P-GRAM+

GROM EMULATOR AND

REAL-TIME CLOCK CARD

OPERATING GUIDE

PRODUCED BY HORIZON COMPUTER

DISTRIBUTED BY BUD MILLS SERVICES

Manual by John P. Guion

(C) Copyright 1988, Bud Mills Services

******* IMPORTANT *******
******* IMPORTANT *******
******* IMPORTANT *******

TI-99/4A QI Console Incompatibility

The P-GRAM GROM Emulator and Real-Time Clock Card does not currently function with the QI (Quality Improved) version of the TI-99/4A console. These consoles, manufactured in late 1983 and early 1984 used different internal circuitry that does not permit access to GROM data through the Peripheral Expansion System.

QI consoles were built only with tan-colored cases, but NOT ALL TAN CONSOLES ARE QI UNITS. Black and silver-colored console do not contain the QI circuitry. To determine if a tan console has the QI circuitry, look into the expansion port on the right side of the console where the Peripheral Expansion System is connected. If the connector in this port is surrounded by silver-colored "fingers", the console contains the QI circuitry and is not compatible with the P-GRAM. If the connector is surrounded by gold-colored "fingers", the console is not a QI unit and will function properly with the P-GRAM.

Attempting to use a QI console with the P-GRAM card will not harm any equipment. However, the P-GRAM will not be able to function properly and may lock up the computer until either the P-GRAM is removed from the system or the console is replaced with a non-QI unit.

P-GRAM REFERENCE CARD

BASIC CALLs:

```
 CALL PG
 CALL PGOF
 CALL PGON
 CALL PGZAP
 CALL PTIME
```

Hot Keys

A and Z pressed simultaneously

CRU Bit Codes:

```
 11=Enable lower 8K DSR RAM
 09=Enable upper 8K DSR RAM

 12=Enable GRAM/lower 8K module RAM
 0A=Enable GRAM/upper 8K module RAM

 00=Disable all P-GRAM memory
```

CRU Bit Usage

```
 Bit 0 - (OFF) DSR paged out
         (ON) DSR paged in

 Bit 1 - (OFF) GRAM/RAM disabled
         (ON) GRAM/RAM enabled

 Bit 2 - (OFF) P-GRAM write enabled
         (ON) P-GRAM write protected

 Bit 3 - (OFF) Low 8K RAM locked in
         (ON) Bank switching enabled

 Bit 4 - (OFF) High 8K RAM locked in
         (ON) Bank switching enabled
```

P-GRAM OPERATING INSTRUCTIONS

The first three sections of this manual describe what is required to get the P-GRAM card up and running and should be read thoroughly. The next four sections deal with using the P-GRAM's clock and using the memory editor to modify modules and the P-GRAM's operating system. The last section contains technical information about the card and software for users who wish to make involved changes or who simply want to understand the P-GRAM better.

Although the information in this manual may at first seem overwhelming, the method of operation that is described is actually very simple. The P-GRAM card and its software have been designed to facilitate easy use without compromising features.

Only two or three commands are needed for regular use of the card and only a few more to handle the initial setup. Beyond this, powerful capabilities have been added to allow access to all of the P-GRAM's features.

Understanding all of these capabilities in detail is not required. They have been provided for those users who wish to further explore the TI-99/4A and expand its abilities. It is recommended, however, that each user read the entire manual to become aware of all the functions of the P-GRAM for a better understanding of the device.

INSTALLING THE P-GRAM CARD

The first thing to do before installing the P-GRAM card is set the CRU base address for the card. This determines the portion of memory at which the P-GRAM card will respond. If a device in the system is not listed below, consult the user's manual on that device to determine its CRU base and avoid using that base for the P-GRAM. The available CRU bases and the commonly used devices at these bases are:

CRU BASE	DEVICE
>1000	Unassigned
>1100	Disk Controller
>1200	Unassigned
>1300	RS232 (Primary)
>1400	Unassigned
>1500	RS232 (Secondary)
>1600	Unassigned
>1700	HEX-BUS adaptor

Since there is no advantage to using a low CRU base for the P-GRAM card, using the highest available CRU base is suggested. Unless another device already occupies >1700, this CRU BASE is recommended for the P-GRAM card.

Once a CRU base is picked, the card must be set to respond at that CRU base. On the P-GRAM card, there is a eight-position switch used for selection of the CRU base address. Only one switch may be set at a time. The switch settings are as follows (note that switch 1 is located nearest the top of the circuit board):

1->1000
2->1200
3->1300
4->1400
5->1500
6->1600
7->1700
8->Unused

Simply turn ON the switch that corresponds to the desired CRU base, leaving the other seven switches OFF.

To install the card, turn off the Peripheral Expansion System (P-Box) and remove its top cover. Be sure to wait two minutes after turning the system off before inserting or removing the P-GRAM or any card in the P-Box. The P-GRAM card may then be inserted into any empty slot in the P-Box.

LOADING THE DSR

To load the P-GRAM's operating system (the DSR), first be sure that the switch on the back of the card is in the UP position so that the card is enabled. This switch can disable all functions of the P-GRAM card except the clock. There is some chance that a badly corrupted operating system could prevent power up (as battery backed RAM-Disks sometimes do). If this occurs, simply move the switch to the DOWN position during power up and then turn the card on again before loading the DSR. Removal of the battery is not required. The DSR included with the P-GRAM is loaded using option #2 of the loader described below and is on the disk named PGSYSTEM. The name of the file is PGDSR.

Before attempting to load the DSR, place the disk labeled PGSYSTEM into drive #1. The DSR loader may then be loaded into the computer using one of three methods:

EDITOR/ASSEMBLER:
Insert an Editor/Assembler module into the console and select option #5 of the module. Enter the file name DSK1.UTIL1. The DSR loader will then load and run.

TI-WRITER:
Insert the TI-Writer module into the console and select option #3 of the module. Press enter to load the file DSK1.UTIL1.

Extended BASIC:
Insert the TI Extended BASIC module into the console. Select Extended BASIC. The program DSK1.LOAD will load and execute the file DSK1.LOAD1 which will load the DSK1.UTIL1 program.

Once the UTIL1 program has loaded, five options will be available:

1 Initialize DSR Memory

2 Load Memory Image

3 Save Memory Image

4 Load Object Code

5 Quit

After selecting an option, type the CRU base that the card occupies (i.e. >1700). The loader will then proceed by asking for a file name, if required by that option. If an undesired option is selected, FCTN 9 (BACK) will cancel that option and return to the loader menu.
CAUTION: Do not select the CRU base of any card other than the P-GRAM. If the CRU base of another card with a RAM-based DSR is selected, that DSR will be corrupted and must be reloaded.

Initialize DSR Memory

Option (1) clears the RAM area on the card in which the DSR is stored. This should always be done before loading a DSR in order to prevent errors caused by mixing code. Once the CRU base is entered, the loader will clear that DSR space.

Load Memory Image

Option (2) will load a 16K DSR into the P-GRAM which has been saved in memory image (PROGRAM) format. This is the method recommended for loading files since it is faster and more compact than the compressed object code format. This is the only format in which the loader program will SAVE a DSR from the card. To load using this option, select the CRU base as described above. Then enter the name of the memory image DSR file that is to be loaded into the P-GRAM card. On the PG-SYSTEM disk included with the P-GRAM card, this file is named PGDSR. PGDSS is the second 8K of the DSR and will automatically be loaded as well. This function will always load the specified file name into the upper 8K of the DSR RAM and then increment the last character of the file name and search for another file. For example, when PGDSR is given for the file name, the program will load that file and then look for PGDSS. The loader will then load the second file into the lower 8K of the DSR RAM then return to the loader menu.

NOTE: If the DSR to be loaded only occupies 8K of the 16K available space, it must exist in two separate files on the disk with consecutive names as described above. The loader will then load the same data into both DSR banks. Failing to do this will prevent the DSR from operating properly.

When this option is used to load an operating system saved in memory image format, the loader program checks the P-GRAM card to see if the optional clock is installed. If the clock is not installed, the loader will disable the clock features in the P-GRAM operating system. If the clock is later added to the card, the operating system must be reloaded to enable the clock features.

This option of the loader also checks the system to see if a CorComp clock is installed. If so, the loader will automatically change the "CLOCK" device name in the P-GRAM's operating system to "clock". This prevents possible conflicts between a CorComp clock device and the optional real-time clock on the P-GRAM card.

Save Memory Image

Option (3) will save the DSR currently in the P-GRAM card in memory image format. This is useful when converting a previously loaded object file to memory image format or when saving changes made directly to the DSR residing in the card. This option will create two files using the naming convention described above with the first file being the upper 8K of the DSR and the second file being the lower 8K.

IMPORTANT: NEVER SAVE THE OPERATING SYSTEM TO DISK IN PLACE OF THE ORIGINAL OPERATING SYSTEM. If the operating system is

saved back to the same disk as the original operating system and is given the same file name, the original operating system will be lost. Always keep a back up copy of the original operating system.

Load Object Code

Option (4) will load a 16K DSR into the P-GRAM card that has been saved as two compressed object files (DIS/FIX 80). Use of this option is only required when the user has modified or reassembled the operating system of the P-GRAM card. This option is not needed to load an operating system saved in memory image (PROGRAM) format. When loading object code with this option, each 8K DSR bank must be contained in a separate file. The files must be consecutively named using the same convention as described for loading memory image files. Two files are required, therefore, the requirement that 8K DSRs must exist in two identical files with consecutive names also applies.

Quit

Option (5) exits the loader and resets the computer.

USING THE P-GRAM

Operation of the P-Gram card is fairly simple, only requiring user intervention when a different module is desired.

When the console is reset or is first turned on, the P-GRAM card will check to see if a module is plugged into the console. If a module is plugged in, the P-GRAM will disable itself and allow normal use of the module. If no module is plugged into the console, the P-GRAM card will turn itself on and any module loaded into the P-GRAM card will appear on the normal module menu.

The only time that any other action is required by the user is when the contents of the P-GRAM card are to be cleared, loaded, saved, or changed. The method used to access these features is by using the one of four CALL statements from TI BASIC or Extended BASIC or by using the "hot keys". The CALLs may be typed in command mode or from a TI BASIC program (these cannot be used directly from a running Extended BASIC program). The CALLs are:

CALL PG	Accesses P-GRAM menu
CALL PGOF	Turns the card OFF until the console is reset
CALL PGON	Turns the card ON until the console is reset
CALL PGZAP	Instantly disables any modules loaded into the card

The "hot keys" are used when a particular module (such as an auto-start module) or some other device prevents the user form accessing BASIC. To enter the P-GRAM loader using the "hot keys", hold down the "A" and "Z" keys at the same time while resetting or turning on the console. This has the same effect as using CALL PG from BASIC.

*ca;; {G

The CALL PG command is used for nearly all operations of the card. When CALL PG is entered, a menu-driven utility program is immediately loaded. When first loaded, a credit screen will appear. This screen will change to the operating screen after a few seconds. Pressing any key will immediately move from this screen to the operating screen. Once the operating screen is visible, six functions will be offered:

1. Initialize GRAM/RAM
2. Load P-GRAM
3. Save P-GRAM (or Save Module)
4. Memory Editor
5. Change Grom Page
6. Quit

Pressing keys 1 through 6 will select the appropriate function. ENTER is not required. The GROM page number is displayed on screen.

If the optional real-time clock is installed, the date, day of the week, and time will also be displayed at the bottom of the screen.

Initialize GRAM/RAM

This is used to clear the P-GRAM card's GRAM and module RAM memory prior to loading a module from disk. This option is not an automatic function of loading the P-GRAM since there may be certain instances when two modules (that do not have conflicting GRAM/RAM is important because if memory is not cleared before loading the desired modules, extra unused memory banks may be saved if the card's contents are saved, thus wasting disk space and loading time.

As soon as this option is selected, a diamond will appear next to each memory bank as it is cleared. When initializing, loading, or saving occurs, the GRAM or RAM banks will have this diamond appear next to the when the function for that bank is completed. When the option is completed and all required banks have been operated upon, press any key to return to the P-GRAM menu.

Load P-GRAM

This option is used to load a module from disk into any PAGE of the P-GRAM + card. The files must be modules saved by the P-GRAM card, a Gram Kracker, or J. Peter Hoddie's CartSaver program. The files may be 34 sector ROM files and 34 or 26 sector GROM files, depending on the files needed for that particular module. When the option is selected, the prompt "DSK" will appear. The entire module file name must be entered. The module may be loaded from any valid drive, including RAM-Disk devices. The P-GRAM will then proceed to load the files in descending order until all files are loaded. When finished, pressing any key will return to the P-GRAM menu.

NOTE: If a module file contains data for only one of the two RAM banks, it will automatically be loaded into both banks. This prevents software errors from occurring due to accidental bank switching of the module space.

SAVE P-GRAM (or Save Module)

IMPORTANT: The Save function may be used for either saving the contents of the P-GRAM card OR saving the contents of a module plugged into the module port. If no module was inserted into the port when the console was reset, this option on the P-GRAM menu will show "SAVE P-GRAM". This indicates that the SAve function will save the contents of the P-GRAM card to disk. If a module was inserted when the console was reset, the P-GRAM menu will show "Save Module" and the SAve function will save the contents of that module into a file to be loaded back into the P-GRAM later. This is the method through which all module files

for the P-GRAM may be created. To date, the only modules that are not supported are the Milton Bradley MBX modules, some DataBioTics modules that use CRU bank switching, and modules which have connections to external devices like the Super Sketch interface. CAUTION: DO NOT SAVE pages 2,3, or 4 from the PGRAM+ to disk.

The P-GRAM can be made to function as a Supercart (an Editor/Assembler module with an extra 8K of RAM memory). To do this, an Editor/Assembler module must be loaded into the P-GRAM card and a small change must be made using the P-GRAM's memory editor. This change is described in the "Module Modifications" section of this manual.

