STOS Newsletter

Issue 12A STOS User Club

What, me, superstitious ?!

At last Year 3 of the STOS Newsletter ! Appologies for the delay in publication of this issue, but a number of factors were involved in the delay, such as my developing RSI in my fingers, which meant that Adam had to do all the typing in of programs and articles for two weeks, and the shift to a new duplication and distribution network, as Sandra Sharkey will no longer be running the STOS P.D Library and doing the duplication and distribution of the newsletter due to the massive expansion of the AMOS P.D library.

Several new factors have appeared in the STOS equation recently, such as Mandarin Software's reluctance to properly support STOS with new releases, as they no longer class the ST market as worth writing for. Needless to say, there are a few of us trying to persuade them to do otherwise, or to supply us with the sourcecode to STOS, so that some of the machine coders who have been very supportive members of the STOS Club, can develop a new version of STOS. Even Francois thinks it can be done better.

Thanks to Francois' idea of making STOS extendable, we can keep STOS up to date with commands for the latest ST goodies such as the new STE's sound chip and Blitter chip. but STOS needs a new editor system, and several of the routines can be written to go faster.

On the plus side, Mandarin have agreed that anyone sending a STOS (or AMOS) written program in to be published, no longer need to mention that it was written in the language, they just need to write to Mandarin (or Europress, which is their new name) telling them when it will be released, so that they can say that it was STOS written three months after the program is released. This stops some of the more uninformed software companies from ditching STOS written programs AFTER they agreed to release them once they find out they are STOS written. By the time they find out, it is too late. They should be judging software on what it is like, not what it is

Inside this issue:

NEWS

written with. Logically, machine code is only used to write games with, so it too is merely a games creation utility !

On the other plus side, there is plenty of third party support for STOS, and the club membership is going well (over 300 have re-subscribed already !). Several companies are developing new extensions for STOS as well as utilities. Apart from ourselves, Goodmans PDL (Who now run the official STOS P.D Library),The Architect, Digital Dimensions, Atlantis Software, Riverdene PDL and others are all supporting STOS.

❑ MEMBERSHIP NUMBERS

In this newsletter, you will find your personal membership number. Due to several magazines printing the STOS/ AMOS helpline number, hundreds of non-members have been phoning the helpline, which as you know, is for subscribers only. Thus when you phone the helpline, please quote your membership number. This will allow us to keep the phone line free so that members can get through !

❑ STOS P.D

As mentioned above, due to the ever increasing size of the AMOS P.D Library, Sandra Sharkey felt that she could not run the STOS P.D Library properly, and so Goodman's PDL will now be handling all the official STOS P.D. Sandra chose Goodmans as it is the best U.K P.D Library, which can handle specialist P.D like STOS P.D with the best care. So from now on, all P.D orders should be sent to Goodmans at the address which is shown in the P.D News section of this newsletter. I hope you join with me in wishing Sandra all the best with the AMOS P.D Library.

❑ GOODIES DISK 2

You might notice a slight change in format for this issue of the newsletter. This is due to the inclusion of around 100 new STOS Commands in the free goodies disk included with this issue. Because of the need to explain most of the new commands, rather a lot of this issue will look very manual like. Things will be back to normal for the next issue.

❑ STOS WORD

Those of you who have started your subscription with this issue, and so missed out on the previous subscription offer of the STOS WORD disk, can still get the disk direct from the STOS Club for £1 (To cover the cost of the disk and postage) quoting your membership number from this issue. STOS WORD is only available for STOS Club Members.

STOS TRACKER, The Official Version

The STOS Tracker extension has been on PD in an unnoficial (ie unsuported) form for a couple of months now. Unfortunately, this version uses the T extension (already in use by TOME) and so cannot be used with any programs that wish to use the TOME extension. The official version on the Goodies disk, has been changed to extension H, and will work with all the other STOS extensions.

❑ Installing

Installing STOS Tracker is slightly different from the other extensions. Because it is such a large extension (180K+ for the 6 extension files) it has been squashed to fit on the disk. To unsquash and install the extension, use the **TRACKIN.BAS** program in the **TRACKER** folder.

Documentation for STOS Tracker is included in the Tracker Folder, along with a demo song.

EXTENSIONS

WHAT ARE EXTENSIONS ?

Extensions are bits you add on to houses to either impress or annoy the neighbours. They are also additions to the STOS language that once installed, give you a set of new commands. STOS can handle up to 25 extensions, each of which has its own new commands. On the goodies disk that came with this issue, there are several extensions with around 100 new commands.

The extension system allows STOS to be upgraded with new commands to cope with improvements to the ST, or just for new command ideas. For instance, the Six Sticks and Blitter extensions (By the Architect) have 50 new commands between them, covering every aspect of the Atari STE's new functions. And just for good measure the Six Sticks extension also allows you to use two joysticks on a normal STFM !

Extensions first need to be installed onto your STOS Disk before you can use the new commands. Thus you should run the **INSTALL.PRG** program on the disk to install the required extensions onto your STOS disk.

□ Compiler Extensions

The STOS Compiler uses a slightly different format for its extensions, and thus, anything that you are going to compile, requires a compiler version of any extensions used in it. Several of the extensions on the disk also have compiler extensions supplied, so that you can compile your programs. Those that are not supplied (such as the **BLITTER, MIDI** and **TOME** extensions) can be obtained with the full versions of these programs. The 512 Colour routine can be obtained in a more powerfull form on the STOS Paint Master Disk (Available from Riverdene PDL). Compiler extensions are installed in the COMPILER folder on your compiler disk, Interpreter extensions are installed in the STOS folder on your STOS language disk.

Please note that due to lack of space, not all the new commands will be covered in full detail in this issue. Commands such as the new Blitter commands are for experts only, and will be covered in later issues.

512 Colour Routine

The 512 Colour picture routine is contained in the **VIEW512.BAS** program in the **OTHERS** folder. Unlike the extensions, the routine is stored as a memory bank, and is simply called by setting the required registers and using the CALL command. It can display pictures created with **STOS Paint Master** (available from Riverdene PDL) a simple demonstration of one of these pictures is included in the **OTHERS** folder.

