How Atari Signature Analysis Works

Written by Phillip Eaton - inbox@phillipeaton.com

Comments and feedback welcome & appreciated

Revision 1.1 – 2001-09-20

- **Updated contact links**
- **Minor text changes**

Revision 1.0 - 1999-12-26

- **Initial Release**

Introduction

Ever wondered what all those 4-digit 'signatures' are, that are dotted around the Atari circuit diagrams? They are used for diagnosing game PCB faults with an Atari CAT Box. They are useful because, armed with a CAT Box and a game PCB and **no technical knowledge** (so Atari say) you can diagnose most PCB faults down to IC level. Atari have issued signature analysis information for their game PCBs since Battlezone.

How is a signature created? If you send a known stream of data through a game board starting from the processor address bus, the data will ripple through the circuits in a repeatable pattern. By testing for that pattern at various locations against the known patterns – the Atari-provided signatures – you can isolate component faults.

So what's a CAT Box, then? A CAT Box is a Computer Assisted Troubleshooter, which is in a box. This was manufactured and sold by Atari from 1981 See Figure 1. To use the CAT Box, you remove the game PCB processor, and plug a connector from the CAT Box onto the game, using the ribbon connector and plug. On the older Atari PCBs, e.g. Missile Command, Centipede, the plug connects to a 50-way fine-pitch edge connector. On newer games, e.g. those using a Z80 such as Dig Dug, there is a convertor 'Pod' which is also required, which plugs into the processsor socket – see later.

Figure 1 Atari CAT Box Control Panel

Basically, the CAT Box has it's own processor which then can control the game PCB address & data bus, using the PCB's clock signal. It can then be used to address individual areas of the board for checking individual components, using the knobs and switches on the CAT Box control panel.

So now you know what a CAT Box is, you have a problem – you don't have one. So why should you bother reading any further? Well, you might want to read on because you're interested, but there are other reasons. Although the CAT Box is a 6502 processor-based piece of kit, the signature analysis bit is almost totally separate from the processor, and you could implement it quite easily with some simple discrete 74-series logic counters and buffers, and some seven-segment LED's.

As well as faultfinding PCBs, signature analysis is especially useful for checking out the integrity of ROM images for games – all you need is eight 4-hexdigit signatures, and the image to test.

The ROM-IDENTifier program written by Lescot Thierry uses this information in reverse. By using a 32-bit signature created from a ROM, it then looks up the signature in its database of known signatures and thus can identify which ROM you have just tested. (However, if your image is corrupt, it will not be identifiable.)

The easy and absolute way to check the integrity of an image is to do a binary compare of the sources using Windiff.EXE or MS-DOS FC.EXE, but you might not have a known-good image...

Signature Analysis Theory

Signature analysis relies on the same mathematical theory as Cyclic Redundancy Checking (CRC). A signature is pretty much the same thing as a Cyclic Redundancy Checksum. In my experience, this is used in communications systems to check for integrity of received data at the end of a communication link.

The general idea is to feed the data you want to send through a CRC creator as you are sending it, and then append the calculated CRC to the end of the message. Then, as the receiver reads the data, it also runs it through the CRC creator. If the calculated CRC and received CRCs are the same, the data has transferred correctly.

The CRC creator can be implemented in software, as is usual in data communications, or hardware, as in the case of the CAT Box.

CAT Box Hardware

Address Counter

Firstly, there is a 16-bit counter which is connected to the address bus and, when enabled by the CAT Box processor, continually cycles around the address space from $0000 to $FFFF. This is shown in Figure 2.

