Desmond’s C64/MAX RAM Test Thing (SwiftLink Edition)

Matthew Desmond
factorofmatt.com

Table of Contents

Introduction

This edition of DesTestMAX (DesTestMAX-SL) uses the SwiftLink-232 (or compatible) cartridge to communicate diagnostics information rather than attempt to use the C64's VIC and (possibly bad) memory.

Sending diagnostics using the SwiftLink requires no system RAM and leverages the fact that we can use the remote terminal to display (and possibly buffer) all fault information rather than attempt to store and display it locally.

DesTestMAX-SL has been successfully used with an original SwiftLink-232, the CMD SwiftLink re-release, the CMD Turbo232 and a home-brew clone.

Building a DesTestMAX-SL Cartridge

The DesTestMAX-SL code runs from a standard Commodore 64 cartridge configured in the following way:

- /GAME pulled low to enable MAX mode (/EXROM can be pulled high or left floating)
- 8K EPROM enabled by /ROMH ($E000-$FFFF)
- A reset button (if available) is highly recommended.

You can find myriad C64 cartridge PCB solutions and any of them should work for this purpose. In the simplest of cases you just need to program and install an EPROM then configure a couple of jumpers and you're all set.

If you already have a Dead Test style cartridge it may be simple task of replacing the ROM/EPROM with a new one containing the DesTestMAX-SL code.

It is recommended that you first test your new DesTestMAX-SL cartridge with a working C64 or C128 so you'll know what to expect when using it to test an unknown machine.

Two plugs, one socket

Since both the DesTestMAX-SL and the SwiftLink use the cartridge port, you'll require a cartridge expander. Many such expanders are available both vintage and modern: even the simplest offering should work.

As with DesTestMAX, you may choose to replace the kernal ROM with a DesTestMAX-SL image. This may be the better (only) option if you don't own a cartridge expander. The only major difference here is that the VIC-II display may be garbled (harmlessly) if the RAM is severely compromised.

The Terminal Environment

Hook the RS232 connector of your SwiftLink to the RS232 connector of whichever terminal you plan to use. You'll most likely require a NULL-modem adapter in the mix for this task. The details here are reliant on the hardware and operating system being used. Modern computers with no built-in RS232 serial port may require a cheap USB to RS232 adapter and appropriate drivers.

Any terminal program capable of displaying the ANSI X3.64 ('ANSI') or VT102 control sequences with an 80 column-display will suffice. Colour is not required for the diagnostic information to be understandable.

The communication protocol should be set to 38400 baud, no parity, 8 data-bits and 1 stop-bit. (38400, N,8,1 or 38400,8,N,1).

Notes:

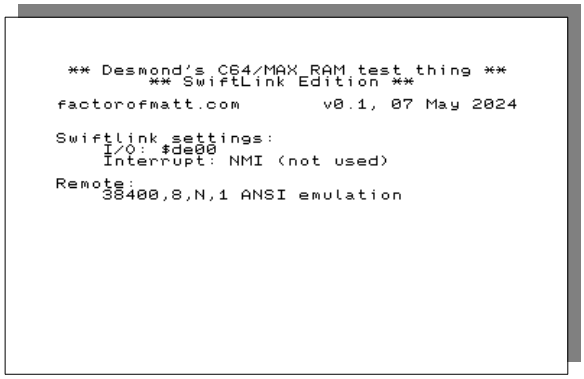
If DesTestMAX-SL continues to display a white VIC-II border at startup (and indeed, nothing shows up on the remote terminal) then you may wish to check that the CTS line is properly connected and is set to High. This may be affected by flow-control settings in the remote terminal. Additionally, it is recommended that the remote terminal be started first.

There is no explicit support for flow-control built into DesTestMAX-SL, though transitions in CTS will be honoured directly by the 6551 ACIA. Xon/Xoff may be added in a future revision.

