



**Speeding Up Your Console**  
**By Jesse C. Slicer**

**INTRODUCTION**

Does your stock TI-99/4A console seem to be dragging in these modern days of computers running on 33 MHz 80486 and 68040 systems? If so, perhaps a quickie speedup is for you. The following instructional will show you how your stock TI-99/4A can be speeded up from 3 MHz to 3.58 MHz. I accept NO responsibility in the damage of anyone's computer equipment; however, I have taken care to ensure success. I credit most of the technical material presented here to Barry Boone, who first told me how this modification was done. Make sure you read this ENTIRE document before you take any action whatsoever.

**BEFORE YOU START**

Before you begin dismantling your console, eager to speed it up, there is a part you may or may not need to buy. This is the 14.31818 crystal (this is NOT a clock crystal). I was once given two defunct Commodore VIC-20s and each of them had these for their video circuitry. Otherwise, it will be a trip to your local electronics store. Most Radio Shacks do not have this in stock but they can order it for you. It takes about four days, and the cost is about four and one-half dollars.

**GETTING STARTED**

With part in hand, and standard tools at your side, you are now ready to begin. Open the console all the way until you have the circuit board facing up at you. About one and one-half inches below the 9900 microprocessor and just to the right of the 9904 sound chip should be a component that looks almost like the one you just acquired. Carefully note the number on the one on the circuit board. If it is not 12.000 (might be 28.000), then this console cannot be modified in this manner.

**REMOVING THE OLD CRYSTAL**

Use your fingers to locate the solder pads for the crystal on the bottom of the circuit board. Flip the circuit board over. Using a desoldering iron, remove the solder pads surrounding the leads. The crystal can now be pulled out of its normal place and set aside. DO NOT THROW IT AWAY!!! You have a definite use for this that I shall describe later!

## INSTALLING THE NEW CRYSTAL

Face the numbers that are on the new crystal in the same direction the old one was, slide the new crystal into the area where the old one was located. Using a soldering iron, place small solder pads around the base of the leads on the circuit board. Then, using wire snippers, cut the leads down to your solder. Clean up the area on the circuit board, close up the console, and turn on your computer.

## THE MAGIC HAPPENS

Run a few programs and note the increase in speed they have. Enjoy how you gained 19.3% increase in pure microprocessor speed. Then, as you run some programs (terminal programs, graphics intensive, for example), you begin to notice.....

## PROBLEMS!

Ack! Why did there have to be a snake in paradise?!? All is not lost. You can still use your terminal programs and graphics intensive programs with your new console. Remember when you saved the 12.000 crystal? How about we put them both in and have a switch between them? Sounds like a good idea. Let's do it.

## INSTALLING THE DUAL SPEED CRYSTALS

Assuming you read through this entire document before you started, this will save you some trouble. First, obtain a double pole, single throw (DPST) switch from ye olde electronics shoppe. This should have six connections on the bottom of it. Also, obtain about ten inches of some thin duralloy wire. Cut the wire in half and solder one end of each of the wires onto the middle leads of the switch. Then, instead of installing the new crystal as shown under "installing the new crystal", install the other ends of the wire into the old crystal "socket". After that is complete, solder (remembering the way the numbers were facing in the socket (key them with the wires)) each crystal to the two paired leads of the switch. Mount the switch somewhere on your console. I cut a hole in the back and glued it there. You now can switch between a standard console for those problem programs and the new SPEEDY console that gets your work done somewhat faster!

## ENJOY!

Programs that do intensive number crunching or memory manipulation will benefit from this the most. Disk I/O will speed up slightly only because the code in the ROMs are being executed by the faster processor. Good luck and warp speed!

DIAGRAM is located on Page 9

*EDITORS NOTE: this article was downloaded from DELPHI  
ON 9-10-91.*

## Mike Ballmann's 32K -- 16 Bit Bus Project

The following is a step-by-step description of how to add 64K of RAM memory on the 16 bit bus. The present modification uses only 32K. This corresponds to the memory space of the 32K Memory Expansion. The modification yields a speed increase of about 50%.