The Save function will ask for the name of the file to be created in the same method as when loading files. It will also ask if short files are to be created. If "N" (the default) is specified, all 8K of data in each occupied memory bank will be saved in a series of 34 sector files. If "Y" is specified, only the first 6K of each GRAM (or GROM) bank will be saved as a 26 sector file and all 8K of each RAM (or ROM) bank will be saved as normal 34 sector files. The reason for this is that nearly all GROM modules only use 6K of the 8K space in each bank. All TI produced modules use only 6K GROMs, so short files may be used for these modules and disk space can be conserved. In some non-TI produced modules (like Triton's Super Extended BASIC) or special module-type files, all 8K of the GROM space in each bank is used. Short files cannot be used for these applications. When in doubt, do not specify short files so that any memory required will be saved.

The files will be saved in descending order relative to their place in memory. The first file of the series will be given the name that was entered above. The second file will have the same name and a "1" will be added to the name. The third file will have a "2" added and so on until all used memory banks are saved. If ten characters are used for the file name, the last character of the file name will be incremented as successive files are created. Any memory banks that have been loaded into the card will be saved and unused banks will not be saved. This is why it is important that the P-GRAM be initialized prior to loading modules because "left over" data from a previously loaded module might be saved as an unwanted file.

When saving the P-GRAM or module is completed, press any key to return to the P-GRAM menu.

Memory Editor

For information on the Memory Editor, consult the "Using the Memory Editor" and "Editing the P-GRAM" sections of this manual.

Quit

Selecting this option will reset the computer. The computer will return to the normal TI title screen or the power up screen of optional devices if they normally do so after a reset.

CALL PGON

This call will turn on the GRAM/RAM memory if it is not already enabled. This might be used to enable the GRAM/RAM from a TI BASIC program in order to access special routines of the GRAM/RAM even though a module is inserted in the port.

This command is normally not needed for operation of the P-GRAM card. It is only provided to simplify custom applications where control over the presence of the P-GRAM's memory is required by the user.

The GRAM/RAM memory may be disabled by executing the CALL PGOF command.

When the console is reset or powers up, control of GRAM/RAM enabling is return to the P-GRAM's software.

NOTE: The CALL PGON and PGOF should not be used from Extended BASIC. Doing so will usually lock up the machine.

CALL PGOF

The CALL PGOF command is the opposite of the PGON command. It is used for disabling the GRAM/RAM memory from TI BASIC.

The GRAM/RAM may be enabled by using the CALL PGON command.

The same considerations for using the CALL PGON command apply to the CALL PGOF command.

CALL PGZAP

In some rare cases, it may be necessary to disable the contents of the P-GRAM card. This might occur if corrupted module data is present in the P-GRAM card. Such data problems might cause problems in operating the computer and may require clearing of the GRAM/RAM memory. This command allows the user to escape from a faulty data situation.

Executing CALL PGZAP will clear all the GRAM/RAM headers from the P-GRAM's memory banks. The CALL PGZAP is different from initializing the P-GRAM card in that it only erases the module headers from each memory bank of the card. Using this command allows clearing of the card without resetting the computer. Also, since ONLY the headers are erased, this allows the current contents of the GRAM/RAM banks to be revived through use of memory editing. This might be useful if the problem was the result of custom modifications being made to the contents of the P-GRAM card.

USING THE CLOCK

The real-time clock on the P-GRAM card is an option that is not necessary for normal function of the P-GRAM card. This clock can provide time and date information to the computer for either display or use by a program.

NOTE: If an MBP Real-Time Clock card is already being used in the system, the P-GRAM will provide all the features described below without having the clock option installed on the P-GRAM card. If a CorComp stand alone clock or Triple Tech card is used in conjunction with the P-GRAM clock, all of the original functions provided by the CorComp clock will be available in addition to the features of the P-GRAM clock. When the P-GRAM's operating system is loaded from a memory image file, the loading program checks the system to see if a CorComp clock device is attached. If so, the loader program will change the P-GRAM's clock name from "CLOCK" to "clock". This prevents possible conflicts from having two clocks installed and allows the other features of the P-GRAM clock to be used.

The clock keeps track of hours, minutes, seconds, date, month, year, and the day of the week. In order to make the clock as useful as possible, three methods of access are available:

1) Access through CALL PTIME

2) Access as a CorComp stand-alone or Triple Tech card

3) Access as an MBP Real-Time Clock card

Any method of accessing the P-GRAM clock may be used. The multiple formats have been provided to make the clock suitable for the widest range of applications possible.

ACCESS THROUGH CALL PTIME

The easiest way to read and set the clock is to select BASIC or Extended BASIC and type CALL PTIME, followed by ENTER. Doing this will display the day of week, date, and time on the screen and list three options:

1) Set the Clock

2) Execute the P-GRAM Loader

3) Quit

To set the clock, press "1" and just type the correct information over the current display. Pressing "2" will have the same effect as using CALL PG. Pressing "3" will reset the computer. If improper information is entered while setting the time (an impossible date or time, etc.) the computer will reset.

NOTE: CALL PTIME does not return to BASIC. Therefore, any program currently in memory will be lost.

ACCESS AS A CORCOMP CLOCK

With the exception of software that directly accesses the CorComp clock chip, the P-GRAM clock is compatible with software written for the CorComp stand-alone and Triple Tech clocks.

The CorComp clocks are accessed as a file named CLOCK. Three variables are either read from or printed to the file, depending on whether the clock is to be read or set. The clock may be read easily from a BASIC program using the following code (this may be entered as a program itself in BASIC or Extended BASIC):

```
100 OPEN #1:"CLOCK"
110 INPUT #1:A$,B$,C$
120 PRINT " The day is:";A$
130 PRINT "The date is:";B$
140 PRINT "The time is:";C$
150 CLOSE #1
```

The day of the week is returned as a string variable, but is a number from 0 to 6 (Sunday=0, Saturday=6). The time is in 24-hour format.

Setting the clock is accomplished by PRINTing to the clock file. The information must be in the same format as when it is read from the clock (i.e. the day of week must be a string variable, the date must be separated by slashes, and the time must be separated by colons). All information must be in one string, separated by commas, and no spaces may be used. The following BASIC program will set the clock:

```
100 OPEN #1:"CLOCK"
110 INPUT " Day: ":A$
120 INPUT "Date: ":B$
130 INPUT "Time: ":C$
140 PRINT #1:A$;",";B$;",";C$
150 INPUT #1:D$,E$,F$
160 PRINT " The day is:";D$
170 PRINT "The date is:";E$
180 PRINT "The time is:";F$
190 CLOSE #1
```

Enter the program in BASIC and run it. To set it for Tuesday, March 23, 1988, 2:04:23 pm, the information to enter will be 2, 03/23/88, and 14:04:23. The program will then read the clock and display the current time.

One popular application that is not directly compatible with the P-GRAM's clock is the Horizon RAM-Disk MENU software by John Johnson. Unfortunately, the simplest way to resolve this problem and still retain all the features of the P-GRAM clock is to alter the MENU program itself. The problem occurs because the MENU program tries to access the CorComp clock chip directly after finding the CLOCK name on the P-GRAM card. To fix this, first save all the files on the RAM-Disk to a floppy in case a mistake is made. Next, use a disk sector editor to alter the CLOCK name in the MENU program. Search for the word CLOCK in the MENU

program file. Type over the word CLOCK with spaces and write the sector back to the RAM-Disk. This prevents MENU from finding the CLOCK name on the P-GRAM card and forces MENU to use P-GRAM clock as if it were an MBP clock card. This same method may be used in other programs that will recognize both CorComp and MBP clocks.

ACCESS AS AN MBP CLOCK

The clock chip may be directly accessed through use of CALL PEEK and CALL LOAD from either Extended BASIC or TI BASIC with the Editor/Assembler module. The data passed to and from the clock chip is in BCD format and must be converted to and from decimal information to be used. All functions available with the other methods of access are available through this method except for the year. The year is provided through software in DSR operating system that calculates the present year based on conditions during console reset.

The following program shows how to set then read the clock in MBP clock format.

```
100 CALL INIT
110 DEF DB(X)=X+6*INT(X/10)
120 DEF BD(X)=X-6*INT(X/16)
130 INPUT "Seconds: ":A
140 INPUT "Minutes: ":B
150 INPUT "Hours: ":C
160 INPUT "Day of week: ":D
170 INPUT "Day of month: ":E
180 INPUT "Month: ":F
190 A=DB(A)
200 B=DB(B)
210 C=DB(C)
220 D=DB(D)
230 E=DB(E)
240 F=DB(F)
250 CALL LOAD(-31164,A,0,B,0,C,0,D,0,E,0,F,0)
260 CALL SOUND(1,110,30)
270 CALL PEEK(-31164,A,A1,B,B1,C,C1,D,D1,E,E1,F,F1)
280 A$=STR$(BD(D))
290 B$=STR$(BD(E))&"/"&STR$(BD(F))&"/88"
300 C$=STR$(BD(C))&"/"&STR$(BD(B))&"/"&STR$(BD(A))
310 PRINT " The day is:";A$
320 PRINT "The date is:";B$
330 PRINT "The time is:";C$
```

Notice that all information to set the clock uses the DB function to convert decimal numbers to BCD and that information read from the clock uses the BD function to convert BCD to decimal. When using this method to access the clock chip, the day of week is offset by one (Sunday=1, Saturday=7). Also note that a CALL SOUND statement must follow the CALL LOAD used to set the clock. This is required to reset the sound generator since setting the clock also enables that chip.

USING THE MEMORY EDITOR

A complete memory editor is available from the P-GRAM menu screen. The memory editor allows the user to inspect, alter, move, search, print, fill, or save to disk any portion of memory in the TI-99/4A. Using this program is similar to using a sector editor for altering or inspecting data on a diskette. This is included to provide an easy and direct way for the P-GRAM user to modify the contents of the P-GRAM card and other devices so that on the P-GRAM +- THE EDITOR ONLY FUNCTIONS ON PAGE 1 OF THE VOLUME MODULE LIBRARY.

NOTE: Any changes made to modules in the P-GRAM card will be lost if a new module is loaded. The user should save the module if the changes are intended to be permanent. This also applies to changes made to the P-GRAM DSR.

The editor allows the user to manipulate CPU, GROM, and VDP memory. Additionally, the ability to set CRU (Communications Register Unit) bits has been provided. The ability to control CRU bits is needed to give the user access to DSR RAM, module RAM, and GRAM on the P-GRAM card. It may also be used to access the DSR memory of other cards or to manipulate functions controlled by the CRU interface. To make modifying memory easier, features have been added to allow searching for particular data, dumping memory blocks to printer or disk, filling blocks of memory, and moving blocks of memory. This section of the manual describes using all these features.

EDITOR COMMANDS

The editor is controlled primarily through the use of function keys. These commands are as follows:

FCTN 1 (CRU Bit Select) - Selects CRU base address and sets CRU bits.

FCTN 2 (Move) - Moves the contents of one block of memory to a user specified location in any other place in memory.

FCTN 3 (Fill) - Fills a block of memory with a byte specified by the user.

FCTN 4 (Page up) - Pages the contents of the memory window forward 144 bytes.

FCTN 5 (Search) - Searches a specified memory block for a particular piece of data.

FCTN 6 (Page down) - Pages the contents of the memory window back 144 bytes.

FCTN 7 (Print) - Dumps the specified memory block to printer in Hex, ASCII, and BASIC Biased ASCII.

FCTN 8 (Dump) - Dumps the specified memory to disk in memory image format.

FCTN 9 (Back) - Moves the cursor to and from the memory editing window to allow alteration of memory.

FCTN 0 (Bias) - Toggles the ASCII memory window display to BASIC Biased ASCII and back.

FCTN = (ASCII Hex) - Toggles between ASCII (or BASIC Biased ASCII) and Hexadecimal memory window displays.

CTRL = (Quit) - Quits the editor and returns to the P-GRAM menu.

ARROW KEYS - When outside the memory window, the arrow keys move among the five input fields at the top of the screen. When in the memory window, they will move to any location in the window.

ENTER - When outside the memory window, this will move to the beginning of the next input field. When in the memory window, this will return to the memory location in the upper left corner of the memory window.

cA00E (Current memory) - This shows the current memory type and location of the contents in the memory window. When the editor first comes up, it will always be set at this location. The "c" in front of the address designates memory type (c=CPU memory, g=GROM memory, and v=VDP memory). To change the type or location, simply type over the existing data.

Start 0000 - This determines the beginning point for a Move, Fill, Search, Print, or Dump operation. The memory type affected is determined by the current type of memory selected for the memory window.

Finish 0000 - This designates the ending point for a Move, Fill, Search, Print, or Dump operation. The memory type affected is determined by the current type of memory selected for the memory window.

Dest c0000 - This determines the beginning memory address and type used as the destination for the Move command. The beginning and ending address and memory type for the source location are determined by selecting memory type, start, and finish points as described above. This allows the user to move a block of memory to another location in the same or different type of memory.

Fill 00 - This designates the byte to be used in the Fill command. The start and finish addresses and memory type for the block to be filled are described above. This byte will be used to fill the specified memory block.

Cursor c0000 - This shows the current location of the cursor within the memory window.

CRU Base 0000 00 - The first four numbers show the current CRU base address being accessed and the second two numbers

are the current CRU bit code.

CRU Bit Select

The CRU Bit Select option allows the user to set or clear CRU bits. To use this command, press FCTN 1 and the cursor will move to the CRU base prompt near the lower right corner of the screen. Type the CRU base address of the peripheral to be accessed. For example, if the P-GRAM was set at >1700 and is going to be accessed, >1700 should be entered as the first four numbers at this prompt.

The next number represents the Hex sum of the CRU bits to be turned on. These bit must be manipulated to modify the contents of the P-GRAM card. Any value may be entered here. The specific values needed for editing the P-GRAM card have been computed and are listed in the section "Editing the P-GRAM".

