including a STOS editor. Very useful for grabbing your background screens from the TV or video. Also useful for putting pictures of yourself in your games (as well as creating alien faces with them!). Quite handy. There is also a software-only version for those people who already have the Vidi cartridge. RRP £99.95 Typical price N.A

**STOS 3D:** Extra STOS commands to create and use interrupt-driven 3D objects just as if they were normal STOS sprites. Should be good! (Available mid-summer) RRP N.A Typical price N.A.

Also worth mentioning in this list are some of the recommended art packages available, and some other useful programs.

**STOS Paint:** Useful as an emergency art package because it can be in memory at the same time as your program. Limited facilities, but worth the price! RRP £5 (STOS Club Shareware)

**Cyber Paint:** Very useful low resolution art/animation package. Great for designing sprites and animation sequences. Well recommended! Typical price £54.95 (Cyber Paint 2)

**Degas Elite:** Much cheaper art package, with lots of useful functions. Works in all three resolutions. Great for the price. Typical price £19.95

**Neochrome:** Basic pro art package for low resolution. Check out PD libraries for the early PD version.

**STOS Word:** Feature-packed word processor. Free with Issue 7 of the newsletter. Soon to be available commercially. RRP £14.95

**STOS Samples Discs:** These include some pretty good sounds for STOS Maestro and can be handy if you have Maestro but not the cartridge. RRP £2 each from the P.D Library

**PCP (Pre Converter Program):** Useful if you don't like line numbers. Can also be used to translate programs from different dialects of Basic when the language definer is released later. PD

**STOS Squasher:** The new data compaction routine provided with the Games Galore compilation. Can compact your programs by up to 60%. Essential if you are writing commercial software! RRP £19.95 (Games Galore) Typical Price £16.95

**Devpack 2 Assembler:** About the most comprehensive assembler for the ST. An earlier and cheaper version 1 is also still available. Typical price £40.

Well that just about covers everything you might want for your STOS programming. Full reviews of the new software will be in the Newsletter as soon as they are released. ■

## Advertising in the Newsletter

How would you like to advertise your products to nearly 500 readers? From the next issue I will be running payed advertising in the newsletter as an experiment. The cost of an advert will cover its own page, and another page of articles. There will also be a limit on the number of adverts in the newsletter as nobody wants too many adverts! Here are the costs for full page and half page adverts

| | |
|---|---|
| Full Page | £30 per issue |
| Double Page | £55 per issue |
| 1/2 page | £18 per issue |

Adverts must be supplied either as an ASCII file on a disc or as an A4 or A5 page suitable for black and white photocopying. Advertisements will only be accepted for genuine STOS-related products, and anyone trying to advertise pirated software will be grassed on to FAST. If you would like to advertise in the next issue of the newsletter, and would like more details, please contact me on the helpline. Also, if you are releasing a STOS-related product and would like it "plugged" and possibly reviewed in the Newsletter, send us a press release and review copy!

# Extending STOS

Okay, it's about time the secret of programming STOS extensions was revealed. This series of articles will be for serious programmers only, as it involves some heavy machine code, but given that you can use it to create your own STOS commands, it is well worth it!

This series will describe how to program STOS extensions for the interpreter and for the compiler. If you can't wait, and for technical reference anyway, I would recommend you buy Stephen Hill's book "The Game Maker's Manual" (see the review and the special offer) as the whole of extension programming is covered in detail in one chapter.

Due to the fact that we have limited space in the magazine, this subject will have to be covered in 2 to 3 articles. First of all I must point out that there is only space in STOS for 26 extensions (A-Z). Several of these have been used or reserved already, and Mandarin have asked that anyone writing further extensions should check with them to make sure that the letter and commands they want to use have not already been defined. Another convention recently implemented is that all major extensions should place an optional "sticker" on the STOS title picture on booting up STOS (By pasting it onto the PIC.PI1 file) so that it is easy to tell that the extension is present when booting up STOS (This is also useful extra advertising!). The extensions that have been used or reserved to date are:

| Letter | Use |
|---|---|
| *A* | *Compactor Extension provided with STOS* |
| *B* | *Not Allowed in V2.3 so not used (a bug with V2.3)* |
| *C* | *Compiler Extension* |
| *D* | *STOS Maestro Extension* |
| *E* | *STOS Squasher extension (free with Games Galore)* |
| *F* | *Double joysticks extension (free with Cartoon Capers)* |
| *G* | *Reserved* |
| *H* | *Reserved* |
| *M* | *STOS Musician extension* |
| *Q* | *Reserved* |
| *R* | *Reserved* |
| *T* | *Shadow Software TOME extension* |
| *V* | *Rombo VIDI extension* |

This leaves you with I,J,K,L,N,O,P,S,U, W,X,Y and Z to play around with and probably a few open eyes on seeing what's been reserved! If you use someone else's extension letter your programs will work as long as that extension isn't present. However it will cause serious compatibility problems if you want to sell your new commands! Mandarin will be keeping a list of the extensions used and the command names so that no confusion arises.

