



★

 West Jax
99er News 

APRIL 1989

The WEST JAX 99'ERS is a non-profit computer users group for the TI-99/4A Home Computer. NOT affiliated in any way with Texas Instruments. The club's mailing address is PO BOX 176 Orange Park Florida 32067.

MEETINGS are held on the Second and Fourth Tuesday of each Month in the auditorium of the Webb Library. It is located two lights west of Blanding Boulevard on 103rd Street. The first meeting of the month is the Business meeting with workshop time after adjournment. The second meeting is strictly workshop time.

*****OFFICERS*****

President...Rick Felzien.....(904) 772-9162
Treasurer...Thomas LeMay.....(904) 282-5220
Librarian.....Zach Ziegler.....(904) 389-2194

For newsletter suggestions and submissions, contact Rick Felzien.

This month we have our usual Basic Assembler installment and an article on the conversion of TI-Multiplan files for use with TI-Base.

NOTE!! In the February issue of MICROpendium there was an article by John Guion on modifying the Super ExBasic cartridge so that it can be used with a "Widget" cartridge extender.

I am happy to report that I tried the procedure and it works great.



BY
RICK
FELZIEN

The Front Ranger Feb 89

1. Macflix review
2. TI-Base review
3. Four Impact 99 segments

QB monitor Jan 89

1. Tale of three drives
2. Adding disk drives
3. Style label

Windy City News Feb 89

1. 3.5-720K drives

TI-Dings Feb 89

1. On the blue horizon
2. Word counter program

Nutmeg Ninety Niners Jan 89

1. Electronics repair?
2. EXBasic, a good choice

TICO Topics Jan 89?

1. Juffy Card review

SFV 99ers Feb 89

1. Complete CALL LOAD list
2. TPA tutorial

Wordplay Mar 89

1. Envelop program
2. Re-mapping the keyboard

LITI Mar 89

1. Avatex modem review
2. Basic Basics

TILT 99ers Jan 89

1. Reference list TIW ed.cmds.
2. Form Shop review

Bluegrass 99ers Mar 89

1. Magic Lines program
2. Ramblings of idle mind

QB monitor Feb 89

1. Console troubleshooting
(comprehensive and well written)

ROM newsletter Feb 89

1. And so forth
2. Control U cmd. list
3. P-GRAM card review

The Ottawa 99ers Mar 89

1. Intro. to Copyrights
2. A look at assembly
3. Fast EXBasic

Aloha 99ers Mar 89

1. List of command tokens
2. MYARC is a 4 letter word

Johnson Space Center Jan 89

1. Re-Setting the CPU
2. TI-Writer cue cards

Cin-Day news Feb 89

1. EXBasic problems
2. Graphics mode made easy
3. TIW tips 3

San Diego 99ers Feb 89

1. Living with spiders

Spirit of 99 Mar 89

1. Several TI-Base tutorials

Erie 99ers Mar 89

1. End of an era

New Horizons Feb 89

1. P-GRAM review

TIMP to TI-Base
Conversion By
by
Rick Felzien
West Jax 99ers

A few months back, I wrote an article on creating a check file using TIMP which would even give you any correction necessary to your checklog.

After acquiring TI-Base, I decided that I would like to do all of that same processing with my new DataBase. I tried with no success to convert the TIMP files for use with TI-Base. I thought it sure would be nice if someone came out with a conversion program. Then, as if in answer to my thoughts, along came version 2.0 of TI-Base which had a neat conversion feature. This new feature is nice but, you the files to be converted must still be of a certain format to be converted.

To convert a TIMP file for TIB, you must first use TIMP to get your files ready for conversion. The following is a step-by-step account of how I converted seven years of check files with relative ease as compared to typing in all that information in the database. One of the nice things about using TI-Base is that I could set up files by the year as compared to by the month with TIMP.

First you must use TIMP to get rid of all unwanted information. For instance, I did not want all the statement and balance data that I had set up in the lower portion of the template so I deleted all that data. I set up three extra columns for Begin Balance, Statement Balance, and Whether the check had cleared at the time of the statement. After placing the begin bal. at row 3 column 7, the statement bal. at the bottom of column 8 in the row that contained the last check for that month, and then placing "Y" for yes and "N" for no in column 9 for each check, I deleted all rows after the last check of the month. Then the first two rows could also be deleted.

Next came the task of reformatting the columns to a nice compact file. TIMP will only let you use a 3 character column or greater so column 9 was set to 3 wide, left for alignment, and general for format. Columns 4 thru 8 are formatted for a width of 8 and for the format I used decimal for alignment, fixed for format, and 2 for the number of decimal places. Column 3 is formatted to general alignment, and general format with a width of 12. Column 2 is 8 wide, general, and integer. Column 1 is set to 4 wide, aligned right, and set for text. I then re-entered the data in column 2 using the format required by TI-Base (mm/dd/yy).