To activate the routine, you need your picture to be on the screen, with the palette zeroed out, and the address of the bank the 512 palette data is stored in, in **Areg(0)**. You then set **Dreg(4)** to 3 and call the program (in bank 15). i.e **CALL 15**. In the View512 program, all this (and unpacking the picture in bank 13 to the screen) is handled in line 90 once the picture is loaded (.HL0 is a low res picture, .HM0 is medium res. .HL2 is the palette data, .HM2 is the medium res palette data). To turn off the interupt driven display routine, set **Dreg(4)** to 4 and do another **CALL 15**.

Once you've seen what the routine can do, you might think it a good idea to get STOS Paint Master so you can create your own 512 colour pictures for your games !

10-LINERS

10x10

Ten Ten Liners !

Once you add a couple of new commands to STOS it suddenly becomes easier to write more powerfull 10 liner games. Several of this issue's selection uses the TOME junior extension and the STE sixsticks extension, as well as a bit of the Usefull extension. The first game is:

☐ SLIME RUNNER

This one uses the TOME extension, and so is on the disk in the TOMEJR folder (no you don't have to type it in !). The object is to avoid the orange walls of the canyon, while shooting (and avoiding) the green slimy thing that bounces down. The longer you survive, the faster the game will go and the slimy thing will get longer.

☐ Dalek Boxing

Based on the old Tank Battle game, this one is a two player fight between the Red and Blue Daleks who are fighting for the hand in marriage of the fair Crisp Vending Machine. Whoever shoots the other 10 times will win. Use left & right on the joysticks to turn, up moves forward and fire fires. Shots can bounce off the walls and obstacles, and you can be hit by your own shots. This one is stored in the STE folder (it runs on normal ST's as well though). This one uses the STE sixsticks extension to handle two joysticks.

☐ Space War

Again based on a classic game, Space War pits you against another human player (again using Six Sticks), but this time, you are flying around a small neutron star, which is pulling both of you (and your shots) towards it. Control of your ships is Asteroids style in that you rotate the ship with stick left/right, thrust with stick up, and fire with the fire button. Getting hit by an enemy shot, crashing into the neutron star or hitting the other player is lethal. First to 10 kills wins !

Space war is in the STE folder (it runs on STFM's as well) and is called **SPACEWAR.BAS**

☐ Shooting Gallery

This one is for those of you who absolutely can't wait for the next movie licensed game to be released ! It's the same as the shooting gallery level from the next of these tie ins (and the next one, and the next one). It's written in 10-lines of STOS, and by changing the occurences of Xmouse and Ymouse to Xlight and Ylight, as well as changing Fire to FStick(3) you can make it work with your recently converted lightgun.

You get a limited amount of ammo and targets and the basic aim (bad puns as well !) is to shoot everything that appears in the windows with your machine gun (controlled with the mouse if you don't have an STE with lightgun).

It's in the **OTHERS** folder, and is called **SHOOTER.BAS**

☐ Dam Buster

Many of you will probably remember the various Dam Buster games that came out a few years ago, that were based on the famous raid by 617 squadron. Well here's the definitive 10-liner version !

You have to fly the Lancaster bomber in low (control by up/down on the joystick) until the spotlight beams below the aircraft match up on the surface of the water. Then when the two flak towers on the dam match up with your sights upright marks, release the bouncing bomb (fire button). If you get it spot on, you'll bust the dam. Watch out though, as there are several flak guns along the shore.

who will be shooting at you !

The program is in the **OTHERS** folder and is called **DAMBUSTR_BAS**, all the necessary sprites are saved with the program, but it does need the usefull extension.

❑ Cheesyroids

Two strange discoveries have been made:

1) the moon is in fact made up of red and blue cheese, and

2) it has just exploded !

This means that loads of red and blue Cheesyroids are floating around waiting to be blasted. You have a Mousomatic armed Catastrophe fighter. Left/Right joystick rotates the ship, Stick Up thrusts and fire launches a Mousomatic beam against a roid. (Mousomatic beams comprise of thousands of tiny space bred mice, which can eat chunks of the roids). Simply avoid getting hit, and shoot the Cheesyroids. (Try compiling this one, it gets difficult !)

Cheesyroids is in the **OTHERS** folder and is called **CHESROID_BAS**

❑ Six Ball

A bit of a strange one this, it's a cross between pinball and breakout and you have to score as many points as possible by firing 6 balls and trying to keep them bouncing off the boxes (which score different amounts of points). The firing spring, and the small box bat are both controlled with the mouse. The neat thing about this game is that it will run in any resolution ! As with Flea Hill, this one is best when compiled.

❑ Flea Hill

This is a small experimental program (which inspired Six Ball), and it is a case of sit back and watch. As with Six ball, it runs in all resolutions, and is best when compiled.

❑ Attack of the Patio Furniture Eating Spiders

Your garden is under attack by giant sunbed eating spiders, and seeing as you have just bought a new sunbed, you must try and stop the spiders getting over the wall by throwing bits of rockery at them. Control is with the mouse (left & right) and either button throws a rock. The more spiders you kill, the more will appear.

❑ Bouncing Green Alien Blobs

A shoot em up with a difference this one ! You have to shoot the green blobs to blast them back up the screen by firing at them with the left mouse button (you get points for this) alternately you can bounce them off your shields by holding down the right mouse button and hitting them with your ship (you don't get points for this) or let them drop off the bottom of the screen and re-appear at the top (You lose points for this).

You can be killed if you are hit by an alien when you don't have your shield on (the shield won't work if you are firing), and you have a limited amount of shield power (shown by the bar at the bottom of the screen).

Have fun with this lot, and there'll be more in issue 14 (some rather good ones sent in by Simon Wilsher). I must have around 30 10-liners now from various issues, time to get a p.d disk full of them together I think !

If you've written any 10-liners, then send them in so we can put them in the newsletter or include them on the P.D Disk !