## *The heavy stuff*

From now on it's machine code (eek, aaagh and other comic effects!). This issue we'll concentrate on Interpreter extensions as these are the easier of the two. Expect all sorts of bugs in your programs as the STOS extension system tends to do all sorts of weird things!

Interpreter extensions are machine-code programs which have extra header information so that STOS can use them. They are saved with the file extender .EXn, where n is the extension letter A-Z. Each extension can (In theory anyway) have up to 64 commands and 64 functions, however you are limited in that you cannot use currently reserved command names (so you cannot have a command called DOOR, as it contains the reserved word OR). Similarly, once your extension is installed (by placing it in the STOS Folder and booting STOS), all its commands and functions become reserved words, so your programs can't use them as variables.

With extensions it is a good idea to work out beforehand exactly what commands and functions you want, what parameters they will use, and exactly what format they will be in. Then pick a suitable extension letter, check it isn't being used and it's in at the deep end!

*Warning! You are entering a machine-code zone!*

Once installed, your extension will be called when STOS boots up, so that it can initialise any variables it needs and set any flags it wants. STOS always calls the start of your program where it expects a BRA instruction to your initialisation routine, such as bra INIT. Then a byte with the value of 128 is required, so do: **dc.b 128**

Next comes the token list for your commands and functions. Remember – a command just does something, whereas a function also returns a value. Commands use the even numbered tokens from 128-254 and functions use the odd numbered ones from 129-255. Your first token should always be 128, apart from that, they can be in any sequence you want except that you cannot use tokens 160 or 184. The format is dc.b "command or function name",token number e.g **dc.b "whiz",128**

Rules for command and function names

1) They must be entered in LOWER case.
2) They can contain almost any character, including spaces
3) All the characters are significant (which can be useful). For instance the COMPTEST ON and COMPTEST OFF commands in the compiler are actually stored as two separate commands.
4) They cannot contain reserved words, such as currently defined command names or reserved variable names. The token list is then ended with a zero byte. So an example token list:

```
TOKENS:
    dc.b “whiz”,128
    ; first token must be 128
    dc.b “zip$”,129
    ; ODD so it's a function
    dc.b “splat”,132
    ; remember, tokens can be in any
    order
    dc.b 0
    ; end it with a zero byte
```

Unless your assembler automatically handles it, an EVEN instruction should be next as byte values have been used. Next up is the "Jump" table which is a bog standard address look-up table to tell STOS where your commands are in memory. Note that the jump addresses aren't offsets as STOS recalculates them when it loads up your extension. However if you missed out any tokens you will have to put dummy jumps in the sequence.

For instance, in the example above, you would have to put dummy jumps in where the jumps for tokens 130-131 would be. The first byte of the jump table is the number of commands, (stored as a WORD). The jumps are then stored in LONG-WORD format. e.g:

```
JUMP:
    dc.w 5
    dc.l WHIZ
    dc.l ZIP
    dc.l DUMMY
    dc.l DUMMY
    dc.l SPLAT
```

After the jump table comes the data for the message STOS will display for your extension when it starts up. As STOS is a multi-lingual language (that last phrase is not to be said after too many drinks) you will need two messages: One for your English users and one for your French users. This is the same with any error messages you do. If you only do the English ones you miss out on rather a lot of French users, and if you only do the French ones then you will (a) miss out on 20,000 English STOS Users who can't speak French and (b) probably be François Lionet!

I have to admit that the current version of TOME doesn't have any proper French error messages (everyone with TOME suddenly types in Français to find out what there is instead!), but future versions will as I have only recently found my French-English dictionary! (Mandarin have now offered to translate any of your extension error messages for free!). So your message would be something like this:

```
MESSAGE:
    dc.b "Whizzo extension",0
    ; English
    dc.b "extension Whizzo",0
    ; French
    dc.b 0
```

Note that the messages are ended with a zero byte, and that they can contain almost any character, so you can use the control codes to give you inverse text for instance (like in TOME), but watch out as they may have funny effects! Two variables will be required by your extension so you will have to reserve space for them next:

```
SYSTEM:
    dc.l 0
RETURN:
    dc.l 0
```

These variables are to store the location of STOS's SYSTEM routines (so that you can use them) and the RETURN address, so that your program can get back to STOS without any bombs (or cherries!). So that's the header out of the way. Now we move on to the initialisation routine.