Mike Ballmann is currently working on a circuit to allow CRU decoding of the remaining 32K. This will open up a whole new area of software, including such possibilities as a real DOS which could be loaded into RAM from disk on power-up. The 32K modification described below can easily be modified for full decoding upon completion of Mike's work.

You will need two Hitachi HM62256LP-12 RAMs. One source of these is Microprocessors Unlimited. They cost around \$13. You'll also need a 74LS21 and a 74LS153. These can be obtained from various electronics supply houses. All wiring should be done with wire-wrap wire. You should use a low wattage soldering iron with a fine, pencil type tip.

The modification is done on the main board of the Black & Silver console, and you'll need to refer to the Logic Board Component Location Diagram in the TI-99/4A Console Technical Data book.

- 1) Remove the board from the console, and identify the two ROMs. They are located between the GROM connector and the 9900 IC. One is parallel to the 9900 and the other is perpendicular to it. They are U610 and U611 on the Component Location Diagram.
- 2) Bend the pins on the HM62256 IC's closer so they will firmly contact the ROM pins when piggy-backed. One way of doing this is to place the RAM on it's side on a table and then move the body of the IC toward the table to bend the pins uniformly.
- 3) Bend out the following pins on both HM62256 RAMs: 1 2 20 22 23 26 27 28. These pins will NOT be soldered to anything on the ROMs. Holding the IC with the notch up and looking at the top, pin numbers start with pin 1 on the upper left, go down the left side, then across and up the right side. Pin 28 is opposite pin 1 on the end with the notch.
- 4) Place one HM62256 over the ROM that is parallel to the 9900. Make sure the notch points toward the 9900 and that the writing on the 9900 and the 62256 can be read from the same direction. Place the RAM such that pins 1 2 27 and 28 extend beyond the end of the ROM. The un-notched end of the

RAM should line up with the un-notched end of the ROM. There should be a sort of "spring tension" that clamps the RAM pins onto corresponding ROM pins below it. This will help to insure good solder joints. If the RAM doesn't fit tightly, remove it and bend the pins closer.

5) Solder all RAM pins not bent out to the ROM pins below. Use a low wattage soldering iron with a fine, pencil type tip. Inspect each solder joint carefully in good light, under magnification.

6) Place the second 62256 on the ROM that is perpendicular to the 9900. The notch on the RAM points away from the 9900 and toward the edge of the board. As above, solder and inspect all pins that were not bent out.

7) Bend out the 74LS21 pins 1 2 4 5 6 8 10 12 14. Note that pins 1 and 14 are across from each other on this 14 pin IC.

8) The 74LS21 will be piggy-backed on the 74LS138 U504. This IC is located adjacent to the end of the board where the edge connector is. There are two 138's next to each other. U504 is the one nearest the end of the board. You will place the 74LS21 so that the UN-NOTCHED end lines up with the un-notched end of the 138 (pointing toward the cassette connector). Pins 1 and 16 of the 138 will extend beyond the notched end of the 74LS21.

9) Before positioning the 74LS21, solder 1/2" lengths of wire-wrap wire to the 138 pins 7 and 9. Then position the 74LS21 on top of the 138 and solder all pins not bent out to the 138 pins below and inspect the connections.

10) Bend out all of the 74LS153 pins EXCEPT 8 and 16.

11) Place the 153 over U613, a 74LS194. The notch will line up with the 194 notch and point toward the edge of the board away from the 9900. Solder pins 8 and 16 of the 153 to pins 8 and 16 of the 194 below.