INSTALLING

How to Install the new STOS Extensions

Installing the extra STOS extensions on the goodies disk into your own copy of STOS and the STOS Compiler isn't too tricky, as the Architect (who also wrote the STE, Blitter and Musician extensions) has also written a general purpose extension installer program. This program is called **INSTALL.PRG** and is on the goodies disk root directory. Simply double click on this from the desktop and it will load up (and unpack, it was STOS Squashed) and you will be presented with a screen with a menu bar at the top.

There are 4 menus to select from. The first is the **About** menu, which shows who has copyright on each of the extensions, next is the **Extension** menu, with the **Install** menu (one option INSTALL which you use when you want to do the installation) and the **EXIT** menu with one option "EXIT".

First of all, you need to use the **Extension** menu, as this is where you select which extensions to install. The extension menu will have a list of the extensions available on the disk. These are:

Sixsticks (STE) Extension

Blitter Extension

Musician MIDI Extension

Squasher Extension

TOME V2.1 Extension

Usefull Extension

The STOS Tracker extensions are installed slightly differently with the **TRACKIN.BAS** program in the **TRACKER** folder.

To install an extension, select it from this menu. You will get two buttons in the centre of the screen (along with info about the extension) to enable you to select the interpreter and Compiler versions (if compiler version is available for that extension). All you need to do is click on the versions you want with the left mouse button, and an asterisk will appear in that button, showing that it is set (to deselect click in the same button). Clicking the right button will return you to the menu.

You should notice, if you look at the Extensions menu again, that the extension you just selected has an asterisk by it, showing that it is ready to install.

Once you have selected all the extensions you want to install, click on the **INSTALL** option in the **INSTALL** menu. You will then get file selectors, asking you to select the STOS and COMPILER folders to install the extensions into. Simply Select the folders and click on the **RETURN** button (without selecting a file). The installer will then procede to add the extensions to your STOS and COMPILER disks .

The Usefull Extension

The Usefull extension is modified from the one shown in issues 9-12, and includes an extra command. The Interpreter and Compiler versions are on the disk, as well as the sourcecode ! The commands are:

SPRPAL Start(1) Gets the Colour Palette from the Sprite Bank.

=RANGE(A,B,C) If A is within the Range B-C, it returns A, otherwise it will force A into range. (so B<=A<=C)

PRNTER data Sets up the Printer port. See Issue 10 for details.

PROF SPECK O.D.D

ABSOLUTE BEGINERS

with Professor Speck O.D.D

COPING WITH THE NEW BITS

I bet most of you absolute beginers out there have taken a look at the disk and gone "What another hundred !" and then gone "We haven't sussed the first 300 yet !". But don't panic ! Here follows the definitive Professor Speck guide to getting used to new commands by the ancient method of Informed Random speculative experimentation and observation. Otherwise known as reading the instructions and guesswork ! This is about the oldest known method of learning how to work computers, and is still used today by even the top computer programmers. Due to his low memory retention, Aaron Fothergill is rumored to completely guess his way through almost every program he writes ! (A gross libel !...Aaron)

The first stage of this technique is to read the manual where it explains the command you want to learn. For this lesson, we will be using a few of the commands from the actual STOS manual, to explain the principle. What do you mean you haven't actually read any of it yet ?! The STOS manual is renowned as one of the greatest works of almost fiction this century (second only to the AMOS manual !)

Open your STOS manual at page 157 (after digging it out from under the wobbly leg of the kitchen table) and you should the start of the "swodniw dna txet" chapter (You're looking at it upside down idiot !.... Aaron) oops sorry ! "Text and Windows". You should first scan through the page and decide how many words you think must be ancient Mandarin Chinese, and then consult the following chart:

More than 50: Consult the ideal reference work called "JANET AND JOHN GO PROGRAMMING "

11-50: Do the normal and copy the page out, ignoring any words you don't understand. You'll find that it tells you exactly the same, and is easier to read.

6-10: This is about average, and on a par with just about every STOS programmer known and the Mandarin Chinese sub editor who edited the page.

Less than 5: Spotted the lack of Chinese characters didn't you !

Now we'll try the first command on the page, the **PEN** command.

Reading the syntax of the command, we can see that it asks for an **Index** after the command itself. An index is a fancy name for a number or variable. So we could for instance type:

```
Pen 0
```

Having tried this, you will find yourself in what is quite a normal situation for programmers. **USC** (Up Slimy Creek). You can't see what you are typing in, because the colour of the text is the same as the colour of the background ! The easy solution in most of these cases is to make up a small sticker with the word "PANIC!" on it and stick it on the UNDO key. Whenever you find yourself in a situation where you can't see what you are doing, press this key twice. This will reset the screen back to normal so that you can try again. If for instance, the program is in an endless loop, then hit **Control** and **C** at the same time to stop the program before hitting the panic button. If this doesn't work, you are **Farleys** (RUSC Right Up Slimy Creek), reset the ST and start again.

ABSOLUTE BEGINERS 1

O.k, now try typing **PEN 2** and press RETURN, the colour of the text should now change. Great ! Try typing **PEN 93**. Oops **Illegal quantity error** ! This means that the value (or one of the values) you used wasn't in the range of values you were supposed to use. (The word ERROR by the way, is ancient mongolian for "Read the Manual"). If you read the manual about the PEN command, you will see that the range of values for this command is limited to 0 or 1 (in Hires), 0-3 in medium and 0-15 in lo res.

Most of the STOS commands (in fact any programming commands) can be understood by trial and error in this way. Apart from a few commands such as **PEEK** and **POKE**, and some of the new Blitter commands, you should be able to play around without crashing the machine too often.

While you are experimenting in this way, there are several error statements you will come across. Here is the **definitive Professor Speck guide to error statements and their meanings:**

Extension not present: You didn't install it did you !

Illegal Direct Mode: The instruction you tried can only be used in a program.