As previously mentioned, this routine is the first thing executed by STOS when it is booted, and it must do two things:

1) Move the address of the END of your extension into register A0

2) Move the address of your cold start routine into register A1 then an RTS is placed after the routine. The usual INIT routine is:

```
INIT:
    lea EXIT,a0
    lea COLDST,a1
    rts
```

Notice that to get the address of the end of our program we are using the Label EXIT which would be placed after the program. The cold start routine is then called by STOS and must perform a few tricks for STOS and can then be used to initialise your extensions variables if they need it (things like memory sizing are usually done here). The routine must pass the addresses of your welcome message,your warm start routine, the token table and the jump table to STOS via the registers A0-A3. The warm start routine is used to reset variables in the extension when UNDO is pressed twice. Most extensions won't need to do anything here, so the warm start routine will simply be an RTS statement.

```
COLDST:
    move.l a0,SYSTEM
    lea MESSAGE,a0
    lea WARM,a1
    lea TOKENS,a2
    lea JUMP,a3
WARM:
    rts
```

That's all you need to do for the extension header. All that's needed now are the actual subroutines for the commands! Each command subroutine must pass the return address from the stack to the previously defined variable RETURN usually by the instruction: **move.l (a7)+,RETURN**

To return to STOS you must jump to this address by doing something like this:

```
move.l RETURN,a0
jmp (a0)
```

Okay, I think it's time we wrote an extension. Nothing flash, just a one command extension that uses the command HELLO, which will print up the message "This is my STOS extension!" at the current cursor position. Not exactly wonderful I know, but we'll be doing more in the next article.

```
* Completely useless HELLO extension
* Aaron Fothergill STOS Club 1989
* adds the command HELLO to STOS

* first jump to our initialisation routine
  bra    INIT
* set up the token list
  dc.b  128
TOKENS:          dc.b    "hello",128
  dc.b  0
  even
```

```
* set up the jump table
JUMP: dc.w 1  ; only one extra com-
  mand
  dc.l HELLO
MESSAGE:
  dc.b 10,"Hello extension",0
  dc.b 10,"extension Hello",0
  dc.b 0
  even
SYSTEM:   dc.l       0
RETURN:   dc.l       0
INIT:
  lea EXIT,a0
  lea COLDST,a1
  RTS
COLDST:
  move.l   a0,SYSTEM ; a0 contains
  the address of STOS' ; system table
  lea  MESSAGE,a0
  lea  WARM,a1
  lea  TOKENS,a2
  lea  JUMP,a3
WARM rts
* now for our extension command
HELLO:
  move.l   (a7)+,RETURN         ; save
the return address
  movem.l  a0-a6,-(a7)  ; it's good
manners to make
*  sure that everything goes back as it
   was, so we save all
*  the registers to the stack here.

  lea  HITHERE(pc),a0  ; point a0 at our
  message string
  move.w  #1,D7        ; do trap 3
  (PRINT STRING)
  trap  #3       ; function (See STOS
  manual p257

*  Here is where we restore the registers
  movem.l  (a7)+,a0-a6

*  And end the routine
  move.l   RETURN,a0; put the return
  address in a0
  jmp  (a0)     ; jump to it
* we need a string to be output for our
  command
HITHERE:
  dc.b "This is my STOS extension !",0
      dc.l  0
    EXIT   equ  *
```

Enter it into your assembler, assemble it as a PRG type file and name it as HELLO.EXZ either by assembling it as that name or renaming it from the desktop. Copy this file into the STOS folder on your STOS Language disc backup and boot up STOS. I'll be reviewing some assemblers in the near future with an aim to finding the best one to use with STOS. So far though, I can recommend Devpack 2 (although Devpack 1 will do quite nicely). Please don't try to write this extension with the STOS assembler, unless you are insured for complete nervous breakdowns!

Once STOS has booted up you should see your extension's startup message on the screen. You can then test it by typing HELLO (RETURN). You should get the message: "This is my STOS extension! "on the next line. If you use the command HELLO in a program, it should print the message out wherever the cursor is. Useless or what! More next issue – like how to pass parameters to extensions, and some useful routines you might need for your extension. ■

This STOS Newsletter was edited by Aaron Fothergill. All articles are by Aaron unless otherwise stated. Layout by Chris Payne. Printing by Sandra Sharkey (Contact Sandra if you could use her services).

## *Drop us a line!*

Whether you've got a problem – or just got something to say, we'd like to hear from you.

We're on the lookout for letters, articles or suggestions to make the Newsletter even better. You can use old-fashioned pen and ink, but we'd be delighted to receive Ascii or STOS Word files together with a printout.

Hope to hear from you soon!

*Aaron*

## TWO WISHES

Here's Wish One for Mandarin: Could I have more speed from STOS? ie Would it make it faster if the mouse could be turned off? Not just to "Hide" the mouse, but to tell the computer to stop checking the mouse port. Would this save processor time? If it does, could the same rule be applied to other features of STOS? Autoback uses this feature – how about extending it to the whole language system?

Wish Two: Is it possible to have a command that increases the number of characters you can have across the screen – like a dot matrix printer when you pass special print control characters to it? I've often required this function which would be very handy.