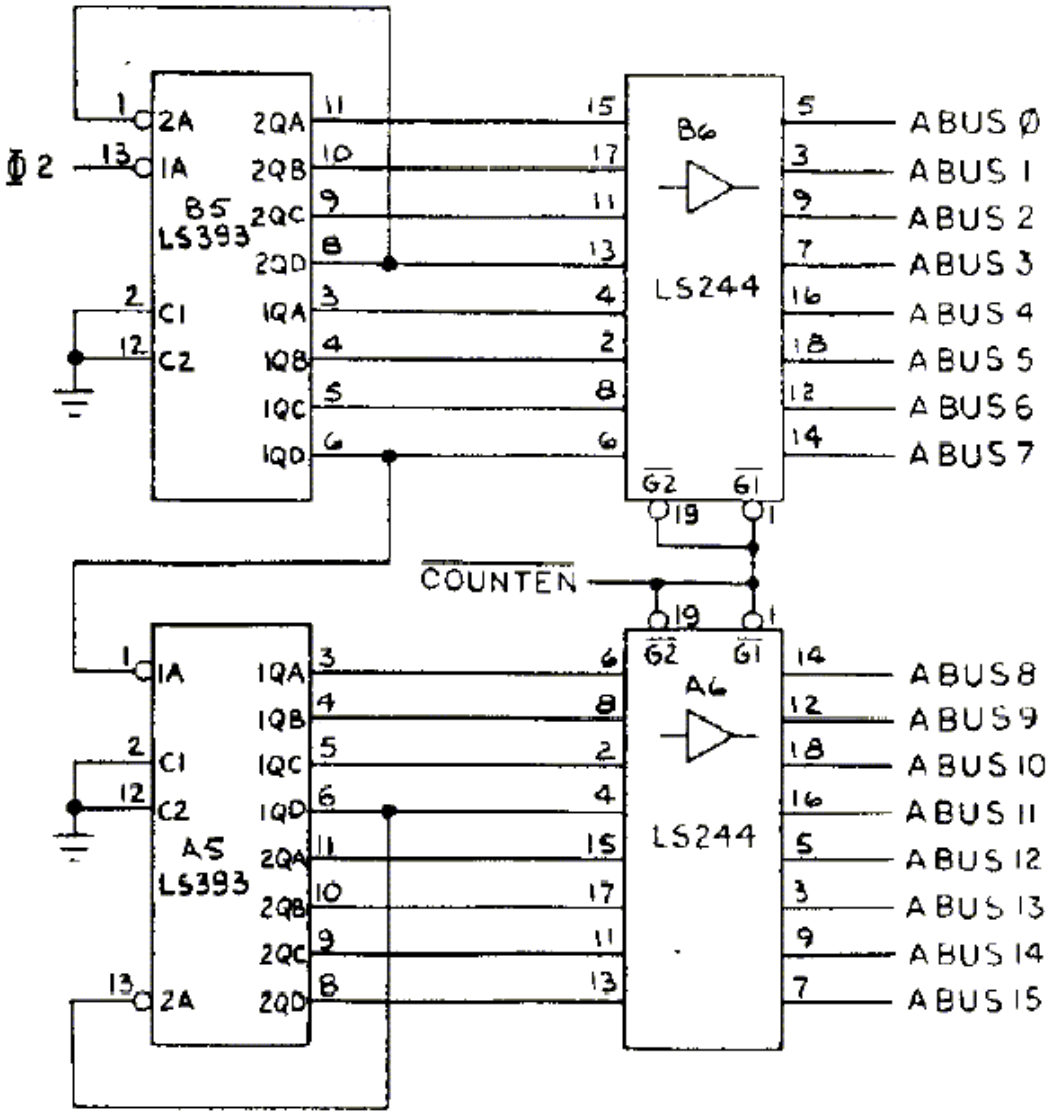

Figure 2 CAT Box Counter Circuitry

This is implemented by four cascaded 4-bit counters on two 74LS393 ICs. The 74LS244 ICs are used as buffers onto the address bus. (Note there appears to be a bug in the Atari diagram – the second counter on IC B5 appears to be cascading itself! If you understand the diagram, you should be able to spot the easy fix.)

Why is discrete hardware used to cycle round the address bus, when the CAT Box has its own 6502 processor, and can address all of the PCBs address space anyway? This is due to the way that the signature checking works - see further on.

Start, Stop, Clock and Data Probes

The next part of the circuit is the Start, Stop, Clock and Data probe inputs, shown in Figure 3 and Figure 4.