12) At the end of the 9900 opposite to where the RAM's have been piggy-backed, you will see a line of three ICs. They are a 74LS00, 74LS32, and 74LS04. The 74LS00 is U606 and the 74LS32 is U605. Turn the board upside down so you can see the traces. Find the trace that runs from pin 11 of the 74LS00 (U606) to pin 13 of the 74LS32 (U605). Double check to make sure you're doing the pin numbering correctly. When you've found the trace, cut it with a knife so there is no continuity between the LS00 pin 11 and the LS32 pin 13.

13) Identify the piggy-backed RAM that is perpendicular to the 9900. Solder wire-wrap wires connecting every bent out

(CONTINUED ON PAGE 6)

pin on this RAM to the corresponding bent out pin on the RAM that is parallel to the 9900. Pin 1 to pin 1, pin 2 to pin 2, etc. There will be eight wires in all to solder.

14) Solder wire-wrap wires to make the following connections on the RAM that is parallel to the 9900. Pin 1 goes to pin 24 of the 9900 (solder the wire to the 9900 pin on top of the board). Pin 2 goes to the 9900 pin 22. Pin 20 goes to two places. Connect pin 20 of the RAM to pin 22 of the RAM and also to pin 8 (bent out) of the 74LS21. There should be three wires coming off pin 20 of the RAM. Pin 23 of the RAM goes to pin 21 of the 9900. Pin 26 of the RAM goes to pin 23 of the 9900. Pin 27 of the RAM goes to pin 61 of the 9900 (fourth from the top on the right side). Finally, connect pin 28 of the RAM to pin 20 of the 74LS244 adjacent to the piggy-backed 74LS21.

15) Connect the following 74LS21 pins together with a bare wire: 1 2 4 and 14. Connect the short wire from the 138 pin 7 to the LS21 pin 5 (bent out). Connect LS21 pin 6 to LS21 pin 12. Connect LS21 pin 8 (bent out) to the piggy-backed 153 pin 2. Connect the short wire coming from the 138 pin 9 to LS21 pin 10. Finally, connect the 74LS21 pin 14 to the 74LS244 pin 20 that you connected the RAM pin 28 to.

16) OK, we're almost done, so take a break and have a beer.

17) On the 153, connect pin 9 to pin 13 on the 74LS32 (U605). Pin 10 of the 153 goes to pin 14 of the 74LS74 next to it (U607). Also connect pin 10 of the 153 to pins 11 and 13 of the 153. Connect pin 12 of the 153 to pin 15 of the 153, and then connect pin 15 of the 153 to pin 7 of the 74LS00 U612 (next to the 74LS74). Connect pin 14 of the 153 to pin 11 of the 74LS00 U606; that's the one you cut the trace on.

18) That's it! Now have another beer before putting your computer back together. When you try it out, remember that this version isn't compatible with any other 32K in the system.

If you have problems with this I can't promise I can help but feel free to give me a call or write EMAIL (419) 874-8838. Ask for John (or Hose-Head.)

*EDITORS NOTE: This file was downloaded from DELPHI on  
8-10-91.*

*EDITOR'S NOTE: The following message from the late John Guion is from Delphi's message base and concerns Mike Ballmann's 64k modification.*

Category 4, Topic 6

Message 3 Sun Sep 06, 1987

JOHN.J [jj] at 23:48 EDT

John Guion of the Dallas Users Group has submitted this modification to the users of Mike Ballmann's 64k on the 16 bit bus modification:

The following is how to bypass the wait-state defeat of the 16 bit memory bus modification to allow memory to be used at normal speed. This is desirable since some programs (particularly games) have compatibility problems due to the increased memory speed). This is based on Mike Ballman's modification as described by John Clulow, but will probably also work on console's modified using Brent Kropf's method. Of course, I can take no responsibility for mistakes. This works fine on two of my own consoles.

You'll need one single-pole single-throw (SPST) toggle switch, about a foot of wire-wrap wire, and soldering equipment. Be sure that the switch you use is NOT a center-off type. It should only have two positions. If you have modified your own console, you probably have all but the switch on hand.