Illegal Function Call: You didn't use the correct number of parameters (for instance Pen 5,6,7,8,9 when you can only use 1 parameter with Pen)

Out of Memory: Time to upgrade

Syntax Error: You either spelt it wrong, the command doesn't exist, or you haven't installed the extension that uses it. For the spelling of a command, check the manual.

Type Mismatch: Basically, this means you used a string when you should of used a number or vice-versa. If you get either Address Error, Bus Error or Illegal Instruction errors, say bye bye and re-boot the machine !

A double dose of the Prof this issue ! Next up, a beginners guide to bug fixing !

STOS Squasher

The STOS Squasher allows you to compact your STOS Compiled Programs and files, so that they take up less memory and disk space. You can either use the **BDLPACK.PRG** program (on the disk) to pack the program or file, or the **SQUASH** command from within STOS (This can't squash programs in this form).

If you use the **BDLPACK.PRG** on a Compiled .PRG program, then the program will automatically unpack itself on running.

The STOS Squasher extension gives you two new commands:

=SQUASH(start,end)

and

UNSQUASH source,dest

It works like the PACK/UNPACK commands, but you can squash any type of bank (although samples don't normally squash too well).

You should add the extender **.PPC** to your STOS Squashed files, as this is the standard file extender for this format. Just out of interest, the standard extenders for other STOS files are:

.MBK normal memory bank
.ABK STOS Tracker song bank (AMOS memory bank)
.PAC Compacted Picture file (Binary data)
.PPC STOS Squashed data (Binary data)

ABSOLUTE BEGINERS 2

A Beginners Guide to Bug Fixing

by Professor Speck O.D.D

Several people have wondered why the Professor Speck articles (Pseudophysics, Spagetti Code etc) have been included in this newsletter, as "They have no relevance to programming" (Cor..just like Points of View). The same several people have obviously not read the articles properly. Although they are presented in a slightly different form by an outright loony (Oi!....Prof) they contain very major points about programming and game design, they are not merely "silly articles". In any case, the Prof is now writing articles for beginners to STOS, as a sense of humor (however bad...Aaron) is needed when learning to program.

One aspect that beginners will come across, is the dreaded BUG ! In my interview with Sir Hugh St Axe (Issue 10), we heard of the more unusual bugs that can occur, and ways for the experts to locate and destroy them, but it has been pointed out that the programmers who come across more bugs are the beginners, who type in program listings from magazines written by "experts" who manage to always include bugs to test the beginner. This newsletter has had the odd accidental bug or two (or three or four..) in its listings from time to time. Most of the time however, you can spot these bugs with a little know how, and fix them. You can also spot where you went wrong in your program.

There are two major types of bug

1) Errors that stop the program (Such as syntax errors)

2) Bugs that make the program do strange things and not work properly.

For this article, I will include a short, bug ridden, program that we will eliminate all the bugs from. **NOTE...THE BUGS ARE IN THIS PROGRAM ON PURPOSE**, so no helpline calls saying "you know the program in the Professor Speck article ?".

```
10 Rem a very bugged program
20 Ram it should print out the 7
   times table
30 For A=1 to 12
40 Gosub 150 : Rem work out
   A times 7
50 Print A;" X 7 =",B
60 Next B
100 Rem a subroutine to work
   out A times 7
110 B=A * 7
120 Return
```

Type in this program (Don't fix any bugs you spot yet !) and run it. **You should get a syntax error on line 20.** This is a straightforward error, the word **Rem** (a command) was typed in as **RAM** (Not a command !). Simply change it to **Rem** to fix this bit. If you get a syntax error on a line, check all the commands on the line with the manual to make sure they are correct, then check that there are the same number of right brackets as left brackets (they should always be the same number).

If you run the program again, you should get an error on line 40 "Undefined Line Number", which means that the line number asked for by the **Gosub** command, doesn't exist. In fact it should be **GOSUB 100.** Bugs like this are hard to fix if you don't know the program, or what it should be doing, but a bit of detective work can help (like spotting the matching rems !).

ABSOLUTE BEGINERS 2

Fix line 40 and run it again. You should spot a bug and an error. **Fix Errors first, as they are nastier.** You should get a **return without gosub** error in line 120. This sort of error can be tricky, as it is not line 120 which is at fault. It is caused in fact by the program finishing at line 60, and then continuing into the subroutine at line 100 without doing a gosub to it. You must make sure that the only way you can get to a subroutine is by a **Gosub** command. This error can be fixed by adding an **END** command in line 70 i.e

70 End

This will stop the program before it gets to the subroutine.

The bug you should have noticed is that the result of the calculation is being printed away from the A X 7 = bit e.g

3 X 7 = 21

This is not a program stopping error, it is just a minor bug (depending on what your program is meant to acheive these minor bugs can be disasterous !).

It is caused by acidentally putting a comma in before the B instead of a semi-colon (read issue 11/12's introduction to punctuation. Replace it by a semi-colon and all will be well !

Another bug that often occurs, is when you make a program start again after running, with a **GOTO** back to the start of the program. If you **GOTO** a line where an array is being set up with a **DIM** statement, you will get an **"Array already Dimensioned"** error, as arrays can only be Dim'ed once in a program. This can be easily fixed by making the GOTO command goto a line just after the DIM commands (By putting the dim commands at the start of the program).

Before I go, a little tester for you. See if you can fix this program.

```
5 Rem program to generate and
  display a grid of 10x10 numbers
6 Rem as 1-100 times the
  number entered by the user
10 Input "a number from 1-10
  ";N 20 Mode 1 : Key of : Curs
  off : Flash off : Cls
30 Dim X(100)
40 For A=1 to 100
50 X(A)=N X A
60 Next A
70 For A=0 to 10 : rem count
  through 10 numbers
80 For B=0 to 10 : As line 40
90 C=(A+1)*(B+1) : rem
  calculate number
100 Locate B*4,A*2
110 Print X(C);
120 Next A
130 Next B
140 Wait Key
150 Goto 10
```

You should find and fix at least 5 major errors and bugs in this program, including illegal quantity errors, a syntax error, and a mathematical bug that causes the wrong numbers to come out !

