

Random Access Memory:

One of the most common failures that occur on arcade PCBs are failures in Random Access Memory (RAM). These failures will bring a grinding halt to your boards. Without working memory the CPU will not be able to do much work at all (often not even running a basic amount of self test code). Even though we are just now covering testing RAM it's actually one of the first things I test, because if you have a good clock, good reset, good CPU and good RAM often it's enough to get into the game boards self test, which will help you narrow down any other issues such as bad ROMS etc. However for the purpose of learning, there are many concepts you must first understand before trying to test RAM, and that is the reason it is presented last.

Understanding memory maps

If you have a Fluke you can use it to read and write to RAM as well run automated tests against the RAM. However to test RAMs you first must know the range of memory that contains the RAM (just like you need to know the range of memory that corresponds to ROMS). As you learned in chapter 4 the table of different addresses correlate to which chips or circuits (RAM, ROM, button inputs etc) is called the "memory map".

There are a few ways to determine the memory map.

1. Read the schematics and determine for each RAM chip, which combination of address bits must be "on" for the RAM to be enabled. You should always be able to perform this step if you have the schematics, however it may take you a while to do and it's easy to make mistakes.
2. Often the schematics will show you the RAM ranges (Atari games are very good about this). The example memory map from the Atari Centipede schematics is shown below. Notice there are 4 columns
 - a. Hexidecimal Address: The address range this row relates to.
 - b. R/W: Whether that range can be read from, written to or both.
 - c. Data: Shows how many bits of memory the system uses at this memory range. Each memory address almost always contains 8 bits of data, however some parts of the circuit do not necessarily use all the bits that the RAM provides. This column shows what bits are used at the different memory locations. Notice the "Motion Graphics Color range" (0x0FF0 - 0x0FFF) which determines the colors of the sprites drawn to screen only use 6 of the 8 available bits.
 - d. Function: A short description of the function of this address range. As you can see in the memory map addresses 0x0000 - 0x03FF are the games working RAM. The DIP switches are mapped to addresses 0x0800 and 0x801 with each bit representing one of the 8 switches on each DIP, and range 0x2000 - 0x3FFF are the game code (ROMS)

Memory Map

MEMORY MAP									
HEXA-DECIMAL ADDRESS	R/W	DATA							FUNCTION
		D7	D6	D5	D4	D3	D2	D1	
0000-03FF		D	D	D	D	D	D	D	RAM
0400-07BF		D	D	D	D	D	D	D	Playfield RAM
07C0-07CF		D	D	D	D	D	D	D	Motion Object Picture
07D0-07DF		D	D	D	D	D	D	D	Motion Object Vert.
07E0-07EF		D	D	D	D	D	D	D	Motion Object Horiz.
07F0-07FF				D	D	D	D	D	Motion Object Color
0800	R	D	D	D	D	D	D	D	Option Switch 1 (0 = On)
0801	R	D	D	D	D	D	D	D	Option Switch 2 (0 = On)
0C00	R	D				D	D	D	Horizontal Mini-Trak Ball™ Inputs
	R		D						VBLANK (1 = VBlank)
	R			D					Self-Test (0 = On)
	R				D				Cocktail Cabinet (1 = Cocktail)
0C01	R	D	D	D					R, C, L Coin Switches (0 = On)
	R				D				SLAM (0 = On)
	R					D			Player 2 Fire Switch (0 = On)
	R						D		Player 1 Fire Switch (0 = On)
	R							D	Player 2 Start Switch (0 = On)
	R							D	Player 1 Start Switch (0 = On)
0C02	R	D				D	D	D	Vertical Mini-Trak Ball™ Inputs
0C03	R	D	D	D	D				Player 1 Joystick (R, L, Down, Up)
	R					D	D	D	Player 2 Joystick (0 = On)
1000-100F	R/W	D	D	D	D	D	D	D	Custom Audio Chip
1404	W					D	D	D	Playfield Color RAM
140C	W					D	D	D	Motion Object Color RAM
1600	W	D	D	D	D	D	D	D	EA ROM Address & Data Latch
1680	W					D	D	D	EA ROM Control Latch
1700	R	D	D	D	D	D	D	D	EA ROM Read Data
1800	W								IRQ Acknowledge
1C00	W	D							Left Coin Counter (1 = On)
1C01	W	D							Center Coin Counter (1 = On)
1C02	W	D							Right Coin Counter (1 = On)
1C03	W	D							Player 1 Start LED (0 = On)
1C04	W	D							Player 2 Start LED (0 = On)
1C07	W	D							Trak Ball™ Flip Control (0 = Player 1)
2000	W								WATCHDOG
2400	W								Clear Mini-Trak Ball™ Counters
2000-3FFF	R								Program ROM

- If you pull the MAME project source code, and find the driver file for the game in question often the developers will document it.