First, locate the 74LS153 that is stacked on top of the 74LS194 (U613). From pin 9 of the LS153, there should be a wire going to pin 13 of a 74LS32 (U605). Remove this wire.

Next, find a convenient place to mount the switch in the console. I've found that mounting the switch on the main board makes disassembly of the console easier and lessens the chances of breaking a connection. In one console, I have it mounted on the empty space right next to the screw nearest the power supply board. In another, I have mounted it at the back of the board near the center where a 1/2" hole exists in the board. A notch is cut in the case to allow the handle of the switch to stick through and the switch is affixed to the board with 5-Minute epoxy.

Now, use the wire wrap wire to connect the center terminal on the switch to pin 13 of the 74LS32 where you removed the wire. Connect one of the outer terminals on the switch to pin 11 of the 74LS00 (U606) next to the 74LS32 (there should also be another added wire to that same pin). Connect the other terminal on the switch to pin 9 on the added 74LS153 that you removed the wire from.

Double check all connections and re-assemble before testing. If you've done everything right, one position on the switch will allow use of the fast internal memory, and the other will use the internal memory at regular speed.

\_jPg\_

The Case for Extended Basic Part 2  
By Art Byers

EASE OF PROGRAMMING  
EASE OF UNDERSTANDING  
Basic Wins Hands down!!

At the start of many Assembly tutorials, authors often give examples of the same program in both Basic and Assembly. Usually, the Basic is one or two lines, but the Assembly is a screen or more. The author then points out, with pride, that the assembly code, when compiled, takes up much less RAM than does the Basic.

I suppose my reaction was not what those authors expected. I saw the programming of one line vs programming more than 20 lines. Then I remembered that in over four years of programming with Extended Basic, I have NEVER run out of memory. Basic knocks Assembly out of the ball park when it comes to ease of programming, and if you write good Basic code, you have enough RAM in the expanded 99/4A for every HOME USE. (REMEMBER my premise is that this is an excellent HOME computer).

There are other very good reasons for having assembly coupled to your Basic programs and I will cover these in another chapter. Now let me offer proof of the pudding with a specific example. The 9900 Assembly code, listed below, converts a decimal number into an Integer number (ie: converts 3.1416 [PI] to 3). The code is from M. S. Morley's book "Fundamentals of TI-99/4A Assembly Language", pages 119 and 120, and is written for the Mini Memory.

What the code does is to read an ASCII encoded multidigit number and replace the decimal point and following digits with blanks. If no decimal point is found, then the number remains unchanged.

```
AORG >70B4
TEXT '3.1416'
DATA >0D00
M1 DATA >70B4
LWPI >70B8
LI R0,>2ED0
```

```
LI R1,>2000
LI R2,>0D00
MOV M1,R3
M1,R3
M1,R3
M1,R3
M1,R3
J1 CB *R3,R1
JEQ J3
CB *R3,R2
JEQ J3
CB *R3+,R0
JNE J1
DEC R3
J2 MOV R1,*R3+
CB *R3,R1
JEQ J3
CB *R3,R2
JNE J2
J3 B *R11
END
```

Remember that all this code does not even take the result and print it out on the screen or a printer - You will need five more lines of code to write to the screen and more than double that to run it out to your printer!!

Here is the same thing done in Basic INCLUDING placing it on the screen.

```
PRINT INT(3.1416)
```

Now Come on!! Which would you rather sit down and program?? I rest my case on this point. There are very good reasons for higher languages and one of them is ease of writing source code.

At this point, you may accuse me of false-weighting the scale. You may say that FORTH, PILOT, FORTRAN and 'c' are equally as easy if you want to print the integer of PI to the screen. Yes, that is true - BUT - Basic does it in English, and because PRINT and INT are reserved words represented by tokens, does it with a minimum of memory.

In general you would be correct. Nonetheless, it is in ease of understanding, for the average home computer programmer, and therefore ease of programming, that Basic shows



its advantage best.