Have fun !

Page 6 New Atari User

Page 6 magazine (also known as New Atari User) regularly run a STOS feature in their ST section. Pete Hickman, the journo hack who writes their column has reviewed STOS products, P.D and just about always has some sort of program listing in his column. Well worth a read !

ATARI STE LIGHT GUN

By Steve Bowgen

Having used Stos for some time and having upgraded to a 1040 STE, I thought I would try and write a Shoot 'em up game for my son, using a light gun rather than a joy stick. After a long search I found that the one thing nobody makes for an ST or STE is a light gun. A quick delve in my box of goodies turned up a light gun from a lesser computer that I once owned, (A good old Sinclair Spectrum no less), would it work I thought.

Well a quick peek inside soon proved it would, in fact the conversion only takes about five minutes. Trying to find a 'High Density 15 pin D-sub connector' (That's what they call the plug that fits in the hole in the side of the STE so I'm told) took me a week.

Parts required are:-Cheetah light gun (For Spectrum +3 Computer). Address at end

15 Pin D-connector male. (Tandy Stores)

BEFORE ATTEMPING THIS BE WARNED THIS CONVERSION REQUIRES THE USE OF A SOLDERING IRON.IF YOU CAN'T USE ONE FIND SOMEONE WHO CAN -

Well here's how you go about it. The light gun I have is a Cheetah Defender for the Spectrum +3 computer and it is wired to work off the 12 volt supplied by that the computer. Inside the gun on the circuit board is a resistor which drops the voltage to +5v that the circuit needs. The STE supplies +5 volt so the resistor is not needed and must be removed and replaced by a wire link to make the gun work.

You gain access to the circuit board as follows:-

1) Lay the gun down with the screws uppermost the sights must be removed by sliding them to the back of the gun and the auto fire switch must be lifted off.

2) The red handle must be removed by sliding it down towards the cable and lifting it off. The seven screws can now be removed (see diagram GUN_EXT) and the gun case opened to expose the circuit board.

3) The cicuit board is held in place by two screws. remove these and carefully lift the board. Identify R1 (see diagram GUN_INT) and remove it replacing it with a wire link, replace the board and screw it in position again.

4) Identify the four wires W1-4 and make a note of their colours on my gun the were:-

W1....RED.

W2....BLUE.

W3....WHITE.

W4....YELLOW.

but check your gun in case they are different. After making a note of the colour and position of the wires the telephone type plug on the end of the cable can be removed.

5) The cable must now be connected to 15 pin D-plug. The connection on the plug looking from the wiring side is as follows:-

5 4 3 2 1

10 9 8 7 6

15 14 13 12 11

Connect the cable as follows:-

W1....Pin 6. (Light gun fire)

W2....Pin 7. (+5 volts)

STE LIGHTGUN

W3.....Pin 9. (Gnd)

W4.....Pin 3. (Light gun pos)

6) Replace the cover on the gun and screw in place, refit the red handle, sights and auto fire switch. Fit the cover to the plug following the directions supplied with the plug.

The plug cover will need shaving down as the STE casing is a bit close to light gun socket, how to do it I'll leave up to you.

7) The STE uses the following memory locations to access the light gun

DEEK($FF9220).....LIGHT GUN X POSITION.

DEEK($FF9222).....LIGHT GUN Y POSITION.

DEEK($FF9202).....LIGHT GUN FIRE BUTTON.

But you have the STE Sixsticks extension now, so you won't need to worry about these !

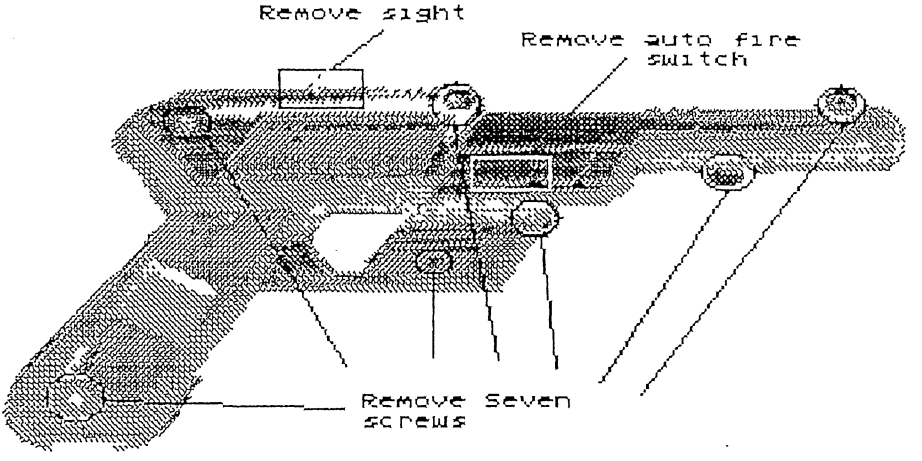
USING A LIGHT GUN FOR SHOOT EM UPS IS MUCH MORE FUN THAN A JOYSTICK.

STEVE BOWGEN, CHESTER

The address of Cheetah is

Cheetah Marketing Ltd
Norbury House
Norbury Road
Fairwater
Cardiff
CF5 3AS
Phone Cardiff (0222) 555525

GUN_EXT



Cheetah light gun

SIXSTICKS EXTENSION

Sixsticks STE Extension

This is part 1 of the STE extension pack by the Architect, part 2 is the Blitter extension (on the disk as an interpreter extension only) and together they give you commands for:

The STE Sound Chip

Six Joysticks

Light gun/Light Pen

4096 colour palette

The Blitter Chip

Hardware Scrolling

Both the Sixsticks and Blitter STE extensions are the sole copyright of Architect and Line productions and may not be copied for distribution on any other disks. The full version (including Blitter compiler extension and a whacking great manual) will be available from them at a later date.

