KUN001 - Kachina 505DSP Transceiver Backup Battery, Consequences and Cures

Written by Don L. Jackson, AE5K - don@ae5k.us
with assistance from Tom Mandell, W3FRG - w3frg@comcast.net

Revision History

Revision 1.00 - 2007 May 26 - initial release
Revision 1.01 - 2007 May 30 - revised freq ref calib info to include oven information, added the two tables to document.

The most recent revision may be found on our Kachina Wiki at http://kachina.ae5k.us under the Kachina User Notes. This document is expected to be revised periodically, so please check for latest revision before using the information.

Disclaimer

The information contained in this document is, to the best of our knowledge, correct and current as of May 30, 2007. It is supplied free as a public service to Kachina 505DSP owners. However, it is up to the reader/user to determine for themselves whether to apply any ideas or suggestions contained herein. The writers of this document are not responsible for any results, good or bad, from following the information or advice in this document. That responsibility remains solely with the reader. Therefore, if your Kachina 505DSP goes up in smoke, your car does not start, you develop a rash or your wife leaves you as a result of applying any information in this document, it is your fault, not ours!

Brief Explanation of Software vs. Firmware

This may not be news to many, but to bring everyone up to speed on this, we wanted to explain a few things first. The Kachina 505DSP (often called just the 505 in this document) has a Motorola MC68HC16Z1 microprocessor chip as its main "brains". This chip does not have any program memory (code, instructions) inside of it, but depends on an external ROM (read-only memory) for its program instructions (what "to do"). The usual ROM for the 505 is actually two EPROM chips found in sockets on the PC401 board. If you have firmware version 4.29 or 4.49 or whatever, this is where it physically resides. An alternate form of ROM was provided in the 505 as a plug in PCMCIA type card containing some form of ROM. Two slots are provided on the PC401 board for this purpose.

The microprocessor also needs R/W (read/write) type of memory to store data and this memory is called SRAM. In this case, it is contained within the MC68HC16Z1 chip itself. When power goes off, ROM memory keeps its data but SRAM memory is volatile and loses its data. Since there are certain pieces of data that vary from

system to system that are necessary to retain from power on/off session to session, the SRAM is "backed up" or continued to be powered separately by a small battery. Fortunately, the MC68HC16Z1 chip has provision to easily do this -- that is, power only the SRAM portion by a battery.

Besides the firmware (program instructions) in EPROM or ROM, and the data held in SRAM, we also need software that runs on a PC to control the 505. This software is called here "control program" and examples are KC505.EXE that came with your transceiver, and the newly written KachinaCAT by W1HKJ.

Communication between the control program operating on your PC and the 505 is accomplished using the PC serial port (COM1 for instance) and RS232 type 9600 baud serial communication. Kachina has documented the communication protocol, but in some cases it is not clear as to meaning or method.

The Kachina 505DSP Battery Situation

We know the SRAM inside the MC68HC16Z1 MPU chip holds certain data necessary for the proper operation of the 505. At least some of this data was entered during the Kachina production line testing while other data may be stored and/or changed during the operation of the 505.

The SRAM is volatile memory, but the data is retained by use of a 3 volt battery on the PC401 card module during the time when the 505 is not powered. Battery life is quite long, most likely approaching 10 years on the average. However, this battery must eventually be replaced and doing so in such a way that interrupts the voltage necessary for retaining SRAM data, the data is lost. By lost we mean that the bytes of data contained in the SRAM may be any value (all zeros, all ones, random patterns, etc.) and not the intended legitimate values.

It is possible for the data in the SRAM memory to become corrupted by means other than battery interruption.

There are three methods of restoring the SRAM data. The first, and presently preferred method, would be to be able to read and store the data in a file before any corruption or possible loss from changing the battery, then be able to restore (write back) this same data to the SRAM after the battery is replaced.

The second method of restoration would be necessary if a copy of the data was not available, as in the case of accidental loss before a copy could be made. This would include cases of mysterious corruption as well as where the user unknowingly removes the battery.

The third method is presently just a proposal and will be mentioned later in this document. It has not yet been implemented or tried, but holds great promise.

SRAM Data

In this section, mention is made of two programs: LAST2.EXE and KCCAL.EXE. More about these in a later section, but these Kachina Communications programs can be used to partially restore corrupted or lost data.

We think saved data in SRAM contains at least the following items:

1. Serial number
2. S-meter calibration table (16 bytes)
3. Carrier balance (2 bytes)
4. Phase detector table (16 bytes)
5. Frequency reference calibration table (32 bytes)
6. Antenna impedance data (various number of bytes)
7. "On" time hour meter, fault record, DVM data, etc.
8. Certain data to allow transceiver to return to frequency and mode at power up.