To set certain CRU bits, the binary code representing the status of each bit is converted to Hex. For example, if bits 5, 2, and 0 are to be turned on, the value >25 must be entered for the second half of the CRU bit prompt. Here is how the >25 value is computed:

```
Bits:   1  0  0  1  0  1
Value:  32 16 8  4  2  1
Sum:    32+0+ 0+ 4+ 0+ 1=37
```

Convert 37 to Hex to get >25. This is the value to enter. For a more in-depth explanation of handling CRU bits and their uses, consult the Editor/Assembler manual.

Once the proper values have been entered, press FCTN 1 again to set the CRU bits.

To avoid serious mix-ups by activating multiple DSRs concurrently, all DSRs used for peripherals are reset when the editor is loaded. Also, the editor resets the DSRs whenever a new CRU base is selected and when CTRL = is used to leave the editor.

Move

Pressing FCTN 2 will execute the move command and copy the contents of one block of memory to a user-specified destination. Before executing this command, however, the beginning, ending, and destination of the memory blocks must be designated.

First, move the cursor to the prompt at the upper left corner of the editor screen. At the first character of the prompt, type c, g, or v to designate from which type of memory the Move is occurring. Next, move over to the Start prompt and enter the address of the beginning of the memory block to be moved. Then move to the Finish prompt and enter the address of the last location to be moved. Now go to the Dest prompt and first enter a letter for the type of memory

into which the block will be moved followed by the address that will begin the new block. Ignore the Fill prompt. For example, to move the contents of console GROM 1 to expansion RAM, enter to following parameters:

g2000, Start 2000, Finish 3FFF, Dest cE000.

Once the source and destination parameters have been set, pressing FCTN 2 will execute the Move function.

Fill

Pressing FCTN 3 executes the Fill command. This will allow the user to set all bytes in a specified block of memory to one value. This is most useful for clearing a block of memory. Before executing the Fill command, the parameters for the beginning and ending of the memory block as well as the byte with which the block will be filled must be specified.

To do this, first move the cursor to the prompt at the upper left corner of the editor screen. At the first character of the prompt, type c, g, or v to designate which type of memory to be filled. Next, move over to the Start prompt and enter the address of the beginning of the memory block to be filled. Then move to the Finish prompt and enter the address of the last location to be filled. Now go to the Fill prompt and enter the byte that will occupy all memory locations within the specified block. Ignore the Dest prompt. For example, to set all bytes between >E000 and >E7FF in CPU memory to FF, enter the following parameters:

cE000, Start E000, Finish E7FF, Fill FF.

Once entered, pressing FCTN 3 will fill the block with the specified byte.

Page up

Pressing FCTN 4 will cause the contents of the memory window to advance 144 bytes. The current memory location will be displayed in the upper left corner of the screen as the contents of the window changes.

Search

Pressing FCTN 5 activates the Search feature. This feature allows the user to search any type of memory (CPU, VDP, or GROM) for a Hex, ASCII, or BASIC Bias ASCII string of characters. Before executing the Search command, the parameters for the type of memory to be searched, the beginning and ending of the memory block, and the string to be searched for must be specified.

To do this, first move the cursor to the prompt at the upper left corner of the editor screen. At the first character of the prompt, type c, g, or v to designate the type of memory to be searched. Next, move over to the Start

prompt and enter the address of the beginning of the memory block to be searched. Then move to the Finish prompt and enter the address of the last location to be searched. Ignore the Dest and Fill prompts. Next, use FCTN O and FCTN = to select Hex, ASCII, or BASIC Bias ASCII mode, depending on what kind of string is to be searched for.

Press FCTN 5 to select the search feature. A flashing cursor on a white background (reversed video) will appear at the bottom of the memory window. Type the string to search for in this window. Once the string has been typed, use the arrow keys to back up the cursor so that it is placed on the last character to be searched for. This means that if a five character string is to be searched for, back the cursor up so that it is on top of the fifth character. Once the cursor has been positioned to indicate the last character of the search string, press ENTER. The editor will then proceed to search the specified memory block for the specified string. If the string is found, the area of memory containing the string will be displayed in the memory window with the search string in the upper left corner. If the string is not found, the editor will return to command mode but leave the reversed video prompt at the bottom of the screen. For example, to search for the "TI BASIC" name in console GROM, select ASCII mode and enter the following parameters:

g0000, Start 0000, Finish 3FFF, press FCTN 5 and type TI BASIC, back the cursor over the C and press ENTER.

The editor will then proceed to search for the TI BASIC name that appears on the TI menu.

Page down

Pressing FCTN 6 will cause the contents of the memory window to show the next lower 144 bytes of memory. The current memory location will be displayed in the upper left corner of the screen as the contents of the window changes.

Print

Pressing FCTN 7 will execute the Print function. This will dump the contents of a specified block of memory to a printer in Hex, ASCII, and BASIC Biased ASCII. Before executing the Print command, the block of memory and the printer device name must be specified.

The default for the printer device name is PIO. If this needs to be altered, it should be done before the other parameters are set. To do this, move the cursor to the memory location prompt in the upper left corner of the screen and enter cA03F. If necessary, use FCTN O and FCTN = to toggle to an ASCII display WITHOUT the BASIC Bias. PIO should now appear in the memory window. Press FCTN 9 to enter the memory window and simply type the new printer name over PIO and then type a space. Fifteen characters are allowed for the new printer name. This name must always be followed by a space.

After the correct printer name is specified, move the cursor to the prompt at the upper left corner of the editor screen. At the first character of the prompt, type c, g, or v to designate which type of memory to be printed. Next, move over to the Start prompt and enter the address of the beginning of the memory block to be printed. Then move to the Finish prompt and enter the address of the last location to be printed. To print the contents of GROM memory from >2000 to >23FF, specify the proper printer name and enter the parameters:

g2000, Start 2000, Finish 23FF

After the proper parameters have been entered, press FCTN 7 and the printout will begin.

Dump

FCTN 8 is used to dump the contents of a specified memory block to disk. This is useful for creating memory image files of certain blocks of memory. As with the other functions, parameters identifying the source and destination of the data must be specified before executing this command.

The default for the Dump file name is DSK1.TRIAL_DUMP. If this needs to be altered, it should be done before the other parameters are set. To do this, move the cursor to the memory location prompt in the upper left corner of the screen and enter cA01A. If necessary, use FCTN 0 and FCTN = to toggle to an ASCII display WITHOUT the BASIC Bias. DSK1.TRIAL_DUMP should now appear in the memory window. Press FCTN 9 to enter the memory window and simply type the new file name over DSK1.TRIAL_DUMP and then type a space. Fifteen characters are allowed for the entire file name. This name must always be followed by a space.

After the correct file name is specified, move the cursor to the prompt at the upper left corner of the editor screen. At the first character of the prompt, type c, g, or v to designate which type of memory to be dumped. Next, move over to the Start prompt and enter the address of the beginning of the memory block to be dumped. Then move to the Finish prompt and enter the address of the last location to be printed. To dump the contents of GROM memory from >2000 to >23FF, specify a file name and enter the parameters:

g2000, Start 2000, Finish 23FF

After the proper parameters have been entered, press FCTN 8 and the dump to the file will begin.

Back

Pressing FCTN 9 is used for moving from the parameter input fields to the memory window and back. When FCTN 9 is pressed, the cursor will move to the upper left corner of the memory window. The cursor location pointer in the lower left corner of the screen shows the current location of the cursor

in the memory block. Once in the memory block, the arrow keys (FCTN E, S, D, and X) will move around the memory window. Pressing ENTER will move the cursor back to the upper left corner of the memory window. If the memory in the window is RAM or GRAM (instead of ROM or GROM) and is not write protected, new information may be typed over the current contents of the displayed memory.

While in the memory window, FCTN O and FCTN = can also be used to change the type of memory display.

To exit the memory window, simply press FCTN 9 again and the cursor will return to the input field.

BASIC Bias

FCTN O toggles the current memory window display between normal ASCII and ASCII with a BASIC Bias. BASIC Bias is often used in modules and some assembly language programs.

When BASIC Bias is active, the ASCII data is not shown according to the Hex values that normally correspond to normal ASCII characters. Instead, all the character values are offset. The BASIC Bias compensates for the offset and displays the ASCII information. Any information altered here will also be offset by the proper amount to eliminate disparities between the displayed data and the actual data.

Pressing FCTN O again returns to the normal ASCII display.

ASCII Hex

FCTN = toggles between the ASCII display and the Hex display of the data in the memory window. When in ASCII mode, the ASCII representation of the data is displayed. When in Hex mode, the Hex data itself is available.

When altering memory, the proper display mode must be selected depending on the information to be changed.

Pressing FCTN = again returns to the ASCII display.

Arrow Keys

The arrow keys (FCTN E, S, D, and X) are used to move around both the input fields and the memory window.

When in the input fields, they may be used to move from prompt to prompt and to backspace over data. When in the memory window, they allow full movement about the data. When the cursor is moved to the top or bottom of the memory window, continuing to press the arrow key corresponding to that direction will page in more data as required. The memory location pointer in the lower left corner of the screen will display the current memory location so that the user does not have to keep track of this.

ENTER

Pressing ENTER while in the input field area will move to the beginning of the next input field. This may be used to quickly move about the various prompts. When in the memory window, pressing ENTER will return the cursor to the upper left corner of the memory window.

CTRL =

Pressing CTRL = will exit the memory editor and return to the P-GRAM menu screen.

EDITING THE P-GRAM

This section contains specific information on using the P-GRAM's memory editor to modify the contents of the P-GRAM card. Instructions for operating the memory editor itself are in the section "Using the Memory Editor". Some examples of possible module software changes are in the section "Module Modifications" and specific information on module software structure is contained in the "Technical Data" section. Although the memory editor does allow other functions of the P-GRAM and other cards to be accessed, the information in this section only addresses those functions required to edit the P-GRAM memory.

The P-GRAM is designed so that all of its functions are software controlled. The configuration of the P-GRAM is controlled by the Communications Register Unit (CRU) interface. These will be manipulated to perform the following functions:

1) Editing the DSR RAM memory

2) Editing the GRAM/module RAM memory

To control these functions, the appropriate combination of CRU bits must be controlled with the CRU Bit Select function of the editor (FCTN 1). To enter a bit code, press FCTN 1 while in the editor. Type the CRU base of the P-GRAM card and then the bit code. The codes needed to edit the P-GRAM memory are listed in the following sections. A complete listing of the CRU bit functions and bank switching is described in the "Technical Data" section of this manual.

After editing memory on the P-GRAM, the memory should be disabled before performing a different function. This may be done by using 00 as the CRU bit code. Doing this prevents accidental corruption of the P-GRAM memory or other memory. Any DSR may be disabled by using 00 as the bit code. Upon leaving the editor, all DSRs and P-GRAM memory are automatically disabled.

EDITING THE DSR MEMORY

The P-GRAM uses a 16K RAM-based operating system. To edit the P-GRAM's operating system (the DSR) must be enabled, the P-GRAM must be write enabled, and an 8K bank must be selected. The bit codes used for this are:

11=Enable lower 8K DSR bank

09=Enable upper 8K DSR bank

00=Disable DSR memory

Although other CRU bit configurations may allow the user to access the DSR memory, they should not be used since they might allow bank switching while writing to memory.

EDITING THE GRAM/MODULE RAM MEMORY

Editing the GRAM and module RAM memory on the card is the primary method for modifying modules. The P-GRAM uses the GRAM memory and 16K of bank switched RAM memory to allow emulation of modules that use bank switched memory. To edit the GRAM and module RAM on the P-GRAM card, the module RAM must be enabled and an 8K bank must be selected. The bit codes for this are:

12=Enable GRAM and lower 8K module RAM

0A=Enable GRAM and upper 8K module RAM

00=Disable GRAM and module RAM

When making any changes to the module RAM space that involves a non-bank switched module (most modules), be sure to make the same changes to BOTH upper and lower banks. Other CRU bit codes may allow access to the module RAM memory, but they should be avoided to prevent accidental bank switching while editing memory.

The P-GRAM + editing should only be done using page one to AVOID errors in saving contents of the wrong page.

To EDIT a module that you want to use in page 2,3 or 4 you must load it in page ONE & EDIT, then save to disk and reload to page 2,3,or 4.

MODULE MODIFICATIONS

This portion of the manual describes some changes to modules that can only be done with a GRAM device such as the P-GRAM card. The changes described here are only a few of the many possible modifications that may be made. The possibilities are limited only by the ability of the user to modify software.

Many of these changes were originally devised for use with the Gram Kracker, but they work just the same with the P-GRAM. These modifications were originated by a variety of individuals such as Tom Freeman, Jim Lohmeyer, Barry Traver, Rory Binkerd, Danny Michael, and others.

Since some modules were released in various versions due to minor software changes, some of the information below may not apply to certain modules. If a module does not contain information which should be present at an address given below, do not attempt to modify that module.

EDITOR/ASSEMBLER

Save the Editor/Assembler module to disk. Remove the module from the console, reset the computer, and load the module into the P-GRAM. Use the memory editor to make the following modifications. Be sure to save the modified module back to disk when finished.

The Editor/Assembler may be modified so that it will function as a Supercart module (an Editor Assembler module with 8K of extra RAM memory). To do this, change the byte at g>6003 to >A5 in Hex mode. This will allow the P-GRAM to emulate a Supercart whenever the Editor Assembler with this change is loaded. NOTE: This change sets the RAM flag in the Editor/Assembler module header so that the P-GRAM will recognize this module as having RAM memory. More information on the module RAM flag may be found in the "Technical Data" section of this manual.

To change the drive number from which the Editor/Assembler loads the Editor, Assembler, and UTIL1 files, change the byte at g>6621 to the desired drive number. This drive may be a floppy drive or a RAM-Disk. Make this change in ASCII mode.