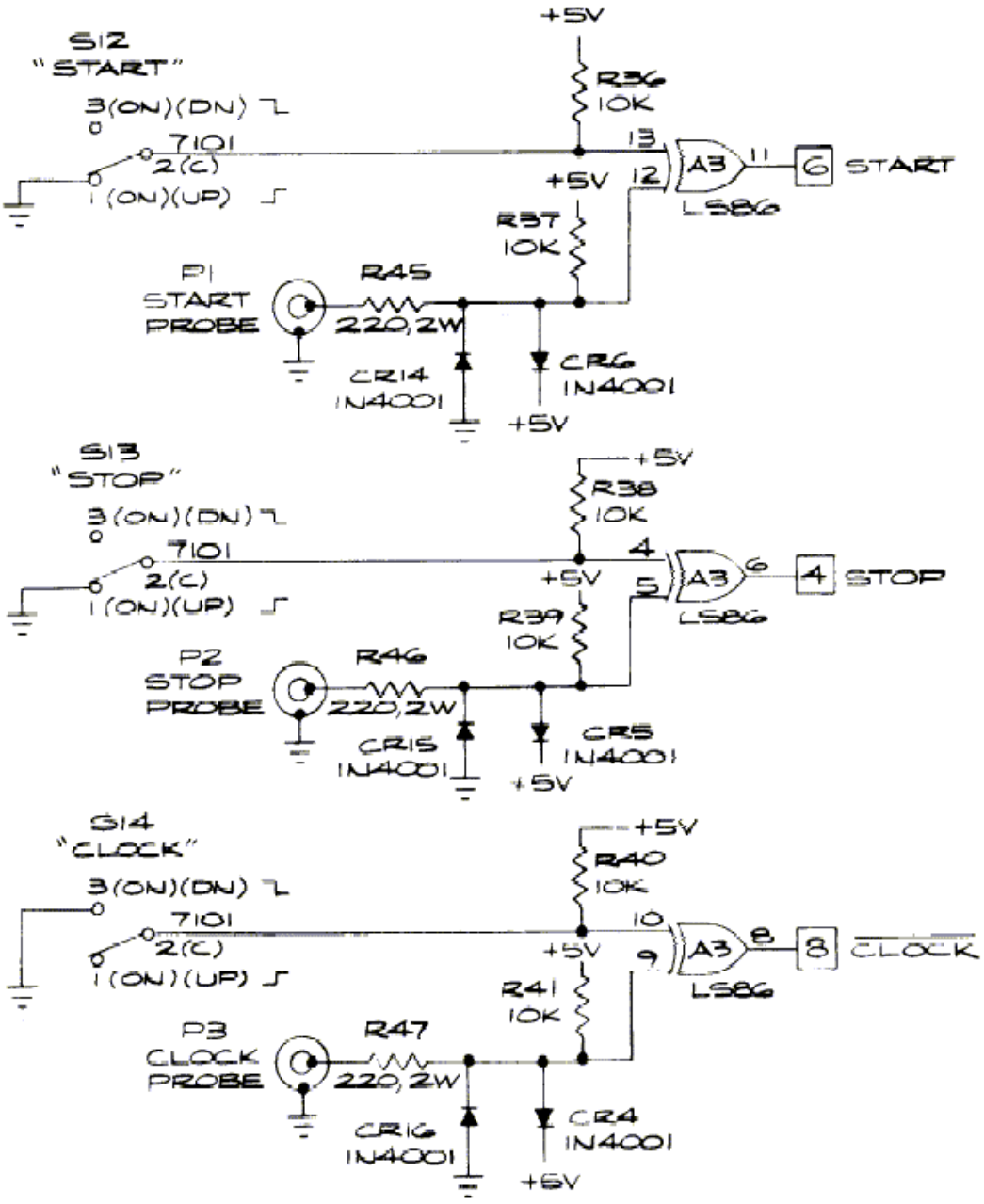

Figure 3 Start, Stop and Clock Probes

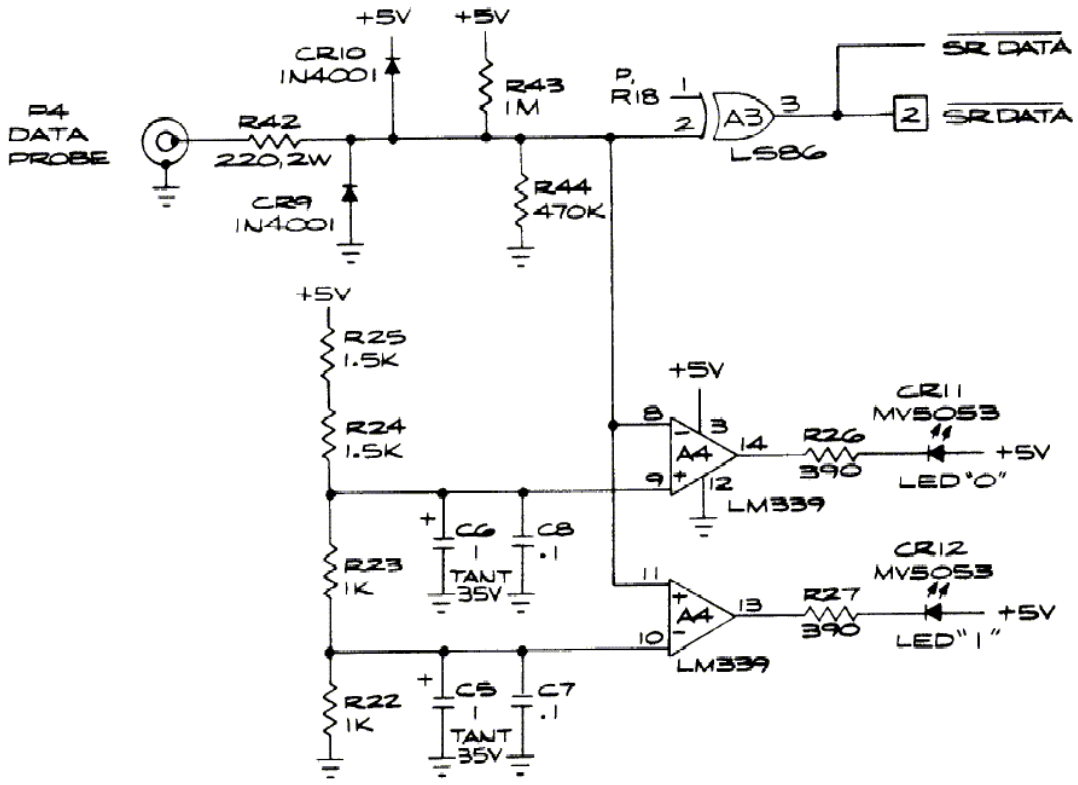

Figure 4 Data Probe

Consider the case where you want to check the signature of a ROM that is still plugged into the PCB. This IC will occupy a certain position in the address space of the PCB. When the 16-bit address counter cycles through the address space, for a period, the ROM will be enabled and then disabled. This will happen once per cycle of the counter.

When testing the signature, you would connect:

- Start and Stop probes to the Chip Enable line of the ROM.
- Clock probe to the clock signal used to clock the data in and out of the ROM.
- Data probe to one of the eight data lines of the ROM. (The Atari signature analysis only works on 1 bit stream at a time, so eight are be required for a complete ROM.)

Note that the Data probe is also used as a logic probe by the CAT Box hardware through the LM339 comparitors and the ‘0’ and ‘1’ LED’s.

Using D-Type latches, the Start and Stop probes control the Run/Stop signal that engages the signature analysis calculation hardware.

We can see now why the processor cannot be used to address each location and and test individually. Using the processor would involve using the address and data lines for the instruction fetch & execute cycle, which would reset the Start and Stop part of the circuit, and thus would confuse the signature.

Signature Analysis Calculation Hardware

Here, the 74LS193 counter is used to clock each piece of data collected by the Data probe (the SRDATA signal) into the 74LS164 cascaded shift register signature ‘memory’.

Whilst the ROM under test is enabled the Start and Stop probes allow the LS193 counter to be clocked by the Clock probe. This circuitry is shown in Figure 5.