An idea: Seeing the amount of add-ons that are currently available and are going to be released, would it be possible (cheaper?) to change the manual type to a type that pages can be added and removed? If the size was kept as it is (I understand it's something to do with the amount of display room taken-up in the large stores) when more utilities/add-on's/up-grades are produced, a new section can be added to the manual with its new commands. This would enable the new Maestro instructions and commands to be put in the same manual under a different heading. It would also make the creation of the instructions and commands for new PD software such as STOS Word to be easily added to the main manual. Assuming the user has a printer, the document can be printed out from the disc which supplies the PD software with the correct section, page size, page numbers etc. This idea would also make it possible for the user to add pages to the manual for his/her own reference, pointing out any quirks or non-documented features. If it is not practical for Mandarin to produce the file with the new version of STOS, could they supply them through the club and we could buy direct from you.

I have noticed that if you try to use PUT SPRITE while the sprite is off the screen, you can get Bus or Memory Address Error, or a complete system crash and lock-up, forcing a complete reset and a loss of program.

Lee Groves, Swindon

*If you've already got the compiler then you'll probably swear at me, but it is usually the best way of getting STOS to speed up. Switching the mouse check off might work, but it wouldn't be a significant amount of extra speed. Infact the time used by the "check if it's off" routine might be more than that saved. Any comments François? It would be great to have smaller text and text that could be placed at graphics co-ordinates (including in OR & XOR modes).*

*If anyone wants to (or already has) written an extension to do this please get in touch! I think the idea of a ring-binder type manual is excellent. Most "Pro" programs use ring-binders as they are usually updated every couple of weeks as they remove the bugs (sound familiar anybody?). I'm going to look into it! The Bus Error with Put Sprite was fixed in Version 2.4 of STOS, so it might well be a good idea to upgrade to the latest version 2.5 (which also helps Wish One, as it runs faster!) or buy the compiler which has the V2.5 upgrade with it. – Aaron*

## DOUBLE JOYSTICKS

I have been using STOS since I bought my ST six months ago, however I have experienced several problems. I am presently writing a game which requires two joysticks, but STOS can only read one port. How do you read the other port?

How can I open a text window without a border appearing around it, and how do you use different fonts without opening a window? I now possess an STE and have discovered that STOS does not run on it – is there any way to convert it to run on the STE so that files will still run on both STs?

Paul Vincent, London

*For a two-joystick routine, it looks like you will have to get Cartoon Capers which will include the twin-joystick extension. Use border number 0 to open a text window without a border. There is no way (apart from changing the default character sets as shown in the manual) other than opening a window, of changing the fonts (some-*

# LETTERS

*thing to work on I think!).*

*To get STOS to run on the STE, you will need to upgrade to V2.5, which is compatible with all STs. You can get version 2.5 from the PD Library.*

## COMPILER QUESTIONS

I have a few questions about the STOS Compiler which I have now updated to V2.5. It is very flexible and will compile almost anything, but the one thing I thought could be improved was the size of the code generated for Gem programs. For example the program listed below was compiled to run under Gem:

```
10 rem No statements exist!
20 end
```

The program compiled into 51274 bytes which, considering it does nothing, seems to be rather large! After some tests I found that the Sprite, Window and Music library files appear in the program, even if they are not used. Why – and can they be excluded in any way?

I have recently upgraded to V2.5 of STOS and it has already cured one of the problems I had with V2.4. I had written an extended disc format routine which ran perfectly under the interpreter but failed with an "address error" when compiled. The program called TOS using the TRAP function. After I upgraded the STOS discs, I compiled the program and it worked first time! The moral of this story is: Don't delay – upgrade today!

I have also come across a few bugs and undocumented features. I have included a short list of some of them. The CENTRE command has an annoying flaw of not moving the cursor to the next line after printing, so you have to use a PRINT command between lines. From time to time, after using disc commands, a disc change is not recognised. For example using a DIR command brings up the directory of the previously inserted disc. For some reason the FREQUENCY command can only be used in direct mode. This is annoying because it is a very useful feature for games or graphic applications. You can simulate the function by clearing bit 1 of the register at location $FFFF820A for 60Hz and setting it for 50Hz (Don't do this if you are using your ST with a television – it might damage it permanently! This trick is for colour monitor users only! – Ed).

The mouse pointer can be changed easily with CHANGE MOUSE. However if you LOAD or NEW a file (or do any bank access, or use FILESELECT$ – Ed) it will change back to the default.
Alastair J Cameron, Midlothian

*Thanks Alastair! You will be glad to know that there is a way of cutting back on what is used in a compiled file (in theory !). You should be able to remove or rename files on your compiler work disc so that they are not used by the compiler. For example, rename SPRITES.LIB to SPRITE.LBI to avoid it being used (only if you aren't using sprites or the mouse in your program). Maybe the next version of the compiler will have software switches so that you can select or de-select which routines you need (including extensions).* ■

**STOS**

## Hotline

Don't forget that membership to the STOS Club entitles you to ring Aaron Fothergill for assistance on any STOS-related subject.