To bypass the LOAD ASSEMBLER (Y/N) prompt, change the following bytes in Hex mode:

Address:	Change:	To:
g>6658	08 92 A0	05 66 7F

The screen colors for both the main menu and the Editor may be changed. The main menu color is located at g>652C and the Editor color is at g>6537. This will be >F5 in an unmodified module (in Hex mode). This can be changed to anything from >00 to >FF. Using >17 would result in black

characters on a cyan background.

The default name for the Utility option may be changed at g>662D. The current name is UTIL1. Since the length of the Editor and Assembler file names must match the length of this name, the UTIL1 name should only be replaced with a name having the same number of characters. Make this change in ASCII mode.

EXTENDED BASIC

Before making any changes, save the Extended BASIC module to disk, reset the console, and reload the module into the P-GRAM. After a change is made, the module should be saved back to disk.

When selected, Extended BASIC will normally search drive #1 for the file LOAD and run that file. This is known as the auto-load feature. With the following modification, this feature may be bypassed by holding down the space bar immediately after selecting Extended BASIC. NOTE: If this change is made, DO NOT specify short files when saving the modified module back to disk. To do this, make the following changes in Hex mode:

Address:	Change:	To:
g>63CD	86 A3 71	58 00 71

Next, move to g>7800 and enter:

86 A3 71 03 D6 75 20 63 D3 43 D0

The name of the auto-load file may also be changed. The file name DSK1.LOAD is located at g>6352 and should be altered in ASCII mode. Use the editor to type the new file name over the existing one. A new drive and/or different file name may be entered. The name of the new file should be the same length to avoid errors. Typing spaces over the DSK1.LOAD name will eliminate the auto-load feature completely.

The delay before the keys start to auto-repeat may be changed. The counter for this delay is located at g>6ABC and should normally be >FE. This may be changed from >00 to >FF. This smaller this number is, the faster the keys will begin to repeat. Make this change in Hex mode.

The background screen color may be changed at g>693A. This will be >07 in an unmodified module (in Hex mode). This can be changed to anything from >00 to >0F. Using >05 would result in a blue background. The character foreground and background colors for text are located at g>6948. This will be >10 in an unmodified module (in Hex mode). The first character is the foreground color and the second character is the background color. Using >F0 would result in white letters with a transparent background.

TI-WRITER

Save the TI-Writer module to disk. Remove the module from the console, reset the computer, and load the module into the P-GRAM. Use the memory editor to make the following modifications. Be sure to save the modified module back to disk when finished.

The foreign language selections may be deleted from the module selection screen by changing the byte at g>6007 from >10 to >CB.

The drive number from which the Editor and Formatter files are loaded is located at g>6763 and g>65A7. Change both of these to the desired drive number in ASCII mode.

The name of the Utility program is located at g>6B27, g>6CDO, g>6EBA, g>70A1, g>725E, g>7469, and g>763B. Only the name at g>6B27 is used in the English version. The other occurrences are for the foreign languages and do not need to be changed unless they are to be used. Make these changes in ASCII mode.

The default colors for the main menu are located at g>6284 and the colors for the Editor are at g>68C1. If the Editor color is changed, it will only affect the color that the Editor uses when first loaded. If CTRL 3 is used to change the screen colors, the color code changed in the module will not be available again until the Editor is reloaded unless the EDITA1 file is changed. The colors used by the Editor with the CTRL 3 command are located in the first sector of the EDITA1 file at bytes >F5, >F7, >F9, >FB, and >FD. Make all of these changes in Hex mode.

TI DISK MANAGER II

Save the Disk Manager II module to disk. Remove the module from the console, reset the computer, and load the module into the P-GRAM. Use the memory to make the following modifications. Be sure to save the modified module back to disk when finished.

The foreign language selections may be deleted from the module selection screen by changing the byte at g>8006 from >2A to >5B.

The maximum number of drives accessed by DM II may be increased from three to nine. This will allow it to access additional drives on double-density controllers as well as most RAM-Disks located at CRU >1000. To make this modification, change the following bytes in Hex mode:

Address:	Change:	To:
g>63F4	33	39
g>6426	33	39
g>65DC	33	39
g>675D	33	39
g>6850	33	39

g>724D	33	39
g>72C0	33	39
g>8802	33	39
g>8812	33	39
g>8DB2	33	39
g>8DC5	33	39
g>937B	33	39
g>9390	33	39

RS232 may be replaced with PIO for users who wish to have the parallel printer available as a menu option. This saves having to enter PIO each time a listing device is desired. Since selecting RS232 defaults to 300 baud, it is generally unused anyway. To add PIO as an option, move to g>61B1 and enter (in Hex mode):

03 50 49 4F 20 20

Next, move to g>86F2. Switch to ASCII mode and enter:

PARALLEL INTERFACE

The RS232 will then be replaced with PIO.

1200 BAUD TE-II

Save the TE-II module to disk. Remove the module from the console, reset the computer, and load the module into the P-GRAM. Using the memory editor, modify the following locations in GRAM (all information is in Hex):

Address:	Change:	To:
g>6F36	17 33 30	16 31 32
g>6F3B	E1 33 30	E0 31 32
g>6F4B	17 31 31	16 30 30
g>6F50	E1 31 31	E0 30 33
g>7164	E0 30 33	E0 31 32
g>7752	33 30 30 20	31 32 30 30
g>833D	31 31 30	33 30 30
g>83BC	30 33 30	31 32 30

When finished, save the P-GRAM back to disk. The TE-II module will then allow selection of 1200 or 300 baud instead of 110 or 300 baud.

PLATO RAM-DISK FIX/MODULE HEADER

The PLATO module is an auto-start module that will try to immediately execute instead of appearing as a module choice. It also does not work properly with a Horizon RAM-Disk at the >1000 CRU base. To add a module header and fix the RAM-Disk compatibility problem, turn the console off and insert the PLATO module. Hold down the "A" and "Z" keys at the same time while turning on the console. The P-GRAM menu will appear. Use the Save Module function to save the PLATO module to disk. Then remove the module, reset the computer, and load the module into the P-GRAM card from disk. Before leaving the memory, make the following changes to the

GRAM memory (all information is in Hex):

Address:	Change:	To:
g>6001	AA 81	AA 01
g>6006	00 00	97 D9
g>D772	0F 00	10 00
c>6694	0F 00	10 00

NOTE: Be sure to change the byte at c>6694 in both banks of the module RAM.

Now move to g>97D9 and enter:

00 00 60 13 05 50 4C 41 54 4F

When finished, save the P-GRAM back to disk. The module will now have a normal module header and will work with the Horizon RAM-Disk.

PIO FOR TAX/INVESTMENT RECORD KEEPING

Save the Tax/Investment module to disk. Remove the module from the console, reset the computer, and load the module into the P-GRAM. Using the memory editor, modify the following location in GRAM (all information is in Hex):

Address:	Change:	To:
g>604A	52 60	50 60

When finished, save the P-GRAM back to disk. The module will now allow use of PIO as a printer name.

TECHNICAL DATA

The P-GRAM card contains 72K of battery backed RAM memory that is divided into 40K of GRAM (GROM emulating RAM), 16K of bank switched module RAM, and 16K of bank switched DSR RAM. All memory is low power CMOS. One 8Kx8 and two 32Kx8 static RAMS are used for all memory. NOTE: The P-GRAM + card contains 192K of battery backed RAM memory that is divided into four pages of 40K of GRAM. One 128Kx8 and two 32K x 8 static rams are used for the P-GRAM+ memory.

GRAM MEMORY

The GROM emulation is accomplished by full hardware emulation of GROM memory using battery backed RAM. No software GROM emulation is used. The GRAM memory occupies the GROM addresses g>6000 through g>FFFF. This is subdivided into five 8K banks that are used to emulate the GROM memory banks usually allocated for module use (GRAMs 3 though 7). Enabling of this memory is accomplished through the CRU interface as instructed by the DSR. When enabled, the GRAM memory performs all the same functions as normal GROM memory except for the Read Address function. Since the console GROM operates with the same address counting as the GRAM, the Read Address function will retrieve the address from the console GROM and normal operation is not affected. The GRAM also has the ability to use the Write Data function. When enabled, the GRAM will respond at all GROM base addresses from c>9800 upward.

MODULE RAM MEMORY

The 16K module RAM memory occupies the 8K of CPU memory space from c>6000 to c>7FFF that is normally used for module ROM. It is divided into two 8K banks. One of these banks at a time that may be switched into the c>6000->7FFF memory space. Switching is accomplished by writing to the c>6000->7FFF memory space. Writing any data to c>6000, c>6004, c>6008, etc. will switch in the lower 8K bank. Writing to c>6002, c>6006, c>600A, etc. will switch in the upper 8K bank. Three conditions must be valid before attempting to bank switch the module RAM space: The GRAM/RAM memory must be enabled, the memory must be write protected, and the RAM space must be in bank switching mode. All of these functions are controlled by the CRU interface.

This method of bank switching is used by most modules that use bank switched ROM, including Extended BASIC.

DSR RAM MEMORY

The 16K DSR RAM memory occupies the 8K of CPU memory space from c>4000 to c>5FFF that is normally used for DSR ROM. Like the module RAM, it is divided into two 8K banks. The same method of writing to memory is used to switch 8K banks as is used for the module space RAM, except that the memory block to write to is c>4000->5FFF. Writing to c>4000 will switch in the lower 8K and writing to c>4002 will switch in the upper 8K. Three conditions must be valid before attempting to bank switch the module RAM space: The DSR RAM

space must be enabled, the memory must be write protected, and the RAM space must be in bank switching mode. All of these functions are controlled by the CRU interface.

Also note that DSR RAM bank switching occurs in unison with module RAM bank switching. Therefore, it should not be assumed that the DSR RAM has not bank switched if any data has been written to the module RAM space.

CRU BIT USAGE

The following CRU bits are used for memory control:

Bit #	Function/Bit status
0	(OFF) DSR paged out (ON) DSR paged in
1	(OFF) GRAM/RAM disabled (ON) GRAM/RAM enabled
2	(OFF) P-GRAM write enabled (ON) P-GRAM write protected
3	(OFF) Low 8K locked in (ON) Bank switching enabled
4	(OFF) High 8K locked in (ON) Bank switching enabled

NOTE: For normal bank switching mode, bit 3 and bit 4 should both be ON. One bank may be locked in while accessing that memory bank by turning the corresponding bit OFF. The locking in of banks applies to both the bank switched DSR RAM and the module RAM. It is not recommended that both bits be turned off simultaneously during direct access to either module RAM or DSR RAM. However, these CRU bits may assume any configuration at other times.

The CRU bits are set with the assembly language SBO instruction and reset with the SBZ instruction. In order to write to any memory (GRAM, module RAM, or DSR RAM), the write protect bit must be reset. To write to DSR RAM, the page bit must be set and the corresponding bank switching bit must be reset. To write to module RAM, the GRAM/RAM enable bit must be set and the corresponding bank switching bit must be reset. To write to GRAM, the GRAM/RAM enable bit must be set. As mentioned above, do not reset both bank switching bits while accessing DSR or module RAM.

Further information concerning direct access to the RAM memory on the P-GRAM may be found in the section "Editing the P-GRAM".

REAL-TIME CLOCK

The clock option uses an MM58167A real-time clock chip. This chip supports counters for thousandths of a second, hundredths of a second, seconds, minutes, hours, day of week, day of month, and month. These functions occupy eight registers in the clock. The MM58167A chip does support other functions controlled by additional registers, but they are not supported on the P-GRAM card.

All information to and from the clock must be in BCD format. The conversion from decimal to BCD is:

BCD=DECIMAL+6*INT(DECIMAL/10)

The conversion from BCD to decimal is:

DECIMAL=BCD-6*INT(BCD/16)

To set a register, simply write the desired data to that register. Information from the clock is simply read from the registers. The addresses for the registers are:

c>8640 = 1/1000 Second
c>8642 = 1/100 Second
c>8644 = Seconds
c>8646 = Minutes
c>8648 = Hours
c>864A = Day of Week
c>864C = Day of Month
c>864E = Month

The day of week is numbered from 1 to 7 (Sunday=1). The hours are in 24-hour format. A register for the year is not available on the MM58167A. The year counter on the P-GRAM clock is accomplished by saving the current month and year to the DSR RAM during the DSR power up routine. Before doing so, however, the clock is read and the month is compared to the month stored in the DSR RAM. If the month has changed from 12 to 1, the year is incremented and both the year and current month are saved back to DSR RAM. The year may be read directly from the P-GRAM by enabling the DSR and reading the year from c>4080 in the lower 8K bank of the DSR.

After the registers are written to, the sound processor must be reset. This is required since the sound processor (which is a write-only device) shares memory locations with the clock chip. Accessing the sound chip at the regular c>8400 address does not affect the clock.

MODULE RAM FLAG

To allow proper bank switching of module RAM, the DSR automatically invokes write protection and enables bank switching for the P-GRAM memory. This presents a small problem when attempting to emulate a module that normally contains RAM memory in the space from c>6000 to c>7FFF (such as the Minimem or Supercart modules) because the write

protection prevents writing to RAM memory. Therefore, a method has been provided to allow these modules to be used normally and allow the user to designate any module as having RAM memory instead of normal ROM memory. This is known as the RAM flag.

During power up following a reset of the console, the P-GRAM's operating system first checks the console for a module to see if the P-GRAM should be enabled. If so, the operating system then checks the fourth byte of each module header for the byte >A5. If found, the operating system will lock in the lower 8K bank of the module RAM memory and write enable all module memory. This way, any module containing the >A5 byte in the fourth byte of any module header will be treated as a RAM module. The fourth byte is not used for data in a normal module header, so this byte was chosen to prevent possible compatibility problems.

When module is saved to disk, the operating system automatically checks every 400 bytes of the c>6000 to c>7FFF module space to see if RAM is present by attempting to alter then restore a byte of memory. If any portion of the module responds as RAM, the fourth byte of the RAM space header will be saved as >A5. Since all headers are checked on power up, only one module header must contain the >A5 byte to designate the module as having RAM memory.