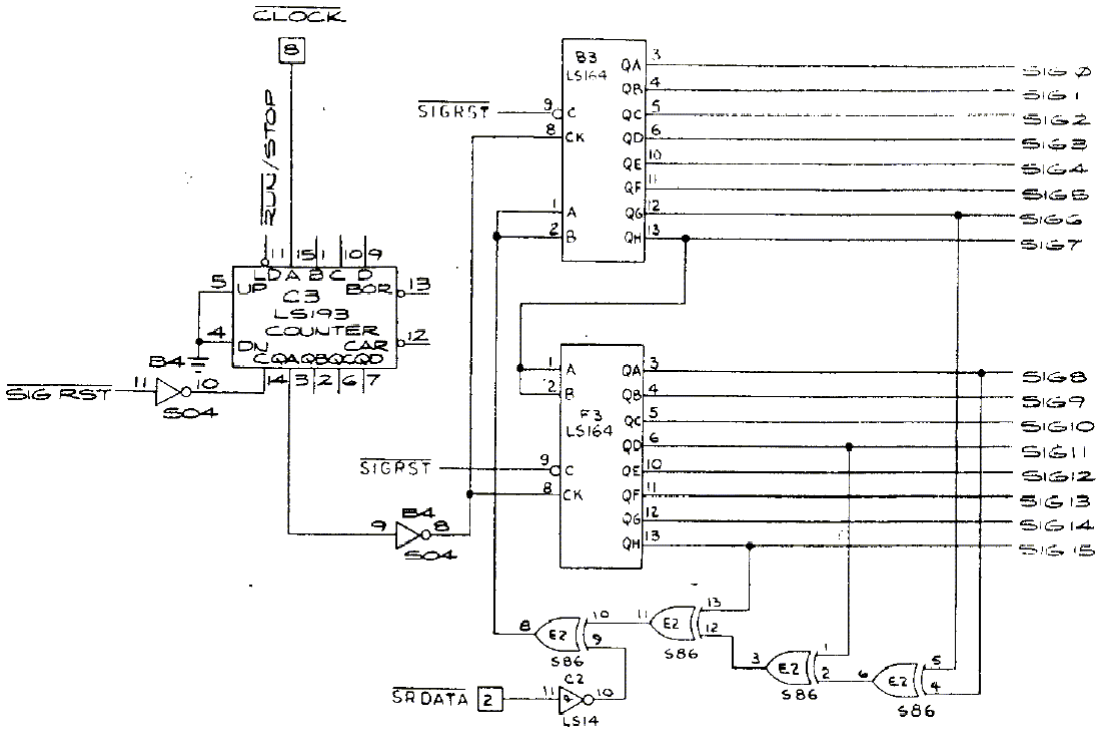

Figure 5 Signature Loop

The data held in the shift register(s) is fed back through XOR gates together with the SRDATA signal to create a new piece of data that is then shifted in to form a new signature.

At the end of each cycle (I guess) the processor reads the SIG0 to SIG15 lines which form a 16-bit (four hex digit) signature, and updates the seven-segment display on the CAT Box control panel. Then, the SIG RST line would be used to reset the LS193 counter.

Signature Analysis Emulator

To prove my understanding of this theory, I knocked up a little program that emulates the Signature Analysis bit of the CAT Box circuit.

OK, I know it's not exactly MAME, but to my surprise, when I wrote it, it worked first time! (If you've ever written a CRC checking program, you'll know that they're difficult to test, because until you've got it right, the results are completely wrong, and offer no clues as to why.)

I used QBasic because:

1. Everyone has a copy of it somewhere so I can provide the source code and you can run it and muck about with it if you want. (Credit my original though, please.)
2. It's quite easy to understand.
3. I wrote the program in the middle of the desert (literally) and it's all I had with me at the time.

This is shown in Table 1 Signature Analysis Emulation Program. Copy the code to a file, call it ATARICRC.BAS or something, and open it with QBasic and run it with Shift and F5.

Table 1 Signature Analysis Emulation Program

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Atari Signal Analysis for ROM Images Emulator Program
'
' Written using Microsoft MS-DOS QBasic Version 1.1
'
' Written by Phillip Eaton 1999-12-18
'            mailto:phil@pjeaton.demon.co.uk
'            http://www.pjeaton.demon.co.uk
'            *Comments and feedback welcome & appreciated*
'
' Program Description:
'   Uses software emulation of Cyclic Redundancy Check method as implemented
'   in Atari CATBOX hardware for checking integrity of game ROM images.
'
' How to run:
'   From DOS, type QBASIC /RUN ATARICRC and press Return.
'   Whilst in QBasic, open this file and press Shift+F5.
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' Version 1.0
'   Initial Release
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

' Declare Functions
DECLARE FUNCTION CRCBit& (Crc&, NewBit%)
DECLARE FUNCTION AtariHEX$ (Crc$)

' Declare Global Constants
CONST FALSE = &H0
CONST TRUE = NOT FALSE
CONST FILENUMBER% = 1

' Main program starts here
CLS

' Get filename and bit to test from user
INPUT "Filename of ROM image <centiped.210>"; Filename$
IF Filename$ = "" THEN Filename$ = "centiped.210"
INPUT "Bit number (0 to 7) <0>"; BitNumber%

' Open the ROM image. 'Filenumber' is used as a handle to the file.
OPEN Filename$ FOR BINARY ACCESS READ AS FILENUMBER%

PRINT "File Length = $"; HEX$(LOF(FILENUMBER%)); " Bytes"
PRINT
PRINT "Press a key to perform Signature Analysis": WHILE INKEY$ = "": WEND