The **serial number** (S/N) will be scrambled if data corruption has occurred. The S/N is not vital to the proper operation of the 505. The correct S/N can be restored using KCCAL.EXE. From our observation, all 505 serial numbers start with 91900 and followed by three more digits.

S-meter calibration can be performed and new data entered. The S-meter will act erratically if the calibration table consisting of 16 values (points of calibration) is random due to data corruption. LAST2.EXE will merely zero out the calibration table and allow a sensible meter action, albeit not exactly correct. (One may argue that for most uses of the transceiver, accuracy is not important, only relative signal indications are.) LAST2.EXE and KCCAL.EXE allow actual calibration entry values to be stored once again in SRAM. It appears that each byte in the table is a signed value to adjust the s-meter + or - at each calibration point spaced 10 dB apart.

Carrier balance consists of two bytes and data from over a dozen 505 owners show that it varies from transceiver to transceiver. As far as we can determine, KCCAL.EXE will restore it. This still needs to be explored more.

Phase detector table still remains a complete mystery to us. Neither program touches it and they are different for each 505. It appears that the 505 may not use this data - but we don't know.

The **frequency reference calibration** table contains 32 bytes and nearly all 505's have exactly the same table. The exception are the 505's equipped with the special option high stability oscillator (oven), often used for CAP operation. If corrupted, this table may be restored by use of LAST2.EXE to the proper table provided that the question about the high stability oven being installed is answered correctly. We are not absolutely sure at this time what the values mean, but plotted on a graph give a nice "s-curve" similar to those often seen for crystals frequency vs. temperature. The table for the oven option plots to nearly straight line.

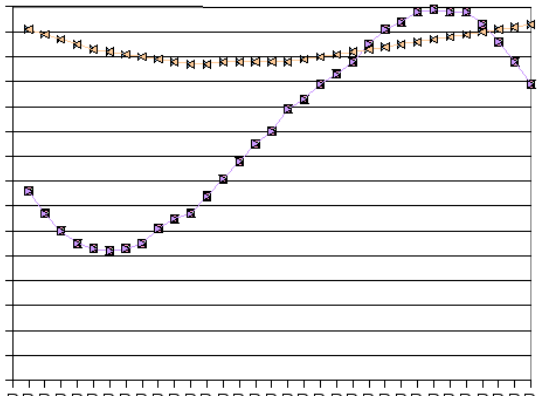
This is the non-oven table:

4c	43	3c	37	35	34	35	37
3d	41	43	4a	51	58	5f	64
6d	71	77	7b	80	87	8d	90
94	95	94	94	8f	88	80	77

This is the oven equipped table:

8d	8b	89	87	85	84	83	82
81	80	7f	7f	80	80	80	80
80	81	82	83	84	85	86	87
88	89	8a	8b	8c	8d	8e	8f

The two tables plotted on a graph appear this way:



Antenna Impedance data appears to be that data just saved from using the antenna tune function of the 505. It can easily be cleared using either of the two control programs for the Kachina 505DSP (KC505.EXE or KachinaCAT by W1HKJ), so there is no worry about losing anything here with battery loss. If SRAM gets corrupted, the user may see wild entries in this table with frequencies well outside the bounds of the 505 (including negative frequency entries!).

On-Time hour meter, Fault Record, Retrieve DVM data and possibly more are unexplored areas for us and do not seem to make a difference in restoring a 505 after data corruption. They are information functions.

Note: we should not confuse the data saved in the SRAM with data that is saved in the Kachina PC control program. The Kachina PC control program saves such things as the station call, time zones, frequency memory recall, CW keyer speed, and some other settings. These are primarily operator conveniences and preferences. If we delve further into this, we'll further refine the list of what is saved where.

SOFTWARE

Presently, from Kachina, we have two programs that run on a PC which may assist us in the second restoration case. They are KCCAL.EXE and LAST2.EXE (and the doctored version for COM1 named LAST1.EXE). All three can be found for downloading on the Kachina Wiki at http://kachina.ae5k.us but PLEASE read on before using them.

WARNING! Both LAST2.EXE (LAST1.EXE) and KCCAL.EXE can alter the data in your SRAM! Use very cautiously.