Using this method, Editor/Assembler may be made to function as a Supercart module (an Editor/Assembler module with 8K of RAM from c>6000 to c>7FFF). This is done by loading an Editor/Assembler module into the P-GRAM, altering the byte at g>6003 to read >A5, and saving the module back to disk. Whenever the modified Editor/Assembler is loaded, the user will then be able to perform all functions normally provided by a Supercart.

MODULE FILE FORMAT

The P-GRAM saves files to disk in memory image (PROGRAM) format. Each bank of GRAM or RAM used by the module is saved as a separate file. Each file begins with six bytes which provide the loader with data on where to place the module in memory, the length of the file, and whether or not more files are to be loaded. This is followed by the actual data to be loaded into the GRAM or RAM memory banks.

The files are saved and loaded beginning with the upper 8K RAM bank, followed by the lower 8K RAM bank, then the GRAM banks starting with the highest numbered bank. If the module file only has one RAM bank, it is loaded into both the upper and lower banks automatically. Unused banks are not saved or loaded. The arrangement of the first six bytes in each file is as follows:

Byte(s):	Meaning:
>00	Load flag FF=More files to be loaded

00=No more files to load

>01 Bank indicator
04=GRAM 3 A000
05=GRAM 4 8000
06=GRAM 5 A000
07=GRAM 6 C000
08=GRAM 7 E000
09=Lower 8K RAM bank 6000
0A=Upper 8K RAM bank

>02->03 Number of bytes to load

>04->05 Address of where to start loading file

MODULE HEADER FORMAT
==========

The module header tells the TI-99/4A's operating system what to use for a module name and how to access the module. This information may be found at the beginning of one or more of the seven GRAM or RAM banks. The table below shows information based on a header at g>6000, but can also apply to GRAM banks at g>8000, g>A000, g>C000, g>E000, or the RAM bank at c>6000.

Byte(s):	Meaning:
g>6000	Validation byte AA=Check bank for header Any other=No header
g>6001	Version number Greater than >7F=Foreign language or auto-start module
g>6002	Number of application programs (not currently used)
g>6003	RAM flag (not normally used) A5=Write enable and lock in lower 8K module RAM Any other=No RAM, enable bank switching
g>6004	Address of power up header
g>6006	Address of module name header
g>6008	Address of module DSR header
g>600A	Address of subprogram header (used by CALL statements)
g>600C	Address of interrupt header (not used in modules)
g>600E	Not used

The two bytes at g>6006 tell the computer where to look

for the module name and where the module begins execution. At the address listed at g>6006, there will be another address for any other module header that exist. If no other headers are used, this address is >0000. The next two bytes following this address are where to begin execution of code for that header. The next byte is a string length counter and is followed by the text used for the header. Multiple headers may be chained together this way provide various entry points into the module memory space.

Further information on module headers and module software may be found in Texas Instruments' GPL manual.

CALLS AND SUBROUTINES

The P-GRAM operating system adds several new BASIC CALLs to the system. All of these were provided to allow various P-GRAM functions to be accessed from BASIC. There are also several subroutines that may be accessed by machine language programs which may be used to control the P-GRAM from a program. The BASIC CALLs are:

CALL PTIME - Provides display and setting of the clock

CALL PG - Load the P-GRAM menu

CALL PGON - Forces the P-GRAM ON

CALL PGOF - Forces the P-GRAM OFF

CALL PGZAP - Destroys any module headers in GRAM/RAM

PTIME and PG both cause programs that are resident in DSR bank 1 to be downloaded to memory expansion and executed. These are the most useful of the BASIC CALLs.

PGON and PGOF were mainly intended for use during development of the loader and editor packages and will seldom be used.

PGZAP can be used in the event that a module becomes corrupted and attempts to gain control of the system. PGZAP will change the >AA validation byte at the beginning of each module header to >00. This prevents the computer from recognizing the module headers and therefore ignores them. This allows the user to later restore the P-GRAM contents by restoring this byte to >AA.

Subroutines that are accessible from a running machine-language program are identified as >31 through >37. These may be accessed via DSRLNK just like the disk controller subroutines except, since each is a special-purpose subroutine, there are no arguments to be passed. If a program does not already need a DSRLNK for some other reason, the programmer may include the much shorter and faster SUBSRH (subroutine search) that is included here. As an alternative, the programmer may decide to manipulate the CRU bits directly. If this is done, care must be taken to

ensure that CRU base address used corresponds to that of the P-GRAM. The subroutines are:

>31 - CRU-Select GRAM and RAM bank 1

>32 - CRU-Select GRAM and RAM bank 2

>33 - CRU-Select GRAM and invoke write protection

>34 - Turn the P-GRAM OFF

>35 - Perform power-up checks

>36 - Download and execute the P-GRAM loader

>37 - Download and execute the memory editor

The following code segment will turn on the card, write enable the memory, and enable GRAM and RAM bank 1.

```
      .
      .
LI    R4,>0131
STWP  R1
AI    R1,8
LI    R0,>1000
MOV   R0,@>8356
LI    R2,2
BLWP  @VMBW
BLWP  @DSRLNK
DATA  10
      .
      .
```

If desired, a program can "remember" the P-GRAM's CRU base address so that future memory manipulation may be accomplished by setting and resetting CRU bits. This may be done by saving the CRU base address immediately following the access via DSRLNK.

SUBROUTINE SEARCH

The following routine may be included in a program for accessing P-GRAM subroutines if a DSRLNK is not needed for some other reason. The example shown will cause the P-GRAM loader to be downloaded and executed. Any of the other subroutines can be accessed by changing the subroutine number.

```
* SUBROUTINE SEARCH *

GPLWS   EQU  >83E0
H20     BYTE >20
HAA     BYTE >AA

        EVEN
SUBWS   BSS  32

SUBSRH DATA SUBWS,$+2
       MOV  *14+,@GPLWS+10     Get Subroutine ID
       SZCB @H20,15            Clear Equal Bit
       LWPI GPLWS              Use GPL WS for Search
       LI   12,>0F00           Init CRU Base
SS1
       MOV  12,12              Base = Zero ?
       JEQ  SS2                Yes, Forget it
       SBZ  0                  Else Turn it Off
SS2
       AI   12,>0100           Next Device
       CLR  @>83D0             Clear Old Device Base
       CI   12,>2000           Out of Peripherals ?
       JEQ  NFG                Yes, Not Found Good
       MOV  12,@>83D0          No, Save This Base
       SBO  0                  Turn the Card On
       CB   @>4000,@HAA        Valid DSR ?
       JNE  SS1                No, Next Card
       LI   2,>400A            Yes, Offset to Subr Chain
       JMP  SS4                Check it Out
SS3
       MOV  @>83D2,2
       SBO  0
SS4
       MOV  *2,2               Get Pntr to Next in Chain
       JEQ  SS1                Zero, No More
       MOV  2,@>83D2           Save Pntr to Next
       INCT 2                  Get Entry Pntr
       MOV  *2+,9              Save it
       CB   *2+,5              ID Match ?
       JNE  SS3                No
       SWPB 5                  Second Byte of ID
       CB   *2+,5              Match ?
       JEQ  SS5                Yes
       SWPB 5                  No, Restore ID Order
       JMP  SS3                Try Some More
SS5
       BL   *9                 Go to the Routine
       JMP  SS3                Card Didn't Like it!
       SBZ  0                  Good Return. Turn Card Off
       LWPI SUBWS              Use Own Workspace
       RTWP                    Return
NFG
       SBZ  0                  Card Off
       LWPI SUBWS
       SOCB @H20,15            Set Equ Bit
       RTWP                    Return
```

```
* This Calls the P-GRAM Loader *

        DEF   TRYIT

SUBR#1  EQU   >0131
SUBR#2  EQU   >0132
SUBR#3  EQU   >0133
SUBR#4  EQU   >0134
SUBR#5  EQU   >0135
SUBR#6  EQU   >0136
SUBR#7  EQU   >0137

WS      BSS   32

  TRYIT  EQU   $
         BLWP @SUBSRH        Search for:
         DATA SUBR#6         Subroutine 6: D/L P-Gram Loader
         END
```

DSR SOURCE FILES
==========

The following files make up the P-GRAM DSR:

Bank 1 - P-GRAM Loader & PTIME Utility:

PGDSR/SRC2 - Main Source, with the following copy files:

MAINDSR - The control program
GRMLODR/S1 - GRAM loader/saver
GRMLODR/S2 - GRAM loader/saver
GRMLODR/S3 - GRAM loader/saver
GRMLODR/S4 - GRAM loader/saver
GRMLODR/S5 - GRAM loader/saver
SETDT/TIME - PTIME
SUBR_NOTES

Bank 2 - Memory Editor:

PGDSR/SRC1 - Main Source, with the following Copy files:

MAINSR - Same file as above
PGED/S1 - Memory editor
PGED/S2 - Memory editor
PGED/S3 - Memory editor
PGED/S4 - Memory editor

The DSR loader (from which UTIL1 is produced) has only five source files:

DSRLDR/SRC - Main source with COPY directives for:

DSRLODR/S1
DSRLODR/S2
GRMLODR/S4
INITV/S

Note that the DSR loader is AORGed in low memory expansion, therefore the SAVE utility that is supplied with the E/A package from TI will NOT correctly create UTIL1. A custom utility must be used instead.

Since the resident DSR (MAINDSR1 and MAINDSR2) must be the same in both banks to prevent locking up the computer, both banks must be reassembled following any change to the DSR. The DSR may be saved to disk in memory image format by loading the object files with the DSR loader and saving them back in memory image format.

P-GRAM+

GROM EMULATOR AND

REAL-TIME CLOCK CARD

CONSTRUCTION GUIDE

PRODUCED BY HORIZON COMPUTER

DISTRIBUTED BY BUD MILLS SERVICES

Manual by John P. Guion

(C) Copyright 1988, Bud Mills Services

DISCLAIMER

The consumer assumes full risk and liability for direct or consequential damages arising from attempted construction of the P-GRAM card.

EXCLUSION OF WARRANTIES: The P-GRAM circuit board is provided on an AS IS basis. No warranty of any kind is assumed by Horizon Computer, Limited. The user assumes full responsibility for quality of all parts associated with construction of the P-GRAM. Bud Mills Services does not recommend or endorse the quality of parts sold by any other party. In any case, Bud Mills Services shall be liable only for the cost of the circuit board, associated manuals, disk-based software, or parts, only if purchased from Bud Mills Services.

Fully constructed P-GRAM cards are available with a 6-month limited warranty for an additional cost covering parts and labor. Contact Bud Mills Services for current list of dealers or builders.

NOTICE: The contents of this manual may not be copied in whole or part for distribution without written permission from Bud Mills Services.

Prior familiarity with construction of digital circuits is assumed. Read all construction suggestions and notes provided with the construction diagrams before proceeding. The following pages show progressive stages of assembly for the P-GRAM. If you encounter a problem or have a question at any step, DO NOT proceed until the problem is resolved. If you have any questions, contact Bud Mills at (419) 385-5946.

The components used in the construction of the P-GRAM card are static and heat sensitive devices. Although we have not seen any component failures occur under normal handling procedures, care should be taken to avoid excessive static and heat transfer to the P-GRAM components. Most important, however, is that all solder connections are of good integrity.

Use a low wattage (about 25 watts) soldering PENCIL and fine 60:40 tin/lead solder. DO NOT use a soldering gun or acid core solder. Make sure that sufficient solder is supplied to all connections with good wet-out, but that there are no solder bridges between connections. Upon completion of all soldering, remove flux from the solder side of the board with a flux removing solvent.

When inserting chips, bend the pins to fit the socket by placing the chip on its side on a flat surface. Bend the pins against the surface by moving the body of the chip. Make sure all pins are properly aligned with the socket holes and that all pins actually go into the sockets upon insertion.

This manual describes the procedure for constructing the P-GRAM card. The diagrams show installation of components required for BOTH the P-GRAM and the optional real-time clock. If the real-time clock is not being installed, some of the components shown in the diagrams will not be installed on the circuit board. For this reason, be sure to read ALL of the text accompanying the diagrams which describe installation of the clock option.