Ring him on:

or write to:

# Public Domain Library

## How it works

Listed on this page you'll find the very latest additions to the STOS Public Domain library. For a list of the titles please refer to Issue 7. Each single-sided disc (SPD prefix) costs just £2 each – or £1 if you supply your own disc, and each double-sided disc (DPD prefix) costs £2.50. If you order three discs or more, deduct 20p per disc (including the first three). For example four single-sided discs will cost £1.80 each – £7.20 in total. (Please add 25p postage per disc if you live in Europe – or 50p per disc if you live elsewhere in the world, otherwise your discs will be sent by surface mail!)

A few discs are shareware: £2-3 goes to the programmer (deduct £1 if you supply your own formatted disc). Each disc has a specially-printed STOS Public Domain purple sticker kindly produced free-of-charge by Mandarin Software so your PD discs will match the rest of your STOS master discs.

In most cases the programs require you to boot your copy of STOS first – that way you can have a good look at the listings.

Ring Sandra on to find out about the latest public domain titles to be added to the library.

If you have any programs which may be suitable please send them along.

Send cheques, postal orders or stamps to:

*Sandra Sharkey,*

### Choose from the following:

**SPD35:** *Patience* – Pretty graphics with sampled sound

**SPD36:** *French Kiss* – Excellent samples with Vidi-grabbed images. Control the speed of samples with the joystick.

**SPD37:** *Slider* – Watch as the program jumbles up the stunning pictures, then it's your turn to rebuild the picture.

**SPD38:** *Tutorial* – On-line help accessory to all STOS commands. Stores the info in a random-access file so it doesn't take up much memory.

**SPD39:** *The Beatbox* – A small graphics and sound demo.

**SPD40:** *Dattrax disc* – Packed full of small STOS games, demos and utilities with lots of good ideas.

**SPD41:** *Jack and the Beanstalk* – Playable demo of a cute platform-type game.

**SPD42:** *Pusher* – Another blockslide game similar to Slider.

**SPD43:** *Wild River Run* – Jump from turtle to turtle before they dive in search of fish (based on a classic LCD handheld game).

**SPD44:** *Stoned Again (hic)* – Surreal demo based on the ubiquitous Robert Crumb comic poster.

**SPD45:** *Outside demo* – Six levels of a super Orbit-based game with topnotch music, features and effects.

**SPD46:** *Quiz list* – Compile your own quiz.

**SPD47:** *Winner* – Train horses then watch 'em go.

**SPD48:** *Clever Clogs and Times Table* – Play a version of Mastermind, or swot up on your maths.

**SPD49:** *Atari ST User Feb '90* – Contains Crisslefridge by Andy Cook, a 60-level arcade startegy game. Also includes Speaktext routine so you can write talking programs (though the speech is distorted by STOS's interrupts).

**SPD50:** *Atari ST User March '90* – Contains vertical shoot-'em-up Deathstar which was origginally considered for the Games Galore compilation, and Bomber listing (based on mag tutorial).

**DPD17:** *Amadeus* – Two-disc music demo from Darren Ithell which plays a specially mixed version of the hit single over scrolling messages. Requires two drives.

**SH5:** *Poker dice* – A superb dice game for one to four players with very innovative graphics and sampled sound. [Shareware: £4]

**SH6:** *Dragon Lord* – Save the princess in this Ninja-style combat game with sampled speech and effects. [Shareware: £4]

**SH7** *Perils of Penfold* – An arcade adventure from Budgie UK. [Shareware: £2.95]

N.B: The Atari ST User programs are not public domain but are available exclusively through the STOS Club

# François' shoot-him-up

After the fake virus in the last issue I have another present for you – and it is the best one: A picture of me! I have digitised my face (who said with a handy scanner??), then reduced the size to a small sprite, and here is the listing of this sprite. Now if something goes wrong with STOS, or if you find some weird thing, or if it crashes, losing four hours of work, you can have your revenge on me! Make a small game in which you shoot things at my face and un-stress you this way! You can throw pies, strawberries, bullets – anything you want! Please send your best "Shoot-Him-Up" programs to the newsletter and they will be published (Note to Chris: Why not make a small contest? Next newsletter, a sprite of Richard and then you!) To get this magnificent sprite:

1- Load INPDATA.ACB
2- Type all data (courage, it's worth it!)
3- Save bank with the name you want.

I have made a small demo program using this bank: BIG BROTHER IS WATCHING YOU! Just like the 1984 book! HeHeHe!

```
10 rem ————————————————
11 rem The STOS user's nightmare!
12 rem
13 rem By guess who? F Lionet
14 rem
15 rem I hear, I see, I know whatever you program in STOS!
16 rem Hehehehehehehe
17 rem ————————————————
50 break off : key off : mode 0 : hide on : curs off
55 A=hunt(start(1) to start(1)+256,"PALT")
60 for X=0 to 15 : colour X,deek(A+4+X*2) : next
65 back=logic
100 for N=0 to 10000
115 sprite 1,rnd(370)-25,rnd(250)-25,1
120 next
```

When you read my texts, you may think that I speak good English. It is not true! Chris always re-writes what I say. That's why I will ask him to stop re-writing what I say from this very word pour que vous ne soyiez plus trompés sur la marchandise. Hé oui, c'est la version originale. No I'm kidding, I write in English, but a very Frenchy one as you will see in what follows.