Now my monthly data is ready to save in a format for use with the conversion feature. This is done by selecting Printer Options and setting up the block to be printed (i.e. R1:40C1:9) and for setup I used DSK2.(month abbrev. and the last two digits of the year). Next I set up the Printer Margins and entered 0 for left, 80 for right, 0 for beginning row, and the number of the last row of data as the last row. Now we can select Printer and the file will be printed to the disk.

After doing the preceeding to all of the months for a given year, we are now ready to combine all of the nomthly data into a yearly file using TI-Writer or equivelent Word Processor. This is accomplished by using LF, DSK2.(filename of month). This should load the month required. To add to the file use LF again and for filename use the last lone loaded plus one. For example, if the last line of the previous file came out to TIW line 85 we would use 085 DSK2.(filename). Once we load our last month's data, we should hav a continuous block of text with no blank lines. If there are any blank lines, they must be deleted. To save the file we should use the PF and a new filename. It is best to save in fixed format and for an example I will use 1985 as the year. Select PF, then use F DSK2.CK85).

Now we are ready to use the Convert feature of TI-Base to set up a nice database. Since I have two drives I changed the SETUP file to SET DATDISK=DSK2. First load up TI-Base then type CONVERT CK85 CHECK85 GO, you will then go to greate mode. Here is how my structure file looks.

CHECK	N	4	0
DATE	D	8	
PAID	C	12	
AMOUNT	N	8	2
DEPOSIT	N	8	2
BALANCE	N	8	2
BEGIN	N	8	2
STATE	N	8	2
CLR	C	1	

Once you have set up your structue the program will then process the file and set it up for TI-Base format. Once the processing stops we must still set up the database. To do this, type USE CHECK85, then RECOVER, at which time the program will again process the file and set up an index file. Next I used SORT ON CHECK which sorted my file by check number, which is my choice, however you can sort on any field that you want for your needs.

I hope this has helped the reader to understand how to convert TIMP files for use with TI-Base. In my next article I hope to include some commadn files for processing these files that were created.

THE BASIC ASSEMBLER #8 By Steve Peacock

ADD TWO NUMBERS AND PRINT ANSWER ON THE SCREEN

This month we will tackle a routine that is sort of hard to understand. How to add two number and print the answer. It is very easy to add two numbers, but printing the answer is somewhat difficult to understand. To add numbers, they are first loaded in registers and then added. This is done with the LI command

```
LI R7,53
LI R8,15
A R7,R8
```

The above loads 53 into register 7 and loads 15 into register 8. It then adds the two numbers together and puts the answer in register 8 (the second register). Register 8 now contains 68, the number we want to print. This number can not be printed as it is. We need to first break it down to a 6 and an 8.

Register 8 is moved into register 5 and register 4 is cleared.

```
MOV R8,R5
CLR R4
```

We now divide register 4 by 10

```
DIV @DIV10,R4 *DIV10 was set to >000A
```

In TI Assembly language the division is done like this: The register specified is divided by the number, the dividend is put in the specified register and the remainder is put in the NEXT CONSECUTIVE register. At this point register 5 holds >0006 and register 5 holds >0008. If we add >0030 to register 5 we get >0038. This is the code to print '8' (38 hex = 56 dec) or the ASCII for '8'

```
A @HEX30,R5 *HEX30 was set to >0030
MOV R5,@PNTANS
```

The MOV command, moves register 5 into PNTANS. This move is done in binary form, PNTANS now contains 0000000000111000b. This is the code to print '8'. Now move register 4 (>0006) into register 5, then divide by 10 and add hex 30 as before. Register 5 now contains >0036. This is the code to print '6'. This number now must be put in the left half of PNTANS, without disturbing the right half of PNTANS. To do this we use the command, SLA.

```
SLA R5,8
```

This command is carried out in binary numbers. Before the shift register 5 contained 0000000000110110b, after the shift register 5 contains 0011011000000000b.

```
MOVB R5,@PNTANS
```

The command MOVB causes the left half of the word to be moved, replacing the current left half. So PNTANS now contains 0011011000111000b. This is the final code that we need to print our answer on the screen. 00110110b is 36h and 54d, while 00111000b is 38h and 56d. All that is left to do is print this word. This is done in the last few lines of the program.