Due to parts availability, certain components (particularly memory chips) may have varying part numbers. Therefore, some references to components may list more than one part number. If you cannot find a component that matches the description given in this manual, contact Bud Mills Services BEFORE proceeding. You should have the following parts:

(1) P-GRAM circuit board

(1) 74LS367
(1) 74LS259
(1) 74LS245
(3) 74LS244
(4) 74LS161A
(1) 74LS139
(4) 74LS138
(1) 74LS74
(2) 74LS32
(1) 74LS21
(1) 74LS08
(1) 74LS00
(2) 43256C, 62256LP, M5M5256, or equivalent 32Kx8 low power RAMs
(1) 4364C, 6264LP, M5M5165P, or equivalent 8Kx8 low power RAM

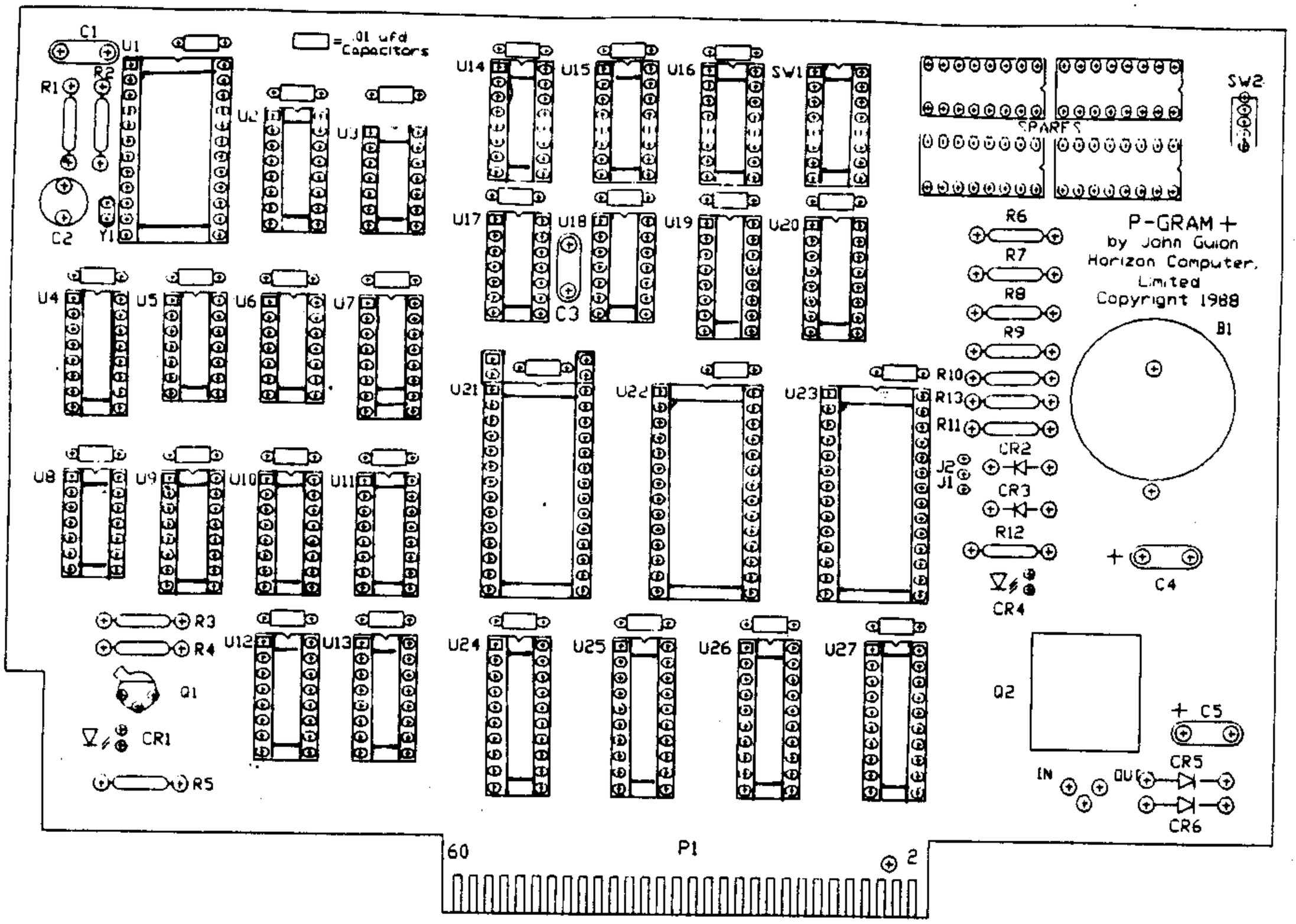
(25) .1 µF or .01 µF bypass capacitors
(1) 4.7K 1/4 Watt resistor
(3) 2.2K 1/4 Watt resistors
(4) 1K 1/4 Watt resistors
(1) 470 ohm 1/4 Watt resistor
(1) 100 ohm 1/4 Watt resistor
(3) 1N914 diodes
(1) 1N4001 diode
(2) Jumbo LEDs (GREEN only)
(1) 2N3904 or 2N2222 transistor
(1) 2.2 µF 16V tantalum capacitor (1.0 to 3.3 µF acceptable)
(1) 47 pF capacitor
(1) 7805 or LM340T5 voltage regulator
(1) 47 µF or 100 µF 16V electrolytic or tantalum capacitor
(1) BR2325 3V lithium battery
(1) Coin-type holder for BR2325
(2) SPST PCB-mount switch
(1) Heatsink for regulator
(1) Machine screw and nut combination for heatsink
(1) 8-position DIP switch
(3) 28 pin sockets
(4) 20 pin sockets
(12) 16 pin sockets
(6) 14 pin sockets

The following components will also be required if the real-time clock option is being installed:

(1) MM58167A clock chip
(2) 74LS138
(1) 32.768 KHz crystal
(1) 200K 1/4 Watt resistor
(1) 20 pF capacitor
(1) 3-30 pF or 6.8-45 pF trimmer capacitor
(3) .01 µF bypass capacitors
(1) 20 Meg 1/4 Watt resistor
(1) 24 pin socket
(2) 16 pin sockets

Once all the parts are accounted for, you may begin the construction procedure outlined in the following pages. Once construction is completed, DO NOT attempt to use the P-GRAM card until the testing procedure described at the end of this manual has been successfully completed. Failure to do so could result in severe damage to the P-GRAM card and your system.

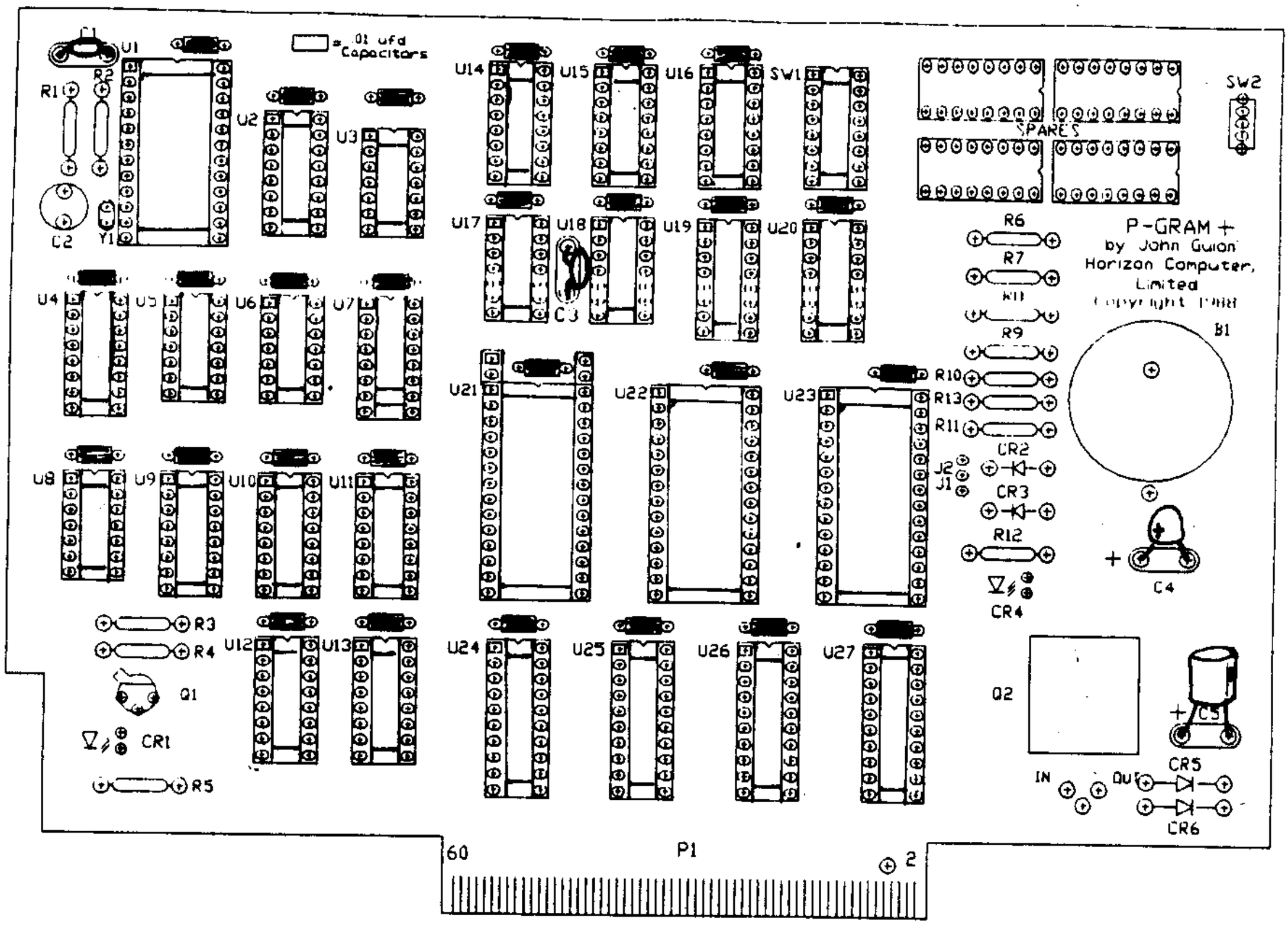
Once the P-GRAM card has been successfully assembled and tested, it is ready for use. Instructions for using the P-GRAM card can be found in the P-GRAM operating manual as well as technical information concerning both hardware and software aspects of the P-GRAM.



INSTALLING THE SOCKETS

Sockets should be used for all integrated circuits (chips) and the DIP switch on the P-GRAM card. The sockets will be soldered into the spaces outlined on the board. These spaces are numbered U1 through U27 and SW1 on the circuit board. Sockets will NOT be installed in the four sockets marked "SPARES". Also, DO NOT install sockets for U1, U15, or U16 if the optional real-time clock not being installed. All other socket positions should be filled with sockets containing the same number of pins as the space outlined on the board.

Place sockets in the outlined spaces with the notches on the sockets lined up with the notches on the outlines. Make sure that the number of pins on each socket matches the number of outlined pins. After placing the sockets in each required position, use a sheet of cardboard or other suitable material to hold the sockets in place while you turn the board over. Next, solder the corner pins on each socket to the board. Once this is done, turn the board over again and check that all sockets are seated close to the board before soldering the remaining pins. Be careful not to use too much solder or the excess solder will flow into the socket and short the pins together.

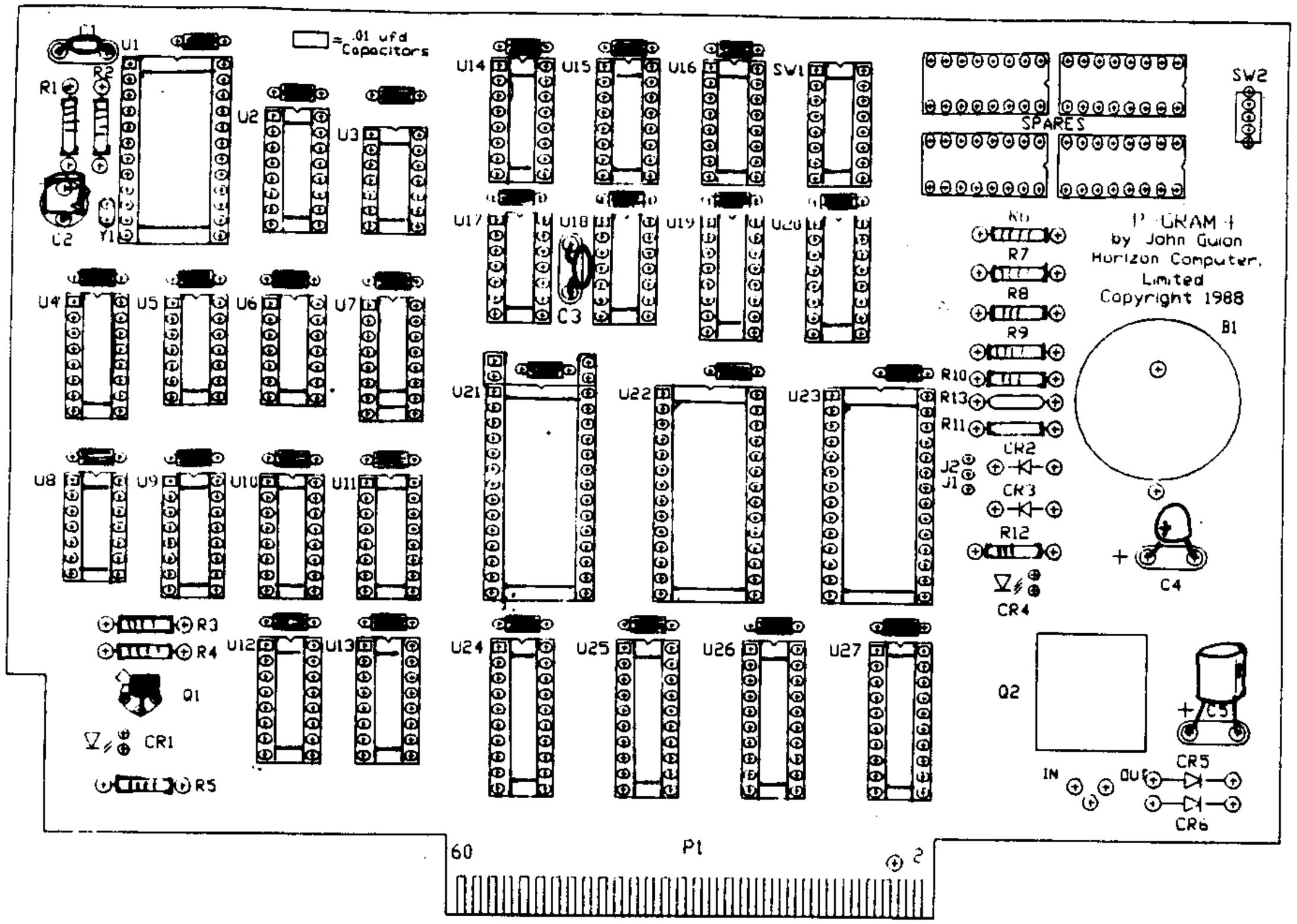


INSTALLING THE CAPACITORS

Above each chip position (U1 through U27), there is a space in which a .1 µF or .01 µF bypass capacitor should be installed. If the real-time clock option is not being installed, these will not be needed for the U1, U15, or U16 chip positions. These bypass capacitors may be either ceramic or glass-bodied devices and should have ".01", ".1", "104", or "103" marked on them. Install each required bypass capacitor and solder both leads to the board. Next, solder the 47 pF capacitor in the space marked C3.

Install the 2.2 µF tantalum capacitor (1.0 to 3.3 µF acceptable) in the position marked C4. This capacitor will have a dot or + sign that MUST be placed in the hole marked with the + sign. Install a 47 µF or 100 µF capacitor in the C5 position. This capacitor will have one lead marked with either a + or - sign. If one lead is marked +, that lead should be placed in the hole marked with the + sign. If it has a lead marked -, it should go AWAY from the + sign.