Part 3 will examine Extended Basic's weakest point and Assembly's (and some other languages') strongest point - Speed of execution!

\*\*\*\*\*EoF\*\*\*\*\*  
\*\*\*\*\*

**DISCLAIMER**

This newsletter is brought to you through the efforts of the officers and members of the HOOSIER USERS GROUP. Every member is encouraged to submit articles.

If you have an article you would like to share with the other members mail it to:

Bryant Pedigo  
6461 N. Sherman Drive  
Indianapolis, IN 46220

Opinions expressed are those of the author and not necessarily those of the HOOSIER USERS GROUP.

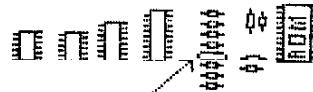
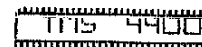
**HUG OFFICERS**

President	Gary McQuade	888-5654
V.President	Bryant Pedigo	255-7381
Secretary	Jeff Overton	299-2333
Treasurer	Walter Farmer	539-2679
Librarian	Bryant Pedigo	255-7381



*Nowadays, when the chips are down, it's usually a computer's."*

Continued from Page 3  
Speeding Up Your Console  
By Jesse C. Slicer



THIS IS WHERE THE 12,000 CRYSTAL IS LOCATED ON THE CONSOLE CIRCUIT BOARD.

FOR SALE BY: BILL LUCID

~~~~ All items must be sold ~~~~

- TI 99 4/A black and silver console \$25
- TI RS232 for P-box \$75
- TI P-Code P-box card and system software \$125
- Myarc 512K memory expansion with Myarc Ext. Basic \$175
- Horizon 3000 ( of memory for **SOLD**
- TI P-box with aux power supply \$75.00
- Pair of Qume 142 1/2 height, as is \$ 50
- Prowriter 8510AP printer \$150
- Personal library. of 300 disks DS/DD. includes deluxe locking trays \$100

Call Bill Lucid at 317-291-3995

BBS

Hoosier Users Group  
Baud rate: 300/1200/2400  
On Line 24 Hours Daily

782-994A

Now with a Hard Drive

**MATERIAL**  
 August 18, 1991  
**TIME DATED**

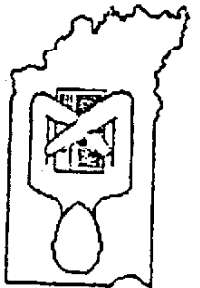
Dan H. Eicher  
 P.O. Box 605  
 Mooresville, IN 46158

May 1992



Forwarding and Address  
 Correction Requested

**HOOSIER USERS GROUP**  
 P.O. Box 2222  
 Indianapolis, IN 46206-2222



**APPLICATION FOR MEMBERSHIP**

Below you will find an application for membership to the Hoosier Users Group. Active membership entitles you to the Newsletter, up and download on the HUGbbs, attendance and voting rights at regular club meetings, access to the HUGger Library of Programs, special club activities and special guest speakers for one year.

Make check or money order payable to Hoosier Users Group. Send completed application to:

**HOOSIER USERS GROUP**  
 P.O. Box 2222  
 Indianapolis, IN 46206-2222

| Check One:                                                                                                                   | please print | cut on line  |
|------------------------------------------------------------------------------------------------------------------------------|--------------|--------------|
| Active Member                                                                                                                | NAME         | TODAY'S DATE |
| New: \$20                                                                                                                    | ADDRESS      | APT #        |
| Renewal: \$18                                                                                                                | CITY         | STATE ZIP    |
| Dues will be due in May of each year. New members subtract \$1.50 for each month from May to Oct. New member minimum \$10.00 | PHONE ( )    |              |
| Amount Enclosed                                                                                                              | INTERESTS/   |              |
| #                                                                                                                            | COMMENTS     |              |
| S                                                                                                                            |              |              |