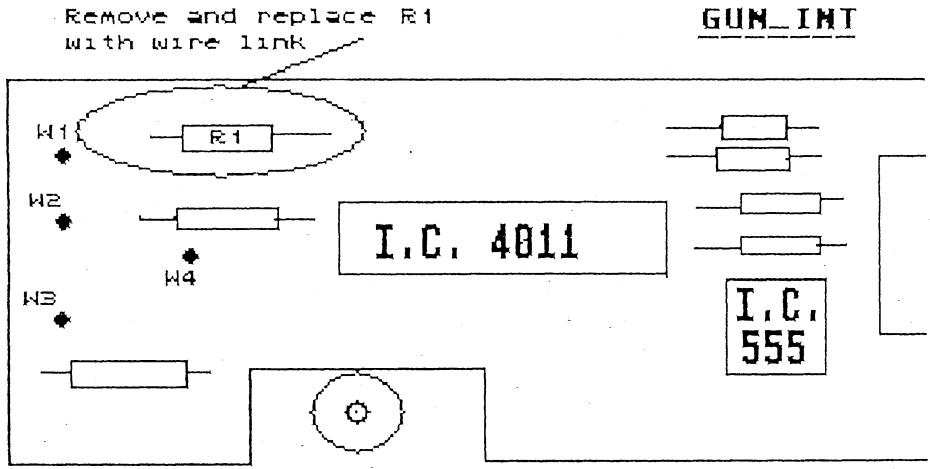
To use the extensions, use the **Install_Prg** program on the goodies disk to install them.

❑ The Joysticks

The STE supports six joysticks (two in the normal ports and two each in Port A and Port B in the side, for which a special cable is needed).

The extra commands to access the sticks are similar to the **JLEFT,JRIGHT** etc commands previously used in STOS (These can still be used). They are:

x=LSTICK(j) left
x=RSTICK(j) right
x=USTICK(j) up
x=DSTICK(j) down



Cheetah light gun
Internal circuit board

SIXSTICKS EXTENSION

x=FSTICK(j) fire

as with **JLEFT** etc, these return -1 if the stick is in that direction. j ranges from 1-6.

To use the normal two joysticks, you will need to use the **STICKS ON** command to activate an interupt routine that will disable the mouse.

The **STICKS OFF** command de-activates this routine, re-activating the mouse.

The Sixsticks extension is also compatible with Twinsticks (it had to be, as it replaces it !), so the two commands:

x= STICK 1

x= STICK 2

have been added and are 100% compatible with twinsticks. These return a bitmap of the joysticks as 1 byte. The following bits are used.

Bit	Use
0	Up
1	Down
2	Left
3	Right
4	Unused
5	Unused
6	Unused
7	Fire

See the 10-liners **Dalekbox.bas** and **Spacewar.bas** in the STE folder for a demonstration of the STICK commands.

☐ The Light Gun/Pen

The STE can also support a light gun/pen. Unfortunately, there isn't an official STE light gun or pen yet ! No problem, you can convert a Cheetah Spectrum +3 lightgun, all you need is the correct plug for the STE end, some wire clippers and the article by Steve Bowgen.

Software wise, you have two new commands

x=LIGHT X

and

y=LIGHT Y

which return the co-ordinates of the light gun/pen the last time the fire button was pressed. To check for the Light gun fire button, use the **FSTICK(3)** function.

☐ Extra Colours

The **PALETE STE** command (Note the deliberate mis-spelling of the Palete command) works exactly the same as the palette command in STOS. However, it uses colour values ranging from $000 to $FFF.

The **STE COLOUR** command allows you to set/check an individual colour. You get

STE COLOR colour,$RGB to set the colour, and

=STE COLOR(colour) to check it. Again, the spelling is deliberate !

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 2.6 works with TOS 1.62 and the Atari STE which is now on sale.

P.D Upgraded

The STOS Public Domain library will no longer accept any PRG versions of programs that have not been compiled with version 2.6 of STOS (i.e any that won't work on the STE). Most of the current P.D library has now been recompiled to run on the STE, and now you've got the STE extension, get writing some STE games !

TOME V2.1

TOTAL MAP EDITOR V2.1

Do you want to write profesional map based games. like Gauntlet, New Zealand Story, Rainbow Islands etc. ?

Did you know that it's possible to do so using STOS ! The TOME (TOtal Map Editor) system gives you 18 extra commands for STOS that enable you to write high speed, pixel accurate scrolling routines just like those used in all the major games ! Over 90% of games written on the ST are map based for their backgrounds or play area, and TOME gives you the power to use maps in your games, saving you huge amounts of memory and giving you a major speed advantage.

The brand new version of the TOtal Map Editor has now been released, including an even more powerful editor, extra commands and two new demo games, including sourcecode and data files. Also included are example programs and a scrolling demo written using TOME.

TOME's features are:

- Single Pixel Scrolling in all directions (Includes simple STOS example to show you how it's done !)
- Over 240 screens can be stored in 64K of memory !
- Block Animation within your maps (So you can do HUGE animations !)
- The MOST POWERFULL MAP EDITOR AVAILABLE on the ST !
- Full 28 Page manual included, and example files on disk, not to mention the two demo games CUTE CUDDLY PURPLE BABY DRAGON GOES FLOWER ARRANGING, and JITTERBUGS][.

"Both Games Galore and TOME deserve to become an integral part of any STOS Programmer's library." Atari ST user March 1990

"If you don't buy it (TOME) you must be raving mad !" Peter Hickman STOS Newsletter Issue 10

Only £14.95 to STOS Club Members (£19.95 to Non

TOME was written by Aaron Fothergill. Send U.K cheque, Postal Order or International Money Order to:

Shadow Software, Barnstaple, N.Devon.

TOME JUNIOR

An introduction.

The TOME Junior extension and editor on this disk is a simpler version of the full STOS TOME 2.1 package available from Shadow Software. The Junior version doesn't include the compiler extension (so you can't compile your programs) and the editor has several of the features removed. There is enough in it for you to use the editor to create your own maps, and to use them in your STOS programs.