If the real-time clock option is being installed, also solder a 20 pF capacitor in the position marked C1 and the trimmer (adjustable) capacitor in the position marked C2. These capacitors may be installed in either direction.



INSTALLING THE TRANSISTOR AND RESISTORS

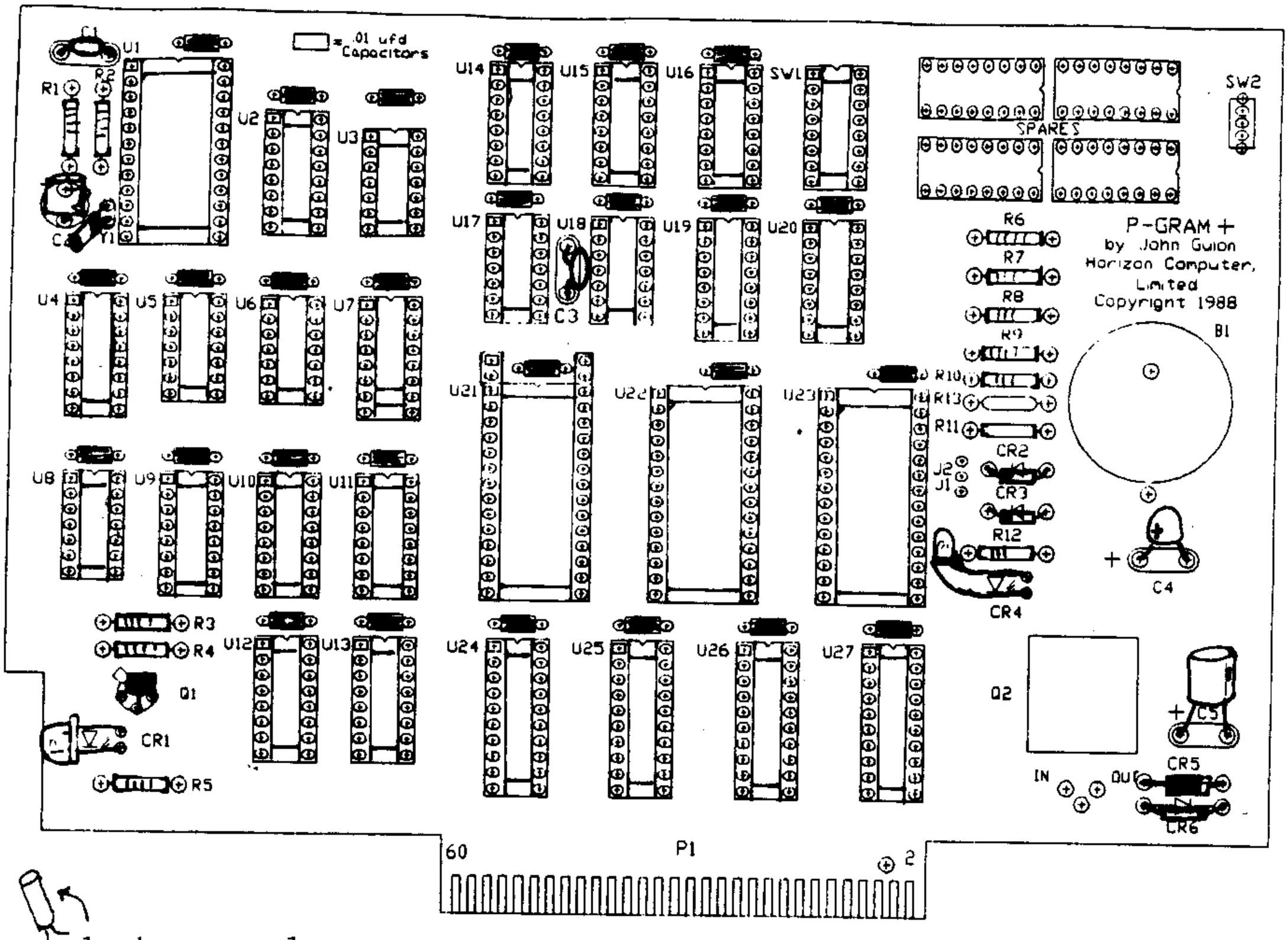
Install a 2N3904 or 2N2222 transistor in the space marked Q1. The transistor will have a tab or flat side that should line up with the markings on the board. Solder the leads so that the body of the transistor is about 1/4" from the board.

Install the following resistors in the spaces R3 through R12 (they may be identified by the bands on each resistor):

R3 - 2.2K (red, red, red)
R4 - 2.2K (red, red, red)
R5 - 100 ohm (brown, black, brown)
R6 - 2.2K (red, red, red)
R7 - 1K (brown, black, red)
R8 - 1K (brown, black, red)
R9 - 1K (brown, black, red)
R10 - 4.7K (yellow, violet, red)
R11 - 470 ohm (yellow, violet, brown)
R12 - 1K (brown, black, red)
R13 - 1K (brown, black, red) ONLY for P-Gram +192k

If the optional real-time clock is being installed, also install the following resistors at R1 and R2:

R1 - 200K (red, black, yellow)
R2 - 20 Meg (red, black, blue)



clock crystal
for Y1

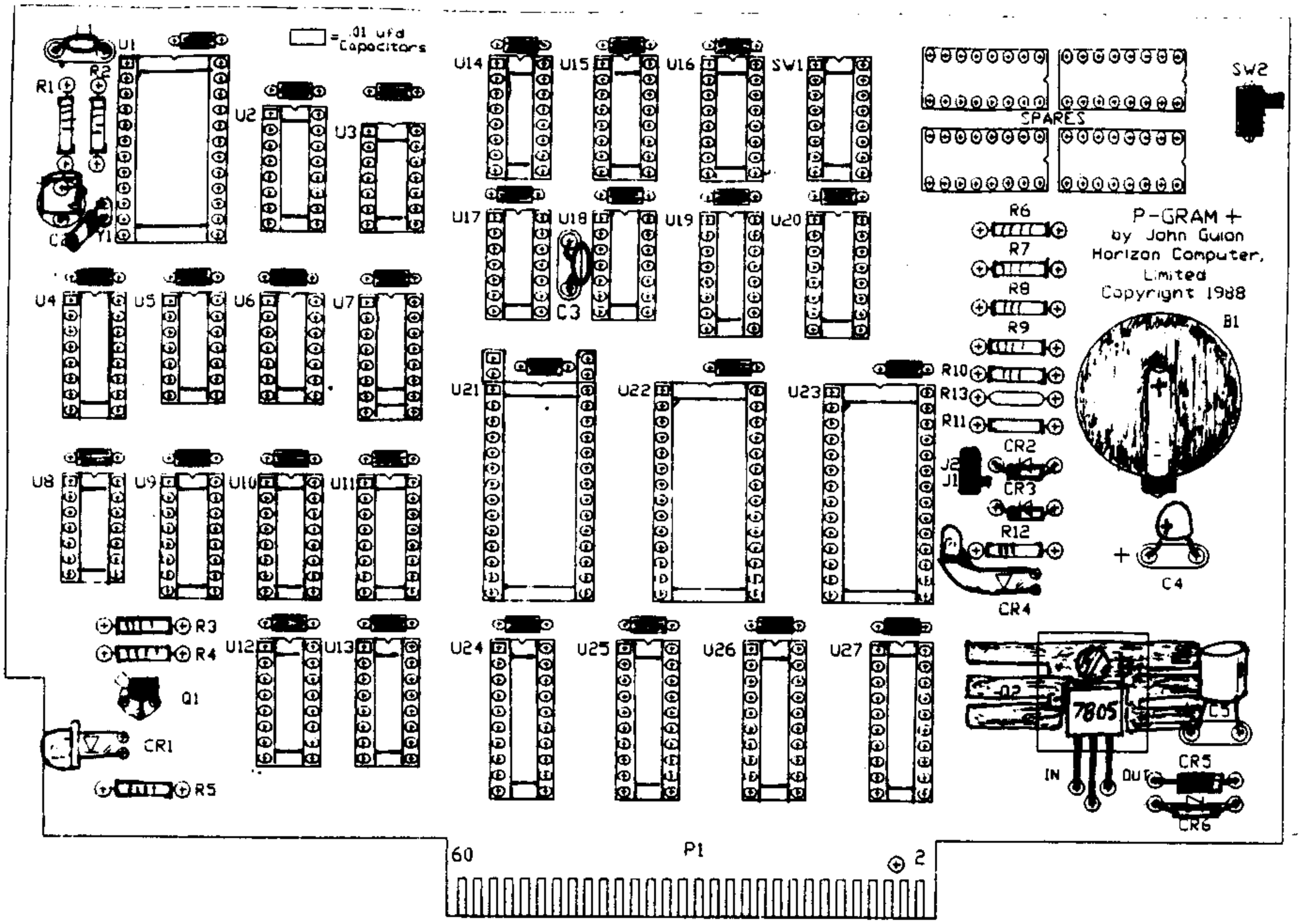
INSTALLING THE DIODES AND CLOCK CRYSTAL

Diodes will be installed in the spaces marked CR1 through CR6. These components must be installed in the direction marked on the board to allow proper function of the P-GRAM. Each diode position is marked with a bar and an arrow. The bar that the arrow points to is the CATHODE. NOTE: The clock crystal and diodes are heat-sensitive components. Avoid over heating these devices.

CR1 and CR4 are green LEDs. On these components, the cathode is the shortest lead and is also on the side of the LED that is flat. Install CR1 so that the front of the LED is even with the edge of the circuit board, but does not extend past it. Install CR4 and bend the leads so that the body of the LED is against the board. pointing upward.

CR2, CR3, and CR6 are 1N914 diodes. The cathode on these diodes is on the end of the diode that has the BLACK band around it. CR5 is a 1N4001 diode that also has a SILVER band around the cathode end. Install these diodes in the appropriate positions.

If installing the optional real-time clock, install the 32.768 KHz clock crystal in the space marked Y1. This component may be installed in either direction.



INSTALLING THE SWITCH AND POWER SUPPLY

Solder the small single-pole single-throw (SPST) switch in the space marked SW2 at the back of the card. Do not overheat this switch since it contains plastic components. Cut tabs off of remaining Mini Switch & install in J1 & J2 beside CR2.

This switch is to option the power for P Gram 92K or P Gram + 192K

Down J1 Up J2

The 7805 (or LM340T5) 5-volt regulator will be installed in the three holes below the space marked Q2 (two of these holes are marked IN and OUT). Bend the leads of this device so that the leads will fit into the holes when the metal tab on the 7805 is lined up with the hole in the outline on the board. The tab should be against the board and the printing on the 7805 should be facing upward. Once the leads have been bent to fit, solder them in the holes. Next, install the metal heat sink on top of the 7805's tab as shown above using a machine screw and nut.

The battery holder should be installed WITHOUT the battery. The holder will fit into the two holes above C4. The battery holder will have one tab (the upper tab) marked with a + sign. This should be installed in the hole next to CR2. The other tab will be installed in the hole near the back edge of the card. The circuit board may have a circle denoting the battery (B1). This circle only shows the relative position of the battery on the card and may not line up with the actual holder when installed.

PRELIMINARY TEST

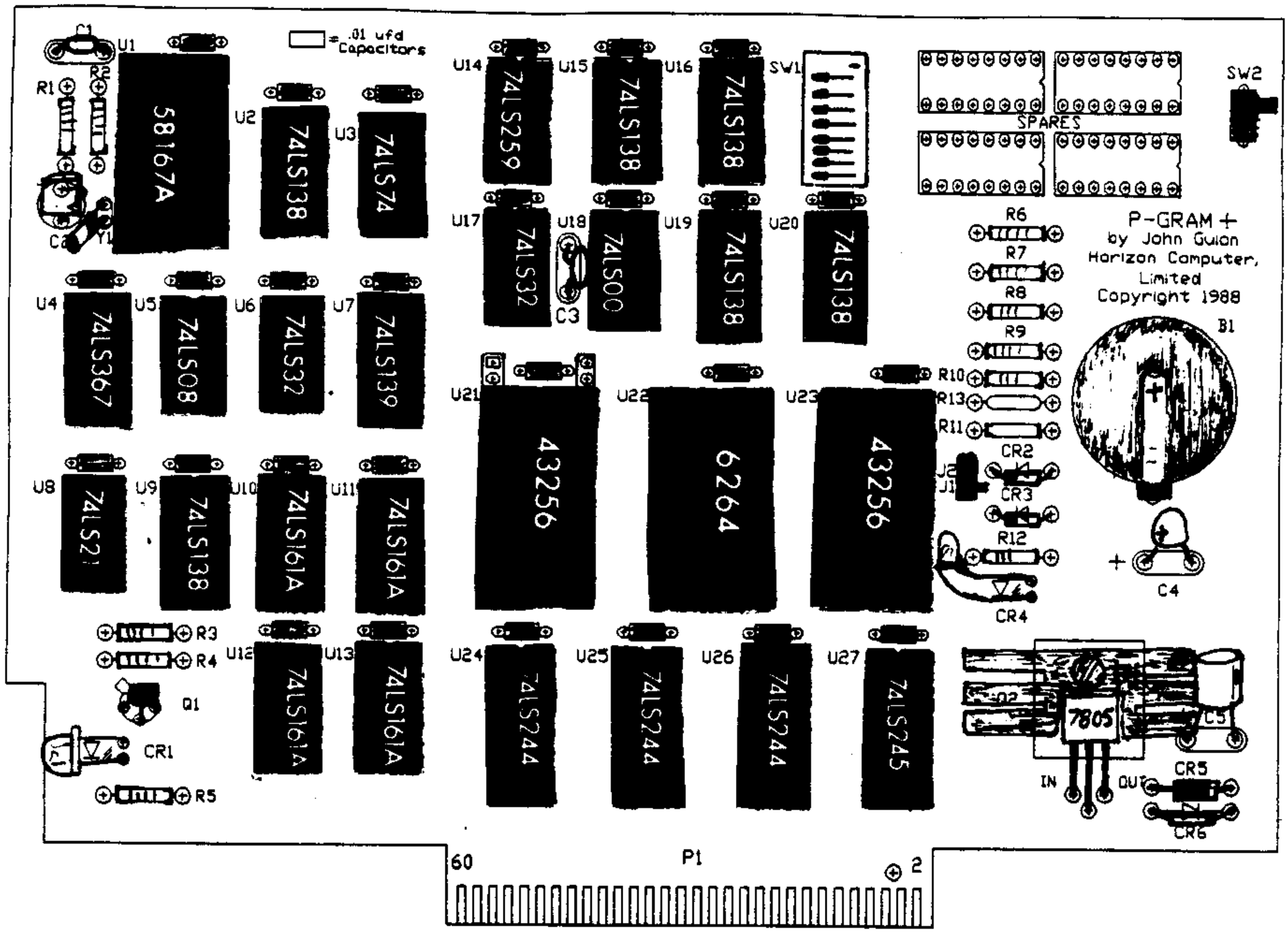
IMPORTANT: DO NOT INSERT ANY CHIPS UNTIL AFTER THIS TEST!