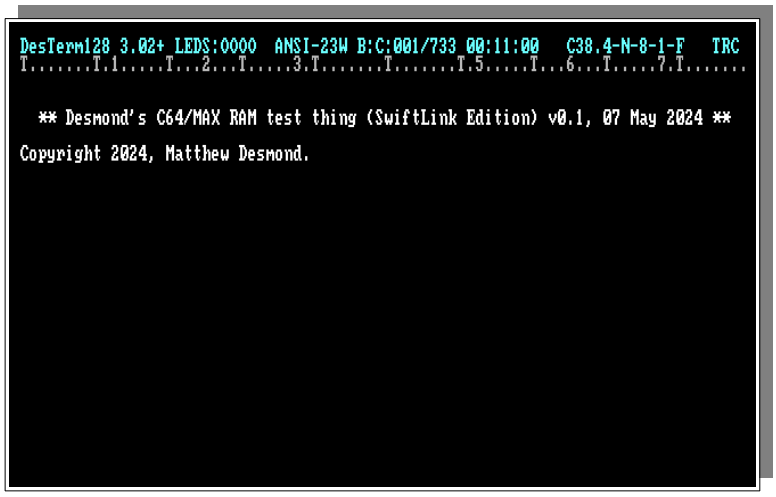
The 6551 ACIA will generate an interrupt (be it IRQ or NMI) if there is a change in the DSR or DCD lines, so stopping your remote terminal or disconnecting the RS232 cable may stop testing in progress (likely not a huge problem) until your C64 is power-cycled or reset. Some SwiftLink clones may be able to disconnect the interrupt source entirely which is a nice option if you tend to fiddle with cabling.

Running the Diagnostics

At startup, DesTestMAX-SL will display a static screen on the VIC-II display. This display comes directly from the DesTestMAX-SL EPROM and doesn't require any working RAM. A working CPU, PLA and VIC-II will be required to see the VIC-II display.



Additionally, a startup banner is sent to the terminal via the SwiftLink. If this initial banner does not appear in your terminal then check your physical RS232 cables and your protocol settings. Naturally a working SwiftLink is required.



Startup Tests

When the cartridge first starts it tests the VIC-II, Zero-Page, the Stack Page and the checksum of it's own ROM.

- The VIC-II has 47 registers mapped into the $D000 block of address space. We don't use all the features of the VIC-II during testing, so many of these registers can be considered as general

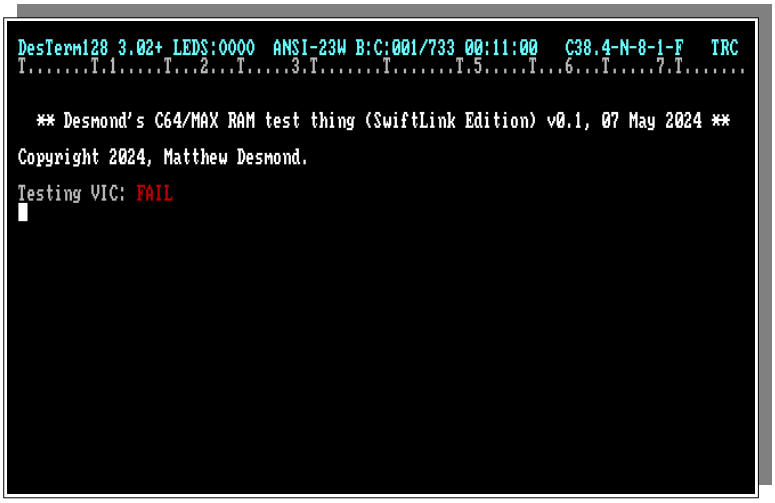
read/write ‘memory’. Testing is performed on these registers to verify that they can be written and read as expected.

- A 16-bit checksum is calculated for the entire contents of the DesTestMAX-SL code ($E000-$FFFF). If the checksum is incorrect the EPROM image could be corrupt or could indicate that address decoding logic in the C64 is faulty.
- The Zero Page is one of the two 256 byte memory pages treated specially by the 6510. It would be impractical to write a full memory test without using at least some of zero-page, so we test it early here. The test does not use zero-page (or the stack) to do so.
- The Stack Page is the other 256 byte memory page treated specially by the 6510. The stack allows the use of subroutines (JSR/RTS). We test the stack page (without using it or zero-page) so that we have some confidence we can use subroutines for the more comprehensive tests.

An error detected in any of these first 4 tests will cease testing and the display updated to indicate the failure:

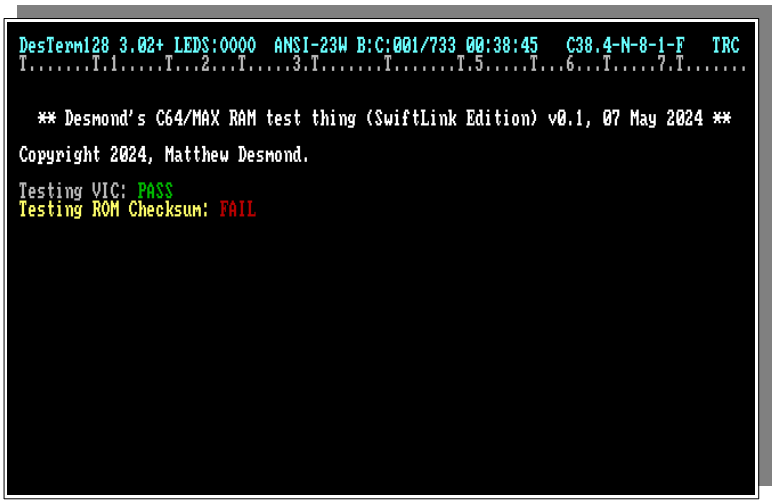
VIC Test Failure

A failure during the VIC test results in a VIC-II grey border and all testing ceases.