Centipede Memory map and Dip Switches

Memory map for Centipede directly from the Atari schematics (1981).

Address	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Function
0000-03FF		D	D	D	D	D	D	D	D	RAM
0400-07BF		D	D	D	D	D	D	D	D	Playfield RAM
07C0-07CF		D	D	D	D	D	D	D	D	Motion Object Picture
07D0-07DF		D	D	D	D	D	D	D	D	Motion Object Vert.
07E0-07EF		D	D	D	D	D	D	D	D	Motion Object Horiz.
07F0-07FF				D	D	D	D	D	D	Motion Object Color

It does not matter how you obtain the memory map, as long as you do obtain it. The memory map is critical as it lays out all of the input and output to the system. You can see where RAM is, where ROMs are, what addresses hold the status of the buttons and joysticks etc, you can even see what addresses may trigger certain sounds, if the game uses separate (discrete) circuits for each sound. For the purpose of this lesson we are only concerned with obtaining ranges of memory associated with RAM. For our purposes these are the range for the main system RAM (RAM) and the RAM used to interface with the graphics hardware to draw images to the screen (Video RAM)

Function	Start address	End address
RAM (main RAM)	0x0000	0x03FF
Video RAM	0x0400	0x7EF

Note: In the memory map above, you may notice that addresses 0x07F0 – 0x07FF do not have a “D” in the leftmost 2 columns on the memory map. This means that those 2 bits are not actually used for anything, and often you’ll need to exclude ranges that do not use all 8 bits from your RAM test as it may confuse the Fluke.

The most important RAM range is the main system RAM, usually called "RAM", "work RAM", "scratchpad RAM", or sometimes "Page Zero RAM". RAM that holds any of these labels are the most important as it is this RAM that the CPU uses directly to do calculations and is critical for fundamental operations. Without these RAMS fully working the game will probably not boot at all, even just into test mode. The "Video RAM" is less important as it is usually written to when the system wants to draw images to the screen, it is not usually required for entering "test" modes or running the main game code. Therefore you usually want to test the main system RAM first when trying to troubleshoot. For example on a Centipede board if the clock, CPU, and the 2114 RAMs are working correctly you can usually run the self test diagnostics and the system will be able to usually TELL YOU what video RAMs or ROMs are not working. Test modes on many boards provide many different diagnostic tests and usually also allow you to view the game sprites (graphics) and play sounds etc. Using the systems self test can save you A LOT of time and effort in fixing a game.

Understanding Fluke RAM errors the "BTS" error.

The first and easiest RAM error you may receive is "BTS error" which is reported like

R/W ERR @ *aaaa* BTS *xx* – LOOP?

In this case *aaaa* is the memory address where an error occurred (in hex) and *xx* is a hex number (usually 2 digits) that describes the bits at that address the Fluke was UNSUCCESSFUL in reading or writing to. For example if we received the error

R/W ERR @ 0000 BTS 08 – LOOP

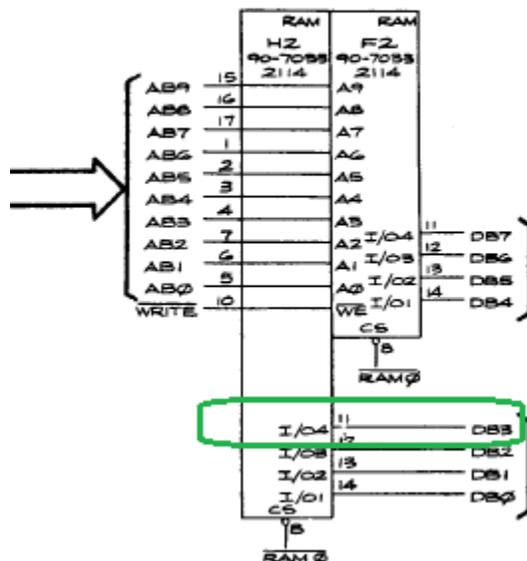
That means the Fluke encountered an error at address 0x0000 where it was unable to set bit 08. In this case 08 is a hex number when turned to binary has a "1" in all the bit positions that were not writable.