The extension. Before using TOME Jr, make sure that you have installed the TOME extension in your STOS folder on your STOS language (backup) disk. You should get the message TOME V2.1 installed when you boot up STOS.

Now load up the TOME Junior editor (TOMEJR.BAS) and run it. It should go to low resolution mode (It will work in high res. but you will need to make up a screen of 32x32 pixel tiles, or convert the supplied picture to hires). Click on the 2nd icon along the top (DISK 1) and click on the Load Tiles icon on the left. If you don't know what any of the icons represent, click on the **HELP** icon on the left menu. It will list all the icons and what they are about !

Once you have clicked on the Load Tiles option, a file selector will appear so that you can select a picture file to load (.PAC format, STOS Compacted). Select **TOMETEST.PAC** from the TOME folder and it will load in.

Now click on the Load Map option, and select **TOMETEST.MAP** from the file selector, and this map will load in.

All TOME maps are made up from **TILES**, which are stored as a screen in bank 5. The positions of these tiles in the map are stored in the **MAP**, which is stored in bank 6. Each tile can be repeated as many times as you want in the map, and the map can be up to 32767x32767 tiles in size (around 15 million screens).

Play around with the map. You can scroll around using the joystick, the arrow keys, or by clicking on the arrow buttons in the top right corner. By clicking on the 9 main menu icons along the top bar, you can select each of the 9 menus. Then you simply click on the icon you want on the left menu to activate the function (Remember, some are only available in the full version of TOME). The EDIT 1 Menu (the fourth one) contains the various modes of placing tiles on the map. You have:

Draw: Normal placing of tiles
Box: Draws filled boxes with the current tile
Circle: Draws filled circles with the current tile
Fill: Flood fills the screen area with the current tile

You can pick what tile you want to use from the tile viewer at the bottom of the screen, by clicking on the PICK TILE icon, or by clicking the right mouse button when on either of the EDIT menus.

Most of TOME's functions are self explanatory, but we will be covering more of them in later issues.

Programming with TOME

One of the things you can use TOME for, is multi directional scrolling games. In the TOME folder on the goodies disk, you'll find the **SCROLL8.BAS** program, which contains all the routines you need to do pixel accurate 8 way scrolling games. For your own games, you just have to change the tiles, map and sprites loaded, and use the display routines (Lines 100-209) and you can create your own scrolling game with little hassle. Included

in this version of scroll8 are joystick control, push scroll (where the map display moves when you are on the edge of the screen, like in Gauntlet) and collision detection against the walls. Not to mention the routine to display another sprite on the map, so that it is only seen when it is in the display area.

You will need the TOME extension and the Usefull extension (used for the **RANGE** function) installed to use this program.

□ Program Breakdown

Lines 10-25: Set up screens and load in graphics and map, as well as the tile values required.

30-39: Initialise variables for player and target sprite (the man you have to find in the maze)

40 Gosub to display routine

45 memorise last plotted map and sprite co-ordinates

50-54 do joystick control of sprite

60-66 handle push scrolling of map, and keep map co-ordinates within range

70-72 check for collision detection with walls

75 check for collision detection with man.

90 if man not found, then go back to line 40 and do the loop again

91 end game

100-149 main sprite and map display routine

150-151 sprite engine routine

200-207 map refresh routine (for scrolling)

□ The new TOME commands used:

MAP BANKS m1,m2,m3,m4,t

This command tells TOME which memory banks contain the 4 maps and tilescreen to be used. If you don't need to use 4 maps, simply use the same bank address for all the 4 maps.

N.B the parameters must be the **ADDRESS** of the banks (using the **START(n)** function).

MAP VIEW x1,y1,x2,y2

Determines the area of the screen that the map will be displayed in.

=MAPX(n),=MAPY(n)

These functions returns the width and height (in tiles) of map number n. If n is zero, then it will return the width and height of the map in use.

DO MAP scrn,x,y

This draws the viewed area of the map to the screen scrn (again, this must be a memory address, i.e logic,back,physic, or the start address of a screen bank).

The parameters x & y control the point on the map that you will be viewing.

MAP LEFT scrn,x,y

MAP RIGHT scrn,x,y

MAP TOP scrn,x,y

MAP BOTTOM scrn,x,y

These four commands are similar to **DO MAP**, and in fact use the same parameters, but they only draw 1 line of tiles at the edge of the viewing area. This can be used for high speed scrolling.

Generally, you should load in all your tiles and maps, then do the **MAP BANKS** command, and the **MAP VIEW** command before you use any of the map displaying commands such as **DO MAP**.

When setting up the banks used by TOME, you need to make sure that you don't change any memory banks (by erase, loading or reserving). If you do, then you must use the map banks command again to reset the bank addresses.

□ Explainations of the routines.

The scrolling routine relies on the variables **MX** & **MY** as its control co-ordinates. Thus by changing these variables (in 1-16 pixel increments), you can move around the map. With a push scroll (as in this case), you only do this when the player's sprite reaches the edge (or near to the edge) of the play area.

MX & MY actually contain the pixel co-ordinate of the top left of the Map display. Other variables used by the routine (so you shouldn't use them) are:

HX & HY Tile co-ordinates of the map

HXO & HYO last plotted of the above. You can force the map to completely redraw with a DO MAP by setting HXO to -1.

FX & FY offset (in pixels) of the display area.

DX,DY,DX2,DY2 various variables used to scroll the map.

The player sprite uses the variables X,Y and R to contain the screen co-ordinates of the sprite and its rotation. The map co-ordinates of the sprite are calculated by using (MX + X)/16 and (MY +Y)/16, as used in lines 71 & 72, where collisions with walls are checked for.