January is time to have a look to what has happened during last year. 1989 was definitely a very good year for STOS. After its launch in autumn 1988, everybody said that STOS would crash. And well, it did not – sales went on! In March the compiler was released and we have all been very surprised by its success: Mandarin sells one compiler for three STOS, and believe me, it is a lot! It means two things: First, at least one STOS buyer out of three uses it and makes programs good enough to be compiled! Second, there are not so many pirates as we thought. We have really hesitated to protect the compiler. Such a product can be very easily used without documentation. But we thought that protecting it would be an offense to all normal users. Anyway, the protection would have been cracked in a matter of days after its release. So we trusted people, and we were right. Thank you very much!

England is one year ahead of the rest of Europe: STOS has only just been released now in France and Germany. It was a little frustrating for me to have my product sold in England and not in France: I could not really see what it was like. Now I see, and I'm very happy. STOS is having a very good launch in France too. In a few years there will be zillions of STOS users all around the world producing good quality games hand in hand (with their feet) on their beloved machine. Peace, love and STOS, that's what we want... sorry, I was dreaming!

I can promise you that 1990 will be as hot for STOS as 1989. Lots of things are to come, like a video digitiser and the long-awaited 3D extension. Then STOS Plus with new things in it like AMAL (the interrupt-driven animation language I have put in AMOS), better sprites routines, full use of STE features (I'll buy an STE when they have removed all the bugs, and you please wait too), etc... I can't wait to see it, I bet you can't too!

What bothers me a little (and Mandarin too) is that I must do my Army this year. You lucky English, you don't have that! But believe me, French army is an enormous bureaucracy (compared to it a communist country is a land of free enterprise!) so I will not work a lot for them, and a lot for STOS (on a Stacy I hope!)

Well that's all for now. Thank you very much for reading me up to the end. I just have now to wish you all the STOS for the new year, and may all your games come true!

*Bonne Année, François.*

## STOS Vidi Digitiser

*Mandarin and Rombo have completed the STOS extension for STOS Vidi and are looking for someone to write a suitable editor which has already been specced up. If you have the time and skills, ring Richard Vanner or Chris Payne on during office hours.*