' Loop for each of the data bytes in the file
'   i.e. at each address of the ROM image
FOR i% = 0 TO LOF(FILENUMBER%) - 1

' Read the next character in the file
DataByteChar$ = INPUT$(1, FILENUMBER%)

' Convert character to an integer number
DataByte% = ASC(DataByteChar$)

' Signal analysis only checks 1 bit at a time, so mask out the others
'  and convert to a true or false
DataBit% = (2 ^ BitNumber% AND DataByte%) <> 0

' Create a new CRC from old CRC and new data bit
Crc& = CRCBit&(Crc&, DataBit%)

PRINT "Address = $"; HEX$(i%);
PRINT "  DataByte = "; HEX$(DataByte%);
PRINT "  DataBit = "; DataBit%;
PRINT "  New CRC = "; HEX$(Crc&)
NEXT i%

CLOSE FILENUMBER%

PRINT "Signature = "; AtariHEX$(HEX$(Crc&));
END
```

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' AtariHEX$
' Description:  Recursive procedure to convert an ASCII representation of a
'               hex number to Atari Sig Analysis format, where B,C,D,E,F are
'               replaced by C,F,H,P,U.
' Parameters:   Crc$ - ASCII representation of a hex number
' Returns:      Crc$ with [BCDEF] characters changed to [CFHPU]
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
FUNCTION AtariHEX$ (Crc$)

' Operation of this function is an exercise for the reader!
IF Crc$ = "" THEN
AtariHEX$ = Crc$
ELSE
SELECT CASE LEFT$(Crc$, 1)
CASE IS = "B"
AtariHEX$ = "C" + AtariHEX$(RIGHT$(Crc$, LEN(Crc$) - 1))
CASE IS = "C"
AtariHEX$ = "F" + AtariHEX$(RIGHT$(Crc$, LEN(Crc$) - 1))
CASE IS = "D"
AtariHEX$ = "H" + AtariHEX$(RIGHT$(Crc$, LEN(Crc$) - 1))
CASE IS = "E"
AtariHEX$ = "P" + AtariHEX$(RIGHT$(Crc$, LEN(Crc$) - 1))
CASE IS = "F"
AtariHEX$ = "U" + AtariHEX$(RIGHT$(Crc$, LEN(Crc$) - 1))
CASE ELSE
AtariHEX$ = LEFT$(Crc$, 1) + AtariHEX$(RIGHT$(Crc$, LEN(Crc$) - 1))
END SELECT
END IF

END FUNCTION

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' CRCBit&
' Description:  Update given CRC & Bit using x15 + x11 + x8 + x6 polynomial
'               Uses LONG, not INT because of left shift (see code)
' Parameters:   CRC&    - Initial 16-bit CRC
'               NewBit% - New data bit for updating CRC with
' Returns:      Updated CRC
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
FUNCTION CRCBit& (Crc&, NewBit%)

' Get the bits out of the initial CRC and convert to TRUE/FALSE
Bit6% = (Crc& AND &H40) <> 0
Bit8% = (Crc& AND &H100) <> 0
Bit11% = (Crc& AND &H800) <> 0
Bit15% = (Crc& AND &H8000) <> 0

' Left shift the original CRC
Crc& = ((Crc& AND &H7FFF) * 2)

' Add feedback into left-shifted CRC
IF (NewBit% XOR (Bit15% XOR (Bit11% XOR (Bit8% XOR Bit6%)))) THEN
Crc& = Crc& + 1
END IF

' Return the new CRC
CRCBit& = Crc&

END FUNCTION
```

So you can test it, Figure 6 shows the codes used for Atari Centipede ROMS (These are extracts from the Signature Analysis Guide to Centipede © 1981 Atari, Inc. document reference TM-192.) I used CENTIPED.210 ROM image for all my tests, and the results show it to be ROM3 (IC J1), as shown in the Figure. (You will need to use your own images.)

4. Checking ROM and Data Lines

A. CAT Box Settings for ROM0 Test (I.C. D1)

Probe	Trigger	IC-Pin	Test Pt.
Start		D1-20	$\overline{\text{ROM0}}$
Stop		D1-20	$\overline{\text{ROM0}}$
Clock		C2-39	Φ2

To obtain stable signatures from ROM0, it may be necessary to install a 1000 pf capacitor from K3-11 to ground.