□ The "bounce"

When the player hits a wall, (detected in lines 71 & 72), it is necessary to put the player back on his old co-ordinates, so that it doesn't look as if the sprite has gone into the wall. By keeping a copy of the MX,MY,X & Y co-ordinates at the last time the sprite (and map) were plotted, when the player contacts a wall, simply reseting the co-ordinates to these copies stops the sprites going into the wall. The routine in this program is a variant, which checks X & Y independantly, allowing the player to move along a wall even if bumping into it, giving better control. To use the shorter, but less usefull variant, delete line 72, and change line 71 to:

```
71 If Tile Type(0,(MX+X)/
 16,(MY+Y))>49 then X=XO :
 MX=MXO : Y=YO : MY=MYO
```

□ Making the player more interesting (standard walking routine)

To make the program look easy (ha ha), I've used a simple 4 rotation sprite for the player, which doesn't animate as it moves. Now most of you are going to want to do an animating sprite, which walks in the direction it is going. In the sprite bank, are the sprites for a man walking in 4 directions, 2 frames per direction (These are from the Jitterbugs II game on the TOME2.1 disk).

The variable R is used to show which direction the man is moving in, and thus which sprites to use. In the bank, these sprites are stored in the following order:

Left1, Left2, Up1, Up2, Right1, Right2, Down1, Down2

and the values of R represent

0=Left,1=Up,2=Right and 3=Down

So by simply multiplying R by 2, we can get the basic frame number for the direction the sprite is travelling (remembering to add one, as the sprites start from sprite number 1). Making the sprite walk (i.e animate between the two frames) is a simple matter of adding a toggled variable to this calculation. So if we used the variable WT for the Walking Toggle, the sprite number to display (on line 105) would be

```
S=R * 2+WT+1
```

So change line 105 to:

```
105 SX=X: SY=Y : S=R *
 2+WT+1 : Gosub 150
```

You now need to change the 4 joystick movement lines, to handle the toggling of WT (between 0 and 1) when you move, so:

```
51 If jleft then X=X-SP : R=0 :
 WT=1-WT
52 If jright then X=X+SP : R=2 :
 WT=1-WT
53 if Jup then Y=Y-SP : R=1 :
 WT=1-WT
54 if jdown then Y=Y+SP : R=3
 : WT=1-WT
```

This causes problems when you move diagonally (WT gets toggled twice) and your sprite doesn't animate. This can be solved by changing all the WT=1-WT's in the above 4 lines to M=1 and adding the following line

```
55 If M=1 then WT=1-WT
```

O.k, that should keep you going for a while, more on TOME next issue.

HELP !

If you are stuck and need help with your programming. Just phone the

STOS Helpline on 027

It is best if you phone between 1pm and 7pm U.K Time, as I'm most likely to be in then.

Overseas members please note ! Check what time it is in the U.K before you phone, it isn't funny being woken up at 4a.m !

As soon as this newsletter is sent out, the helpline will only be open for STOS and AMOS Club subscribers, so make sure you have your membership number handy when you call !

□ Public Domain News

The first bit of P.D News this issue is that Sandra Sharkey is no longer running the STOS P.D Library. Due to the astounding response to the AMOS P.D Library, Sandra felt that she could no longer give the level of support to the STOS P.D library that she thought necessary, so the STOS P.D Library has now been moved in with Goodman's PDL which Sandra thought was the best to run the STOS P.D. So from now on, please send all P.D orders to Goodman's PDL.

Goodman Enterprises

Longton,

Stoke-on-Trent.

Right, now for some really good news ! I've just got my mits on a new STOS P.D disk, **"The Revenge of the Routines disk!"** by Dattrax Software. As with the Dattrax Disk 1 (reviewed in issue 10) it is full of short but very inventive STOS programs, which show some brilliant programming ideas. Such as a very simple pseudo fractal landscape generator and some great scroll and fade effects, not to mention 32 colour screens (without using machine code). The Sunset program showing a beach at sunset with waves moving up and down the beach should have an award for the most inovative program that doesn't actually do much ! I am now waiting for the final version of this disk, which should include more short game ideas .

Brilliant 9/10

MIDI Extension

MIDI Bits

Also in the STE Folder is the STOS Musician MIDI extension (also written by the Architect) which gives you basic handling of the MIDI ports in a simpler (and faster) form than normally done in STOS. This extension is in interpreter form only, the full Musician extension will be available with STOS Musician, if the project is restarted.

The Musician MIDI extension gives you 4 simple commands. These are:

```
MIDI ON buffstart,buffend, start,
 filter,end
=MIDI IN
MIDI OUT
MIDI OFF
```

To start the MIDI routine working (it is interupt driven), you must use the **MIDI ON** command. The parameters are:

buffstart= the start address of a bank you will be using as your MIDI buffer (normally about 256 bytes).

buffend= the end address of this bank

start= start code. If not set to 0, the MIDI routine will only start accepting data once it receives this code (so you can wait for particular messages) .

filter= filter code. If not set to 0, any occurences of this value will be filtered out by the MIDI routine (Handy for getting rid of $FE active annoyance messages from some synths) .

end= end code. As with the start code, if not set to zero, the routines will stop receiving data once this code is received.

For instance, you can set up to routine to receive a system exclusive message (always starts with $F0, always ends with $F7) with the following 2 commands:

```
10 Reserve as work 5,4096 :
 rem sys ex messages are
 usually quite long
20 MIDI ON start(5), start(5) +
 length(5) , $F0 , 0, $F7
```

The **MIDI IN** function pulls the next byte of data from the buffer. If it is -1 then there is no data available.

The **MIDI OUT** command allows you to send 1 byte of data to the MIDI OUT port. Of course, you can do multiple **MIDI OUT** commands to send longer messages.

The **MIDI OFF** command switches off the MIDI interupt.

Two example programs are on the disk in the STE folder. These are:

MIDICOM.BAS: MIDI comunicator , allows you to communicate between 2 ST's, simply run the program on both machines and type in your messages to each other.

MIDITEST.BAS: MIDI message analyser, tests incoming MIDI commands from synths, and tells you what they are.

I'll be covering more programming using these commands in a later issue, such as how to write MIDI linked games.

Membership Number

Below is your personal membership number, which you must quote when using the STOS Helpline. Make sure you don't lose this newsletter, and copy the number onto further issues of the newsletter as you get them. This means that you will be able to have the number ready when phoning.