*

*PROGRAM BABA==>Basic Assembler #8 Assembly Version

*ADD TWO NUMBER AND PRINT ANSWER ON SCREEN

*(C)1985 S. PEACOCK

*

REF VMBW

DEF START

PNTANS BSS 2 *BlockStartingSymbol

*****RESERVES 2 BYTES FOR

*****THE ANSWER

HEX30 DATA >0030 *HEX 30 TO BE USED TO ADD

*****TO REMAINDER AFTER DIVISION

*****BY 10. THIS IS NEEDED TO CONVERT A

*****NUMBER TO IT'S CODE, THAT CAN BE

*****PRINTED. EX >0009 + >0030 = >0039

*****>0039 IS 57d, THE ASCII FOR '9'

DIV10 DATA >000A *HEX 10 TO BE USED TO DIVIDE

*****BY 10

START LI R7,03 *PUT 03 IN REG. 7 (THESE ARE THE TWO

LI R8,15 *PUT 15 IN REG. 8 (NUMBERS WE WILL ADD

A R7,R8 *ADD REG. 7 AND REG. 8 (68d)

MOV R9,R5 *MOVE REG. 8 (68d) INTO REG. 9

CLR R4 *CLEAR REG. 4

DIV @DIV10,R4 *DIVIDE REG. 4 BY 10.

*****THIS COMMAND IMPLIES THAT REG. 4 AND REG. 5

*****WILL BE USED IN THE DIVISION. WHEN

*****REG. 4 IS SPECIFIED THE NUMBER IN THE

*****NEXT REGISTER IS DIVIDED BY THE DIVISOR.

*****THE DIVIDEND IS PUT IN REG. 4 AND THE

*****REMAINDER IS PUT IN REG. 5

*****BEFORE THE DIVISION REG. 4=>0000 REG. 5=>0044

*****AFTER THE DIVISION REG. 4=>0006 REG. 5=>0008

A @HEX30,R5 *ADD >0030 TO REG. 5. REG. 5 NOW =

*****>0008+>0030 OR >0038. THIS IS THE

*****THE CODE TO PRINT THE NUMBER '0'. THE

*****SECOND DIGIT OF OUR ANSWER

MOV R5,@PNTANS *(PrintAnswer) NOW CONTAINS >0038

MOV R4,R5 *MOVE REG. 4 (>0004) INTO REG. 5

CLR R4 *CLEAR REG. 5

DIV @DIV10,R4 *DIVIDE REG. 4 BY 10 PUT DIVIDEND IN REG. 4 AND

*****REMAINDER IN REG. 5

*****BEFORE DIVISION REG. 4=>0006, REG. 5=>0000

*****AFTER DIVISION REG. 4=>0000, REG. 5=>0006

*****($6/10=0$ remainder of 6)

A @HEX30,R5 *ADD 30 TO REG. 5. REG. 5 NOW = >0036 (CODE FOR '6')

SLA R5,8 *ShiftLeftArithmetic

*****THIS COMMAND TAKES THE CONTENTS OF REG. 5 AND SHIFTS

*****IT 8 PLACES TO THE LEFT. THIS IS DONE

*****USING BINARY NUMBERS.

*****BEFORE THE SHIFT REG. 5=000000000110110

*****AFTER THE SHIFT REG. 5=0011011000000000

MOVB R5,@PNTANS *MOVE LEFT HALF OF REG. 5 INTO PNTANS

*****BEFORE THE MOVB PNTANS=0000000000111000

*****AFTER THE MOVB PNTANS=0011011000111000
*****THIS MOVE USES THE MOVB COMMAND. THIS MOVES
*****THE LEFT HALF, NOT THE WHOLE, AS DOES THE
*****COMMAND MOV THAT WAS USED EARLIER.

```
LI   R0,35      *SCREEN ADDRESS TO PRINT ANSWER
LI   R1,PNTANS  *LOAD REG. 1 WITH PNTANS
LI   R2,2       *TWO BYTES TO PRINT '6' AND '8'
BLWP @VMBW     *PRINT '68'
JMP  $         *JUMP TO 'SELF'
END
```

```
100 REM PROGRAM BABB==>Basic Assembler #8 Basic Version
110 REM ADD TWO NUMBERS AND PRINT ANSWER ON SCREEN
120 REM (C)1985 S. PEACOCK
130 REM YOU MAY WANT TO PUT A 'CALL CLEAR' HERE
140 DEC48=48 !HEX30 IS DEC48
150 DIV10=10
160 REG7=53
170 REG8=15
180 REG9=REG7+REG8
190 REG5=REG8
200 REG4=INT(REG5/DIV10)
210 REG5=((REG5/DIV10)-REG4)*10
220 REG5=REG5+DEC48
230 PNTANS1=REG5
240 REG5=REG4+DEC48
250 PNTANS2=REG5
260 CALL HCHAR(2,5,(PNTANS2))
270 CALL HCHAR(2,6,(PNTANS1))
280 GOTO 280
290 END
```