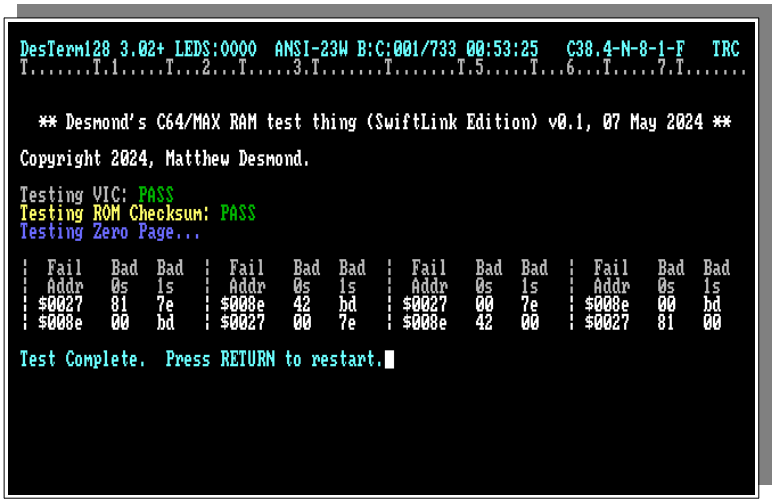
Code Checksum Failure

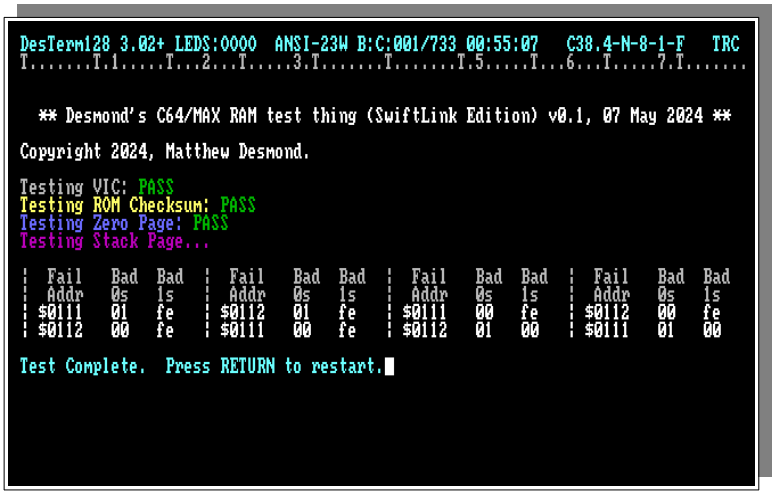
A checksum failure is indicated by a VIC-II white border and a message is sent to the terminal.



Zero Page or Stack Page Failure

A failure during either of these tests results in a blue (zero-page) or purple (stack page) VIC-II.





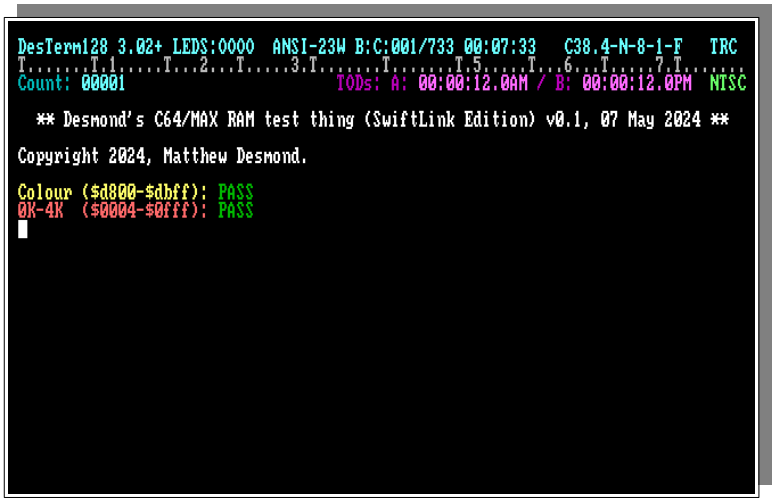
In either of these cases, all addresses and bit-failures are sent to the terminal. See the Failures section for details about diagnostic output. You may restart the testing process by pressing return on the terminal.