## SPRITE BANK DATA

```
—> Bank number: 1      Length: 1024

| Adrs | Data                                        | Check |
| 0000 | 1986 1987 0000 0012 0000 0336 0000 0336 | 0398B |
| 0010 | 0001 *0006 002C 0226 0D1B 5041 4C54 0035 | 1AC3E |
| 0024 | 0000 0011 0022 0032 0042 0052 0062 0072 | 001CD |
| 0034 | 0073 0173 0273 0373 0473 0573 0673 FFF0 | 11815 |
| 0044 | 0FFF FF00 003F FC00 003F F000 000F E000 | 3DB8C |
| 0054 | 0007 C000 0003 8000 0001 8000 0001 8000 | 2400C |
| 0064 | 0000 8000 0000 8000 0000 8000 *002C 0001 | 2802D |
| 009E | 0000 0001 0000 0001 0000 0003 8000 0007 | 0800C |
| 00AE | 8000 0007 8000 000F 8000 003F 8000 003F | 20094 |
| 00BE | C000 007F E000 007F E000 00FF F000 01FF | 373FC |
| 00CE | F800 03FF FC00 07FF FF00 1FFF 0008 0000 | 31F05 |
| 00DE | 000F 0000 1000 0000 F000 0000 000A 0002 | 1001B |
| 00EE | 00FD 0000 2640 2040 DF80 0000 0028 0000 | 12725 |
| 00FE | 03FF 0000 2C00 2C00 D3C0 0000 0F33 0033 | 13F25 |
| 010E | 0FCC 0000 E1F0 E1F0 1E00 0000 0D23 0003 | 1FED2 |
| 011E | 1FFC 0000 5428 5478 AB80 0000 03E3 0000 | 177FF |
| 012E | 3FFF 0000 E510 0418 FBE4 0000 20B0 01D0 | 2478B |
| 013E | 5C6F 0380 B99C 381C C7E2 0000 405B 06BB | 2609F |
| 014E | 3D24 03C0 F08A F10E 0FF0 0000 07CB 0C4C | 24683 |
| 015E | 73CF 0FB0 6797 7257 8E68 0180 72A8 7831 | 2D82E |
| 016E | 17C1 0FFE 8E48 01A8 F037 0FC0 7089 6831 | 29060 |
| 017E | 07C1 1FFE 10B8 ED68 F037 0FC0 55B8 4DC0 | 2C94E |
| 018E | 2200 1FFF B9AB B84F B800 47F0 D578 CD80 | 456E1 |
| 019E | 2200 1FFF 0D50 0C87 0C08 F3F0 5640 CF80 | 2818E |
| 01AE | 2000 1FFF 3E09 3E07 3E08 C1F0 781F C7E0 | 2FC06 |
| 01BE | 2000 1FFF 2E38 0E37 0E38 F1C0 D80F 67F0 | 2BC65 |
| 01CE | 2000 1FFF F820 0037 0038 FFC0 5644 C800 | 35692 |
| 01DE | 0000 3FFF ECEF 10F8 00F0 FF00 79D5 8E35 | 345E0 |
| 01EE | 4FF5 300A 00E8 00FF 00F0 FF00 CE66 D06F | 320AB |
| 01FE | 5F9F 2000 B379 317E CF70 0080 9BA9 8DBE | 35DED |
| 020E | 8E5F 7000 B3F9 4C7E FFF0 0000 9DBB 9F89 | 43C0A |
| 021E | BE7B 6004 8773 8DFE F9F0 0600 FA41 83A4 | 4B1C5 |
| 022E | 83E1 7C1E FC6A 19DF E7F7 0000 DD56 A018 | 47BAD |
| 023E | 8011 7FEE 09EA E7EE FFEE 0010 E811 03C8 | 3DDAE |
| 024E | 0011 FFEE 0042 FE4E FE4E 01B0 34EF 0CDD | 34059 |
| 025E | 80C1 FF3E EE9E 6E9A EE9E 1160 BD29 019F | 49BFD |
| 026E | 0181 FE7E 1C78 F17C F17C 0E80 1D9F 1E01 | 3498F |
| 027E | 1F81 607E 6C68 1E78 7E78 8180 611B 7ECE | 2EAC0 |
| 028E | 7FFF 0000 71E8 0FF8 FFF8 0000 7F9A 6DA4 | 2EF15 |
| 029E | 7FBF 0040 73F0 0FF0 FFF0 0000 402E 3CF0 | 280ED |
| 02AE | 7FFF 0000 20C0 07C0 FFC0 0000 088F 3C70 | 1ED3E |
| 02BE | 7FFF 0000 F480 03C0 FFC0 0000 0162 1FF8 | 29959 |
| 02CE | 3FFF 0000 7A00 4580 BF80 0000 03BE 1DC1 | 1E07E |
| 02DE | 1FFF 0000 1000 EF80 FF80 0000 19BC 07C3 | 2407E |
| 02EE | 1FFF 0000 1000 EF00 FF00 0000 0A30 063F | 22E6E |
| 02FE | 0E3F 01C0 7200 8C00 FE00 0000 01F8 07FF | 215F6 |
| 030E | 07FF 0000 1400 E800 FC00 0000 0197 0068 | 201FE |
| 031E | 03FF 0000 F800 0000 F800 0000 000E 0000 | 1F40D |
| 032E | 00FF 0000 2000 0000 E000 *00C8 0000 0000 | 201C7 |
```

# THE ART OF PSEUDOPHYSICS IN COMPUTER GAMES

*By Prof A Speck O.D.D*

So you want to do a simulation game? Tons of vector maths and physics later you have your game. Unfortunately the gameplay is so terrible you have to offer a free T-shirt with it to sell it (a bit of Amiga humour there!). So what went wrong? Simple! Maths and physics relates to real life, okay? So how many computer space games involve you sitting in a tin can with vacuum all around you and hordes of slimy things (real ones) attacking you? Answer: None – because the computer can't display things too realistically when it's only outputs are sound (however limited) and a TV screen. What's needed is a variation to normal maths and physics to take into account that your computer world is ever so slightly strange...

Pseudophysics is the application of **rule of thumb** in physics, to match the laws of physics to a computer environment (For a more accurate explanation of rule of thumb and conversion tables to metric and imperial measurements, read "Random Walks in Science"). This means that in your game you must discard any laws of physics that don't look or feel right. However beware: There are rumours that the government is bringing in fines for breaking laws of physics. For instance, having a game where people can fly, thus breaking the law of gravity, will carry a maximum fine of £2,000. You have been warned! (This doesn't apply yet to overseas readers, however other governments are looking at this system).

The first thing you must learn about pseudophysics is that nothing is fixed (or rather everything is!). Thus leading to Speck's 1st Law of Pseudophysics: **The fiddle factor.** All constants are variable until the game looks and feels right. This means that you should play around with any of the multipliers in your equations until the game plays the way you want it to and looks correct. Don't worry if your physics teacher would scream at seeing them (my physics teacher used to scream at my equations anyway). Equations should also be tested out by trying the game on someone else, preferably someone familiar with the style of simulation you are trying to do. Pseudophysics doesn't discard everything from physics however, this would be silly. It merely uses the laws that are useful at the time (sort of like record companies).

Sometimes you might have to create a new law to apply to a situation never used before, such as the time taken for a spaceship to get from A to B if it uses hyper-drive (Einstein is now spinning in his grave at three revs per minute now!). This law might take into account the factor of C being in the way, and the drag factor of a spaceship with planet C wrapped around it.