Use a flux-removing solvent to clean the solder flux from the back side of the P-GRAM card (this can be obtained from Radio Shack or other electronic parts distributors).

Under good light (and magnification, if available), visually inspect all solder connections for good flow and contact. Be sure that no unwanted bridges exist between adjacent solder pads or between circuit traces. Also inspect the board for loose traces of solder or other debris that could cause unwanted shorts on the board. This is particularly important in the area of the voltage regulator and the connector on the bottom edge of the card.

Next, turn off your computer system and wait two minutes. Remove EVERY card from the Peripheral Expansion System (PEB or P-Box). Do not leave any cards, including the flex-cable interface card, in the PEB. Install the BR2325 coin-type battery in the holder with the + side of the battery towards to the + lead on the battery holder (upward). Insert the P-GRAM card into the PEB and turn on the power to the PEB. The CR4 LED near the battery holder should glow. Leave the power turned on for two full minutes, checking that the LED remains lit.

If the LED has not gone out after two minutes, turn off the PEB and remove the P-GRAM. Remove the battery from its holder. If the LED failed to glow or went out, or if the battery is warm to the touch, inspect your construction, especially R11, CR2 through CR6, C5, and Q2. If none of these problems have occurred, you may begin installing the chips.



INSTALLING THE DIP SWITCH AND CHIPS

Install the 8-position DIP switch in the socket marked SW1. The end of the switch with the first switch position should be lined up with the notched end of the socket.

Install the chips in the sockets making sure that none of the leads are bent under or away from the socket and that the notches on the chips line up with the notches on the sockets. Be careful when handling these devices since they are sensitive to static electricity. The battery should NOT be installed when inserting the chips. If you are installing the optional real-time clock, you will need to install chips at U1, U15, and U16 as shown above in addition to all other chips shown. If the clock is not being installed, no chips will be used in these positions. The chips are as follows:

U1=58167A
U2=74LS138
U3=74LS74
U4=74LS367
U5=74LS08
U6=74LS32
U7=74LS139
U8=74LS21
U9=74LS138
U10=74LS161A
U11=74LS161A
U12=74LS161A
U13=74LS161A
U14=74LS259
U15=74LS138
U16=74LS138
U17=74LS32
U18=74LS00
U19=74LS138
U20=74LS138
U21=43256
U22=6264
U23=43256
U24=74LS244
U25=74LS244
U26=74LS244
U27=74LS245

For the P-Gram +
U21 will be a 128x8 module
U22 = 43256 or equivalent
R13 must be installed
J2 must be closed

TESTING THE P-GRAM

Once the construction and preliminary test described in the previous pages is completed, the P-GRAM is ready for one last set of tests before using.

Install ONLY the flex-cable interface, 32K memory, and disk controller cards into the Peripheral Expansion System (PEB). Do not install any other cards. These are the only devices required to test the functions of the P-GRAM card. Other cards are not needed and should not be installed because a serious failure of any card can damage other cards as well. This method of testing is recommended whenever installing a new or modified device in the PEB.

Set the CRU address of the P-GRAM to >1700 by turning the seventh switch on SW1 ON (seventh from the top). Leave the other seven switches OFF. With all the required components and chips installed, install the battery as described earlier. Move the SW2 switch near the back of the P-GRAM to the UP position to enable the P-GRAM. Install the P-GRAM in any empty slot in the PEB.

Turn on the PEB and then the console. Insert the PGSYSTEM disk included with the P-GRAM into drive #1. Use Editor/ Assembler option #5 to run the program DSK1.PGTEST1. This program will test all memory locations and switching on the P-GRAM card. Follow the instructions given in the program once it is loaded and enter 1700 when it asks for the CRU base. Since the module must be removed after loading, this program may take multiple attempts to run. If the program fails to run, try reinserting the module and loading it again. This test will take a few minutes to execute.

When the PGTEST1 program is finished, turn off the system and wait several minutes. Next, turn on the PEB and console again and use Editor/Assembler option #5 to run the PGTEST2 program. Also remove the module and enter 1700 for the CRU base with this program. This program reads the data in the P-GRAM card that was left by the PGTEST1 program to check the battery backing circuit. If either test fails, inspect all work and resolve the problem BEFORE attempting to use the P-GRAM card.

If both of these tests are completed successfully, the P-GRAM card is finished and operating properly. Consult the P-GRAM operating manual for information on selecting a CRU base before installing other cards. The operating manual also contains complete instructions for use of the P-GRAM card.

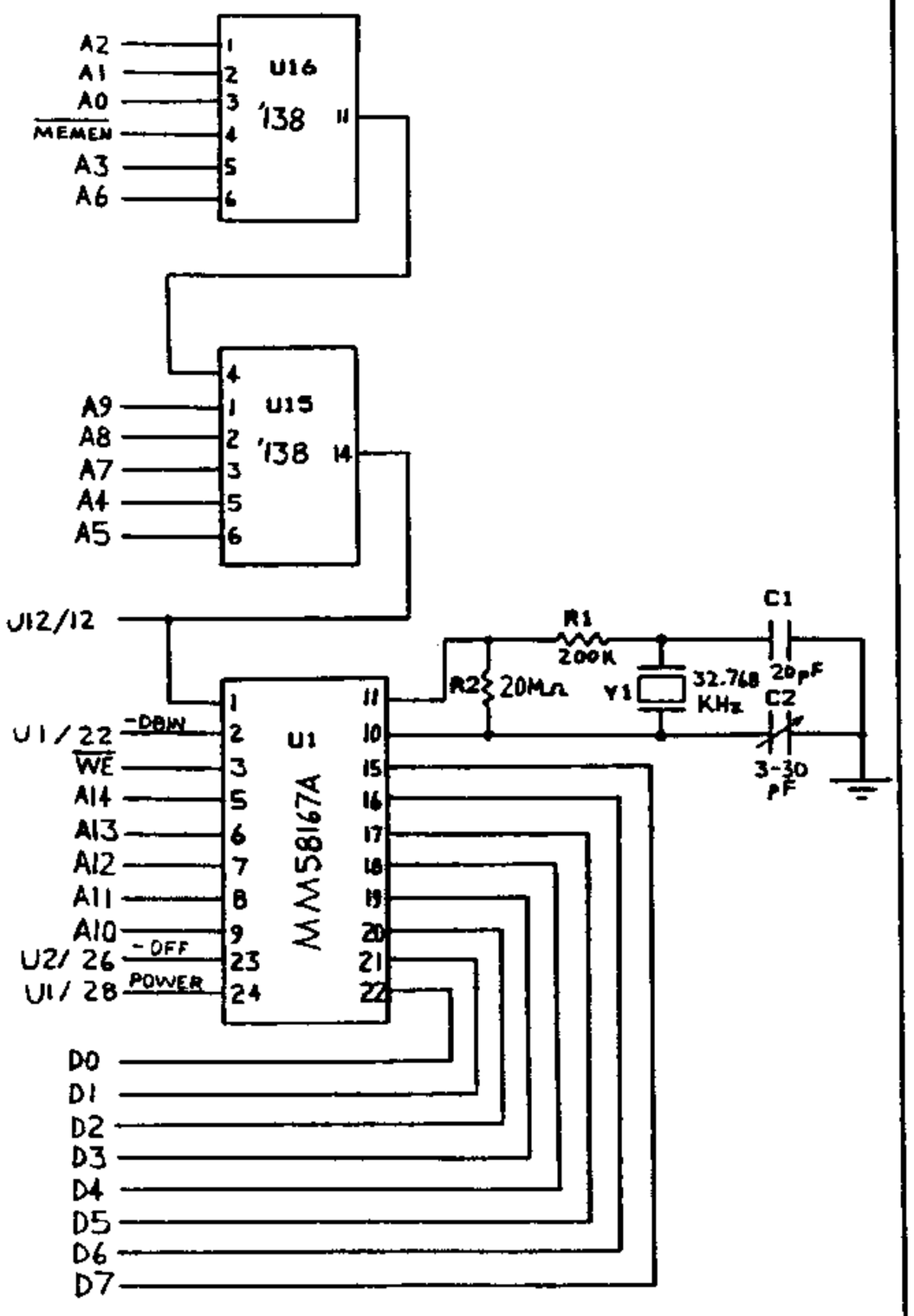
REAL-TIME CLOCK ADJUSTMENT

If the optional real-time clock appears to gain or lose several seconds each day, the clock's speed may be adjusted by using a small screwdriver to turn the trimmer capacitor (C2). Since the amount of adjustment necessary varies with each different clock chip, this must be done on a trail-and-error basis until acceptable accuracy is obtained.

P-GRAM Clock

For P-Box GROM Emulator Card

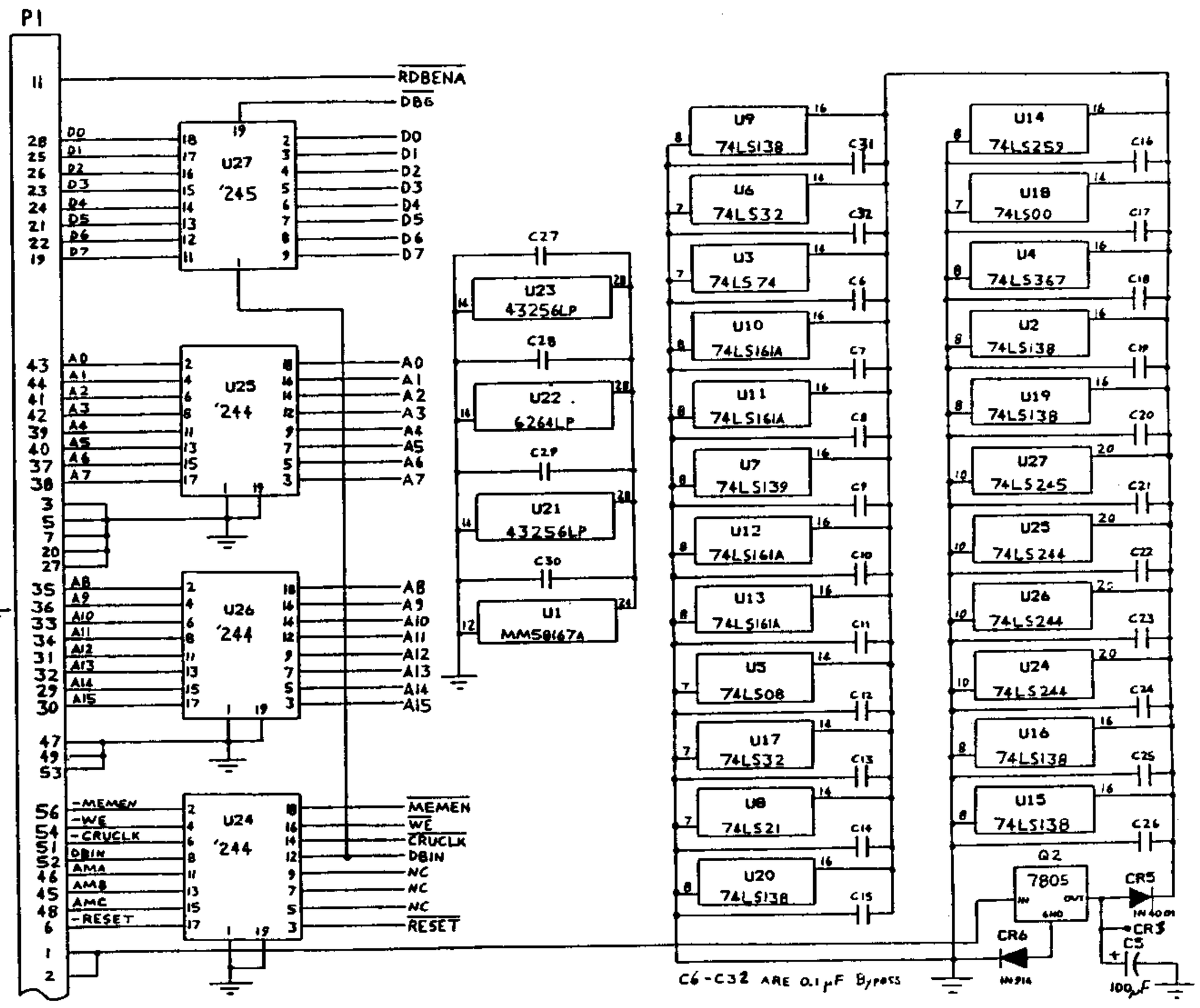
By John Guion



P-GRAM Card

P-Box GROM Emulator/Clock Card

By John Guion



Copyright 1988 John Guion

P-GRAM Card

P-Box GROM Emulator By John Guion