B. Signatures

Logic Probe on IC-Pin	Signal Name	Signature Should Be
D1-9	DB0	5AF2
D1-10	DB1	3276
D1-11	DB2	48UH
D1-13	DB3	P316
D1-14	DB4	PF7A
D1-15	DB5	H973
D1-16	DB6	3F34
D1-17	DB7	U638

C. CAT Box Settings for ROM1 Test (I.C. E1)

Probe	Trigger	IC-Pin	Test Pt.
Start		E1-20	$\overline{\text{ROM1}}$
Stop		E1-20	$\overline{\text{ROM1}}$
Clock		C2-39	Φ2

To obtain stable signatures from *ROM1*, it may be necessary to install a 1000 pf capacitor from K3-11 to ground.

D. Signatures

Logic Probe on IC-Pin	Signal Name	Signature Should Be
E1-9	DB0	13PH
E1-10	DB1	C4P5
E1-11	DB2	11F3
E1-13	DB3	098P
E1-14	DB4	5H24
E1-15	DB5	0548
E1-16	DB6	33P7
E1-17	DB7	80AA

E. CAT Box Settings for Address Decoder Test (I.C. F/H1)

Probe	Trigger	IC-Pin	Test Pt.
Start		F/H1-20	$\overline{\text{ROM2}}$
Stop		F/H1-20	$\overline{\text{ROM2}}$
Clock		C2-39	Φ2

F. Signatures

Logic Probe on IC-Pin	Signal Name	Signature Should Be
F/H1-9	DB0	CU62
F/H1-10	DB1	9553
F/H1-11	DB2	7756
F/H1-13	DB3	A7CF
F/H1-14	DB4	6081
F/H1-15	DB5	5HAC
F/H1-16	DB6	6U43
F/H1-17	DB7	F83H

G. CAT Box Settings for ROM3 Test (I.C. J1)

Probe	Trigger	IC-Pin	Test Pt.
Start		J1-20	$\overline{\text{ROM3}}$
Stop		J1-20	$\overline{\text{ROM3}}$
Clock		C2-39	Φ2

H. Signatures

Logic Probe on IC-Pin	Signal Name	Signature Should Be
J1-9	DB0	476H
J1-10	DB1	2A2C
J1-11	DB2	2337
J1-13	DB3	FP07
J1-14	DB4	A9AF
J1-15	DB5	12HA
J1-16	DB6	2367
J1-17	DB7	8P82

Figure 6 Centipede Signatures

Most of the circuit diagrams for Atari Games have signatures marked on them, but these are usually just logic circuitry signatures, not for the ROM's. I guess this is so the circuit diagrams were not software dependent, which allowed them to modify the software without invalidating the circuit diagrams that were already released. You'll need to get your hands on the actual Signature Analysis Guides (or at least the signatures) to check your test results.

Atari Signature Conversion Characters

Note that the signatures have characters in them that don't look very hexadecimal. Don't worry – these are simply substituted for different characters for display by the CAT Box.

See the emulator code for the conversion – it's very simple. I don't know why these characters are used instead of straight hex digits, but other manufacturers of test equipment of the time did likewise, so it's not just an Atari thing!

Other Processors and Different Manufacturers Boards

The Atari CAT Box and the signature analysis methods could be used on any manufacturer's hardware using the same processor – Atari made it simple for owners of their games by adding a dedicated connection point on the side of their PCBs and providing the signatures from known-working boards.

Atari also produced some ‘pods’ which allowed the CAT Box’s 6502 to control Z80 and other processor-based hardware. These were simple passive devices that were used to modify the read/write control line format. This differs slightly between the different types of processor used at the time.

My Dig Dug board, which has 3 Z80 processors, doesn’t even have the 50-way connector, so the CAT Box needs to be connected through the processor socket.

If you wanted to use it on a non-Atari PCB, you would need to make a connector that would attach your external circuitry in place of the processor in the same way as the Dig Dug PCB. (You need to remove the processor anyway when using the CAT Box.)

Further Development

The CAT Box was designed when a 6502 was Hot Stuff, no one had a PC and RAM cost a bomb. Through new technology, there is at least one development underway which will re-create and improve the functionality of the CAT Box. Watch this space…

Links

Checkout the Atari CAT Box User Guide and loads of other test equipment info at Spies:
http://www.spies.com/arcade/schematics/atari/catbox/

Get Lescot Thierry’s ROM-IDENTifier program:

http://www.system16.com/romident.htm