Sometimes you have to do a series of extra equations to make up for something that wouldn't otherwise look right. For instance, in the writing of Skystrike & Skystrike Plus, the aircraft is supposed to stall if the airspeed gets too low. To achieve this, a variable ST was used. This variable was subtracted from the altitude of the aircraft each frame. When the aircraft was flying normally, ST was set to 0. When the airspeed got too low, ST was set to 10, thus making the aircraft fall out of the sky rapidly. As the airspeed picked up again, the variable was lessened, until the aircraft pulled out of the stall. Various other routines were then able to use the ST variable to determine if the aircraft was stalling, and the speed at which it was dropping (the landing checker for instance).

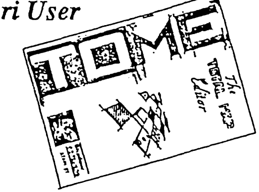
If you have Skystrike check out lines 50-69

---

*(Professor Speck is a lecturer in applied Pseudo-Physics at the University of Advanced Gameswriting in Barnstaple. He has also worked as an assistant test-pilot in games such as Skystrike, Skystrike Plus and Yomo. He is currently playing an end-of-level guardian in Yomo 2389.)*

which contain the sprite updater for the aircraft and the basic pseudophysics routines for its flight pattern. These were developed and test piloted over two weeks by a team of around eight part-time test pilots until they were as good as possible. So remember – real physics just don't work in the computer games world (just look at FOFT for an example!). So start re-writing those equations. You might want to use the general pseudophysics symbol of ?= which stands for 'sort of equals' and is very important in pseudophysical equations. For example V ?= D/T (Velocity equals Distance over Time multiplied by a Fiddle Factor) or E ?=MC^2 (Or: I'm using hyper space – eat ion exhaust Einstein!) By the way, this also has the use of making your equations less useful to anyone who wants to steal your game idea! Have fun... ■

*Next Issue: How to program in spaghetti code.*



# NEWS

## Look out for...

New Atari User have been doing a lot of reviews of STOS programs recently, In the November/ December issue Pete Hickman reviewed Skystrike Plus, giving it a definite thumbs up, and giving us this quote: "Skystrike Plus is one of the most playable games yet to appear on a 16-bit micro, ranking alongside Choplifter and Typhoon Thompson." The March/April issue also contains reviews of TOME and Games Galore, with TOME getting 82% for Value For Money and Games Galore was a recommended buy and noted as being "...fun to play.. and ...having that certain something that 99% of 16 bit games are missing."

## STOS Musician

By the time you read this I will have nearly finished my work on STOS Musician and the packaging will be getting ready for its release in April. The Musician project has gone through all sorts of problems, including a complete re-write and a change of programmers when one of the team had to drop out of the project. I was brought into the project in mid-November after the original design for Musician didn't meet its release date. It was decided that Musician should be completely re-written for a March rèlease with a completely new music extension (as the original Music routines supplied with STOS were a little lacking!). The new specification for the extension routines was sent to Bobby Earl, author of the STOS Squasher extension, while I got down to programming the new editor. The new music routines will be installed as an extension (alongside the old ones, although only one can be in action at any time) and will give you several new commands for music. The actual output facilities of the music routines have been changed to give you five more tracks. Musician's tracks are like this:

Tracks 1-3: Normal Soundchip tracks
Track 4: Maestro Sample track (Also compatible with Maestro II)
Tracks 5-8: MIDI tracks

All the tracks are stored in the same format to make it easy to copy data from one track to another. The data format is also a lot more simple and easier to understand than the old routines – it is also a lot more compact. The editor allows you to use conventional score editing (dots and squiggles), as well as grid-style step editing (like in professional sequencers). It also has sound editing facilities for the internal soundchip and a really easy to use song editor. Music is recorded into patterns one bar long which are then arranged into a song. Patterns can, of course, be used again any number of times, and only use one byte for each repeat. There are also loop and tempo change functions,so that you can change the tempo within the song. MIDI has also been implemented in STOS Musician so you can record patterns from a real keyboard. You can also play music into a pattern with the ST's keyboard. Along with all sorts of tricks like transposition, quantisation and the Fit Key function (which allows you to automatically fit notes into the correct musical key), Musician is going to be the ultimate music writing tool for STOS Music writers. Also included will be a converter program so that you can convert your old tunes to the new format, and a jukebox program to play all the many tunes provided with the package! With the release of the new Atari STE, work is already underway on an upgrade for Musician which will enable you to map the four MIDI channels to the STE's new stereo soundchip. Also to be included in this upgrade are a MIDI file format converter and a sample editor. More details in the next issue!

## Educational titles

STOS has become the language to use when it comes to writing educational software. Not only is there Fun School 2, but there's also Learn to Read with Prof from Prisma Software, Primary Maths Course and others from LCL, Let's Spell at the Shops and Let's Spell in French from Childrens Learning Software, and more to come. Prisma are already working with a development copy of AMOS so they can release Amiga versions.