Main Tests

After the startup tests complete we have enough confidence in the system to go ahead and test the entire 4K of memory. Unlike DesTestMAX, there are only 2 tests here:

- Colour RAM (orange VIC-II border): VIC-II colour information is stored in this region.
- All RAM (brown VIC-II border): All 4K is tested at once.

Since the VIC-II is not required to show any information, there is no need to split the memory into smaller sections in the hopes that the VIC-II might see enough good RAM for a valid display.



At the top of the terminal screen is the count of the number of times we've looped through all the tests. Each cycle takes about 15 seconds. The TOD information (derived from the CIAs if they're present) is mostly for decoration though may indicate CIA issues if the times look crappy.

The tests will loop forever if no failures are determined.

Any test failure will halt testing, though tests in the current cycle are finished first: we might as well collect as much information as we can.

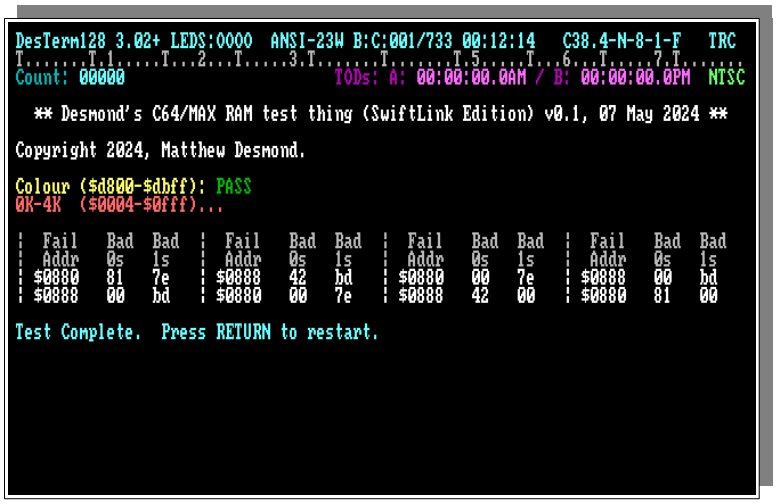
Failures

In the event that a memory fault is detected, an error-record is sent to the terminal (4 per line on 80 columns). Each error record is 8 hex digits wide:

- First 4 digits: The address at which an error was detected
- Next 2 digits: An indication of bits that were written as 0 but were read as 1
- Last 2 digits: An indication of bits that were written 1 but were read as 0

An error record of '$0880 81 71' indicates that at address $0880 bits 0 and 7 ($81) were mistakenly read as 1 and bits 1, 2, 3, 4, 5, 6 and 7 ($7e) were mistakenly read as 0. Patterns may emerge both in addresses and bad bits that can indicate the source of problems.

Note: Addresses usually appear multiple times since the testing algorithm walks through memory both backwards and forwards a couple of times each (and there is no spare memory to catch duplicates).



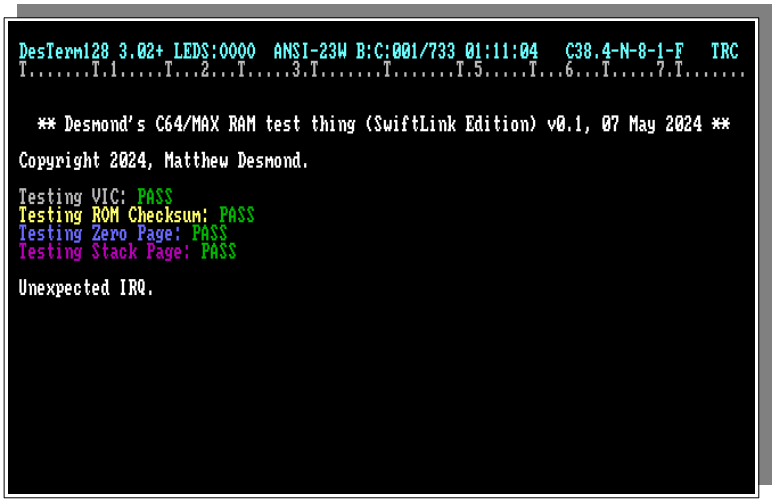
Interrupts

An unwanted interrupt could cause havoc during the middle of a test: interrupts write to the stack which really wouldn't be great if we're testing that memory or if that memory is suspect. The SwiftLink can generate interrupts (IRQ or NMI via a switch), though DesTestMAX-SL disables them

the best it can. Changes in DSR and DCD will still generate such an interrupt (at least on some variants of the 6551 ACIA), though in general use this shouldn't be an issue.