Hex error code	0	8
Binary representation	0 0 0 0	1 0 0 0

Here the Fluke has indicated that the 4th bit from the right (often called line D3) does not seem to be working. At this point you'd want to find what chip corresponds to DB3 on the RAM at location 0x0000.

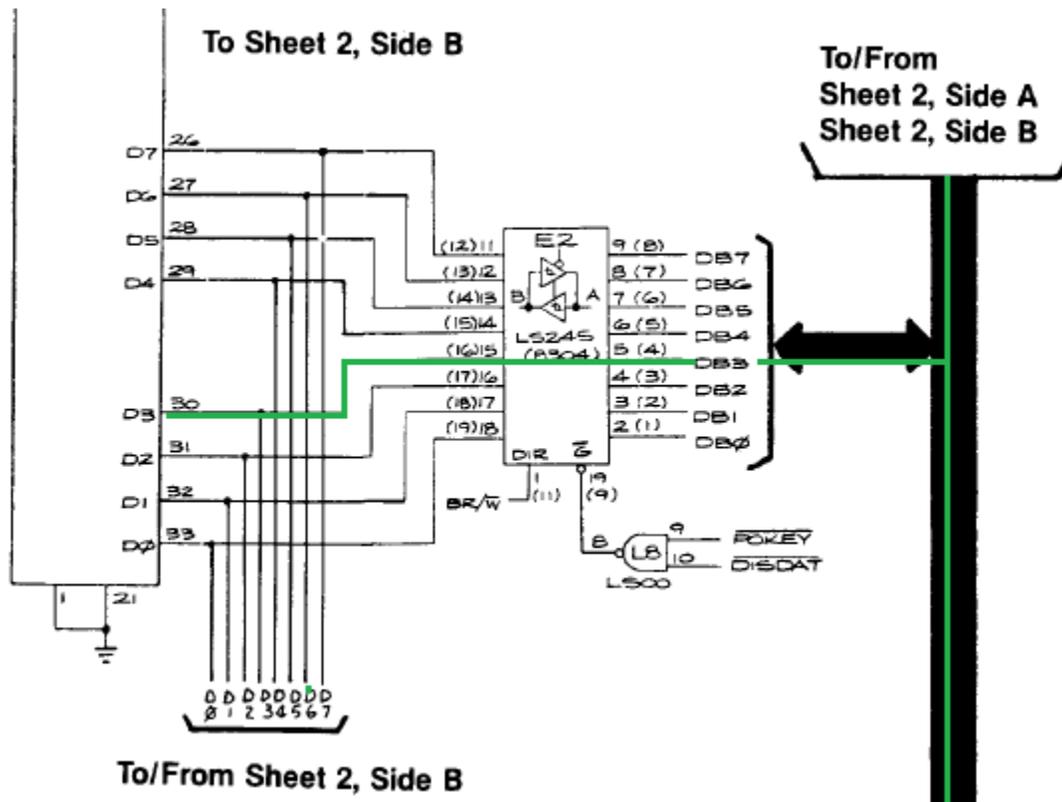
In the case of Centipede you know that address 0x0000 corresponds to both 2114s at H2 and F2 (each 2114 holds 4 bits of the 8 bits stored at that address). However what chip holds the data line associated with DB3? You can see from the schematics that it DB3 corresponds to pin 11 on the 2114 @ H2, so H2 is the chip that is responsible for data bus line 3 coming from the RAM.