LAST2.EXE is a PC program that appears to work on most versions of MS Windows. From what we've learned, it was an attempt by Kachina's technicians to allow the user/customer to restore their 505 to operating condition after SRAM corruption. As mentioned previously, it leaves the s-meter calibration table at all zeros. Unless you need to, we would suggest you not run this program with your 505.

KCCAL.EXE is a PC program that does **NOT** work on newer MS Windows versions. It appears to require Win98 and is known not to work with Win2000 or WinXP. One of the major uses for this program is to restore the serial number, even if unimportant to the operation of the 505. This program was the one used at the factory but is not the program which produced the nice full page printout of various tests performed that you received with your 505 (or received if sent back to factory for repair). Unfortunately KCCAL.EXE is incomplete and undocumented. Since it has the potential of changing things you may not wish changed, we would suggest not running it unless you need to.

There is also a program from Kachina named KCSMETER.EXE. It appears to work only if the 505 in question has a PCMCIA card installed. It will ask this question during setup and will automatically quit if the answer to the question: "Are you using a ROM option card (Y/N)" and the reply is "N". Tom reports that "I have always been advised never to use this program by Kachina."

Making a Backup of SRAM Data

So, as you can see, we have limited options at this time for backing up and restoring corrupted SRAM.

To our rescue comes W1HKJ, Dave Freese, who not only writes some swell programs for digital modes, but also has written a control program for the Kachina 505DSP called KachinaCAT. Information may be found on his web pages at http://w1hkj.com including free downloads. His KachinaCAT works on both MS Windows and Linux. Dave has just recently become a 505 owner and has interest in expanding KachinaCAT for all of us. If you have a suggestion, email him.

The KachinaCAT program not only shows us the serial number, but also allows displaying and saving dumps of the various SRAM tables (he calls it NRAM). We

would suggest that you immediately use his program to dump and save your SRAM data before your battery goes dead. (Hint: find it under "Util" menu) And after you do that, please send a copy to don@ae5k.us giving your serial number and any known history of SRAM corruption/restoration -- that data as accumulated will help us in our further research!

It appears at the present time, that at least the first method mentioned above (reading data to file and restoring) might be best handled by an addition to W1HKJ's "KachinaCAT" program, or alternately a separate utility program to perform the specific save/restore operation. We favor a separate utility due to nature of program which, improperly done, *might* cause a 505 to become inoperable. This then frees KachinaCAT of any stigma of possibly causing corruption, does away with "feature bloat", and allows the utility to carry all kinds of warnings to the user.

Restoring SRAM Backup Data

At present time, there is no nice automatic reverse of the SRAM dump, so the user must resort to using LAST2.EXE and KCCAL.EXE to manually restore things. At present time, we are not absolutely sure about the meaning of some values and how they are determined for proper 505 operation for some of the saved parameters. Tom is presently exploring how to present the data for LAST2.EXE and KCCAL.EXE to accept it and we will update this documentation when good procedures are worked out.

Restoring the Corrupted 505DSP

This section outlines the procedure to follow to restore a Kachina 505DSP transceiver whose battery has been replaced without maintaining the backup voltage for SRAM data retention, or other corruption of the data has occurred.

1. Run LAST2.EXE to restore the frequency reference calibration table. If you have a high stability option 505, you will need a different table. Email AE5K or W3FRG for this information -- it will eventually be posted when time and experience permits.

2. Run LAST2.EXE to zero out the s-meter calibration table. Either LAST2 or KCCAL will allow entry of new values if you have the proper equipment to do the level calibrations. At least a zeroed table will allow reasonable s-meter operation.

3. Run KCCAL.EXE to restore the Carrier Balance value. Most likely the value will have hex FF as its first byte from what we've seen.

4. Run KCCAL.EXE to enter and restore the proper serial number.

Battery Replacement Procedure

If your 505 is still operating properly and you wish to replace the battery *before* it

no longer allows data retention, we highly suggest you first follow the procedure under the "Making a Backup of SRAM Data" above before even thinking of battery replacement!

Personal note from AE5K: I would leave well enough alone if my battery is still retaining data. This is the "if not broke, don't fix it" philosophy. We may have better solutions down the road saving you the trouble of replacing the battery by the time you would really need to.

There are probably several good methods of changing the battery. The method outlined in a companion document has been successfully performed, but it is not the only method. We would welcome directions and experiences from others who have tackled this task. Refer to the companion document "KUN002" for step by step instructions on battery replacement.