IRQ

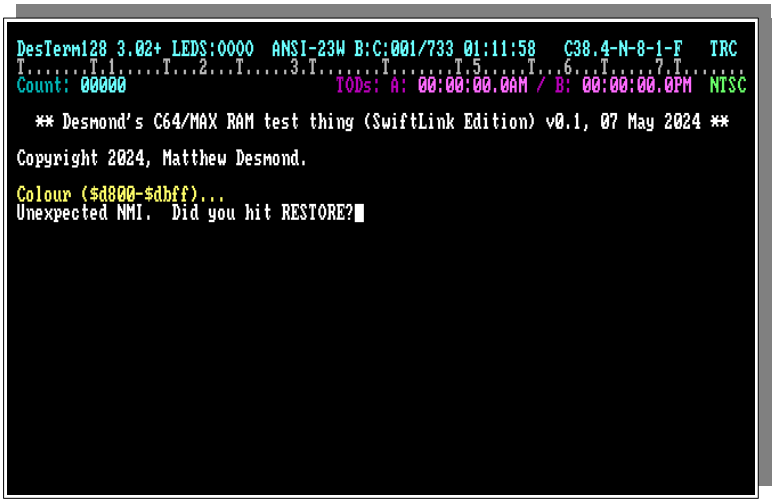
Under normal circumstances no IRQs should occur during normal DesTestMAX-SL operation. The CPU is left free to respond to IRQs and will display the following message should any occur.



The receipt of an IRQ during the execution of tests could very well indicate something wrong with the interrupt-signal path or a misbehaving interrupt source (VIC-II, CIA or SwiftLink etc.).

NMI

NMIs cannot be disabled. The best we can hope is to ensure that any sources of an NMI are disabled. Unfortunately the RESTORE key when tapped will always generate an NMI and there's really nothing that can be done about it. If an NMI is received, the following message is displayed:



If you see this screen and did not hit RESTORE then something (a CIA or indeed the SwiftLink) generated an NMI or something is wrong in the NMI signal path.

If you hit the RESTORE key (possibly in frustration if nothing seems to be happening) and you see this message, then at least you know the CPU works.

Limitations

Memory Errors might not be the memory at all. The shared address and data busses in the Commodore 64 are susceptible to accidental hijack by malfunctioning ICs. If a chip writes data to one of these busses when it shouldn't then it can very easily seem like a RAM error when in reality the bus is being corrupted elsewhere. A useful technique can be to remove all non-essential chips in case they are dirtying a bus. The ROMs, CIAs and SID can all be removed (if socketed) and might provide clues if symptoms change after. DesTestMAX-SL is happy to run without those chips.

Testing methodology

The memory testing algorithm used in DesTestMAX-SL is called March-B. A good description of common memory problems and test methodologies can be found here: <https://redirect.cs.umbc.edu/~reza2/courses/418/Slides/15MemoryTest.pdf>

The March B test performs 4 testing passes over the memory-region-under-test and ultimately verifies that any read or operation performed on a given bit is correct and doesn't affect any other bits in the region. The test is order 17N meaning that each bit under test is written and read a total of 17 times during the test. The test of the entire 4K region available in MAX mode takes about 10 seconds.

Good care has been taken to ensure that no assumptions are made about the validity of memory before it has been tested. Neither Zero Page nor the Stack are used before those two memory regions have been verified since errors in either would cause havoc with the running code.

From Matt

If you've ever used the Commodore Dead Test then chances are you'll find DesTestMAX-SL a useful addition to your diagnostics arsenal. Please do give it a try. And tell me what you think.

Though I've worked hard to ensure that DesTestMAX-SL will give reliable, accurate results under the widest set of circumstances I just haven't been able to physically test much more than removing chips and inducing incorrect behaviour with jumper wires.

I'd like to see how DesTestMAX-SL works out there in the real world. I'd like to hear about your experiences:

- Does it work at all for you?
- Does it reliably show a specific, traceable fault?

- Does it give misleading or plainly incorrect results?
- Could it be made more useful?
- Does the VIC-II font make your eyes hurt?

Please send feedback to destest@factorofmatt.com

Thank you, -M@