RAM



If you get a BTS 08 error on a centipede this means that the error is occurring with data read/written to pin 11 on the 2114 RAM at H2. It's probably a safe bet to just pull the RAM and replace it, however keep in mind it might not be the RAM itself, perhaps there is a cut wire on the path from the CPU to pin 11 on

H2? or maybe there's a solder blob connecting this wire with something else. If you follow the path of D7 from the CPU to pin 11 on H2, you might also notice it passes through a 74LS245 at E2 the signal enters through pin 15 on the LS245 and then comes out the buffer on pin 5 heading to pin 11 of H2. Perhaps the LS245 @ E2 is bad. Tracing the circuit from the schematics and determining what chips could actually cause the RAM errors you see is an important skill. It is just like any other analytical thinking though, determine what systems might be the cause, then try to determine what chips are likely candidates and which are unlikely candidates. You can often create "tests" to rule out chips from being bad. For example you'll notice that E2 is not only used in the RAM circuits, but it is also on the path to the ROM circuits. If E2 is a problem it should also be messing up data being read from ROMs, so if you can do a ROM test and get back a good signature it's highly unlikely that E2 is the problem. (though not 100% impossible either :)



Note: D3 goes from the CPU to pin 15 on the 74LS245 @ E2, then comes back out pin 5 from the 74LS245 @ E2 before going to the 2114 RAMs as well as other locations such as the ROM circuitry.

Besides the LS245 @ E2 there are other chips that connect to DB3. For example the ROM chips connect to DB3, the video RAM chips also connect to DB3, the DIP switch circuits also connect to DB3 etc. Any of these circuits could also be affecting the test if they are not behaving. That said, always look for the simplest solution first. If you narrowed the problem down to bad data on DB3 during RAM tests, go ahead and replace the RAM in question (2114 @ H2). if you find the problem still occurs after replacing the RAM, then look for other places where a bad chip might be causing "interference" on DB3 making it look like a RAM problem.

Understanding Fluke RAM errors the “DCD” error

The other common error is the DCD error. The DCD error is given when the Fluke detects that there is an error with the addressing circuitry such that there are two addresses that seem to write to the same place.

For example imagine I have a RAM with addresses 0 – 1023. Now imagine I write zeros to all the addresses.). Now imagine I start writing the number 0xFF to all addresses. Let’s say I first write 0xFF to address 0x0000. What the Fluke will do in the RAM tests is before it writes 0xFF to the next address (address 1) it will first READ back the contents of address 1 to make sure it still is 0x00. If address 1 is NOT zero and it is instead 0xFF the Fluke would realize that writing to address 0 actually caused a change in address 1, too. This is usually caused by a failure in the address decoding circuitry directly on the RAM chips, and replacing the RAM that corresponds to the "bad address" is usually the fix. However it is also possible (but less likely) that the error could be in the address decoder circuit on the game PCB board. However this condition is less likely.

When the Fluke detects a DCD error it will provide an error message like

RAM DCD ERR @ address BIT x- LOOP

The *address* is the address that the Fluke tried to write to and the *X describes* and address bit that when 0 or 1 it acts the same (ie there are two addresses differing by one bit that "map" to the same storage location.)

For example in our example above the fluke might give us a

RAM DCD ERR @ 0000 BIT 0-LOOP

TO understand this error we first have to turn the address into binary

0x0000 = 0 0 0 0 0 0 0 0

Then we have to highlight the “BIT” that it detected is bad, in this case bit “0”

0x0000 = 0 0 0 0 0 0 0 **0**

What this means is when the Fluke writes to address 0000 000**0** (0x00) OR 0000 000**1** (0x01) it is writing to the same place. Another example is taken directly from the Fluke 9010 operators manual.

If you received the error

RAM DCD ERR @ 0FE0 BIT 4-LOOP

it means there the Fluke detected writing to 0x0FE0 OR 0x0FF0 wrote to the same exact place. (That is if I wrote the number 14 to 0x0FE0 and then read @ 0x0FF0 it would also read 14.)

0x0FE0 = 0000 1111 111**0** 0000

0x0FF0 = 0000 1111 111**1** 0000

This error USUALLY is in the RAM itself, so the easiest way to fix this error is to find the RAM chip that should hold the address the Fluke highlighted (such as 0x0FE0) and replace that chip(s). Due to the way some CPU board use multiple RAM chips to make up any given address this however is not as straight forward as it may seem. For example 2114 RAM chips each hold only 4 bits, so for any address there are 2 2114 chips, each holds 4 bit of the 8 total bits of storage. There are techniques to try to determine which chip is bad, but for a beginner it's often just easier to replace all the chips that correspond to that address.

Special thanks to KLOV users TROXEL and BARITONOMARCHETTO for reading through this guide, doing the exercises, catching many many typos, and generally making this a better document.