Tom, W3FRG, reports that the method he used in KUN002 did not utilize a "Keep alive Voltage" on the PCB during removal and installation of BT-401 (the battery). However, he did find that all four of the retrieved by KachinaCAT NRAM data files were corrupted as expected, but it did not kill the 505. By running "LAST2.EXE" both the Frequency Reference Calibration and S-Meter tables appeared to be restored to sane values. S-Meter calibration values were all set to zero.

Unknowns

I think we could write a book for this section. In fact, we don't even know enough about some things to ask intelligent questions! We're exploring, discovering, and learning.

For instance: There are probably some slight differences in hardware for productions of the PC401 module. Can we figure out what they are and do they have any impact on what we are trying to do here? There appears to be different hardware versions reported in the "Serial Number" lists maintained on the Kachina Wiki: B1, C1, C3, etc. Are these changes even significant for our purposes?

We feel like we are flying blind much of the time without our instruments working!

The Future

After putting together the information we know, it appears at this time the only practical method of accessing (reading and restoring) the SRAM data is via the serial port communication commands partially documented by Kachina. We suspect there could be additional commands, undocumented, but most likely only some disassembly of the EPROM firmware or obtaining the source code will reveal this.

An alternative to much of this might be found by doing EPROM firmware disassembly and additions made to the commands to allow easier dump and restore of SRAM contents. This disassembly might also lead to future

enhancements and expansion of 505 functionality if someone is interested in pursuing this path. Only about 50% of the EPROM firmware space is presently utilized. Of course, enhancements can be made only to the maximum extent that the current 505 hardware configuration will allow.

Let's Junk the Battery!

Why not just junk the battery that is backing up of SRAM? Eliminate the battery! This is part of a proposal by AE5K discussed by a couple of us working on this project. This is the third method alluded to at the beginning of this document.

If we have a method of easily saving and restoring the contents of SRAM via the serial port, then eliminate the battery back up of SRAM and merely download (restore) the SRAM data from a PC file each time the control program (KC505.EXE or KachinaCAT) is started. This method is often done in todays computers and peripherals, especially in the USB (serial bus, not upper sideband!) arena. Simple? Actually W1HKJ is doing some of this with KachinaCAT at the present time with saving and restoring 505 status information (in a file).

Actions Going On

At present time, AE5K is working on disassembling the EPROM firmware. It is estimated that disassembly will yield probably 10 to 12,000 lines of assembly source code, so it will be no easy task to make sense of what is happening in the firmware. If all goes well and he does not get discouraged too badly, we may have some useful information eventually. It will obviously take time!

Meanwhile, W3FRG is still working of cracking the information necessary for the LAST2.EXE and KCCAL.EXE programs. Once we figure out exactly how to send the necessary information to the 505 in order to restore tables, then W1HKJ might be able to write a utility to make it easy to save and restore everything necessary ... or even implement the "let's junk the battery" method.

Users can help by responding to calls for data, information about their 505, etc. Also, their experiences in battery replacement, etc. to fill in missing information from this document. Your words of encouragement are appreciated also.

It should be said that some other attempts are being made, but cannot be reported at this time due to their sensitive nature. If they bear fruit, we'll report them -- or even if all hope is lost, we will report that too.

We would like to see more Kachina 505DSP owners supporting our discussion list. The current number on the list is about 160 and this has remained fairly constant over the past several years. We can deduce from the serial numbers that probably a little over 600 Kachina 505DSP transceivers were manufactured. It could be that 3/4 of the owners either don't know about the discussion list or they are not interested in the day to day messages. To solve the latter, would an announcement-only (1-way) list be attractive to many who don't wish to see all

the message traffic, but still wish to be informed of news? (This would just supplement the present discussion list, not replace it.)

Appendix A - File Extensions

LAST2.EXE files currently used have the extension:

- *.IMP - Antenna Impedance
- *.SME - S-Meter
- *.REF - Freq Ref
- *.DET - Phase Detector
- *.BAL - Carrier Balance

Appendix B - PC Software

KC505.EXE - control program to operate the Kachina 505DSP under MS Windows. Version number used is sensitive to the firmware installed in the 505.

LAST2.EXE - a partial restore program, used to restore frequency reference calibration table and to zero out the s-meter calibration table. Requires use on serial port COM2.

LAST1.EXE - same as above only COM1.

KCCAL.EXE - a factory calibration program (see text) that runs on MS Windows 98.

KCSMETER.EXE - an s-meter program used with ROM PCMCIA card in early versions.

KachinaCAT - control program to operate the Kachina 505DSP under MS Windows or Linux. Written by W1HKJ and still under active development by him. Feedback welcome by him. Useful for making SRAM partial dump.

--EOF--