Tektronix®
COMMITTED TO EXCELLENCE

4051R05
BINARY PROGRAM LOADER

OPERATOR'S MANUAL

Tektronix, Inc.
P.O. Box 500
Beaverton, Oregon 97077

MANUAL PART NO.
070-2171-00

First Printing AUG 1976
This Printing AUG 1979

WARRANTY

This product and options, excluding customer supplied equipment, is warranted against defective materials and workmanship under normal use and service for a period of ninety (90) days from date of initial shipment. CRT found to be defective within twelve (12) months from the date of shipment will be exchanged at no charge. (This does not include installation.)

In addition, in those areas where Tektronix has service centers available for this system, on-site warranty repair is provided at no charge during the first ninety (90) days from date of initial shipment.

Tektronix shall be under no obligation to furnish warranty service if:

a. Attempts to install, repair, or service the equipment are made by personnel other than Tektronix service representatives.
b. Modifications are made to the hardware or software by personnel other than Tektronix service representatives.
c. Damage results from connecting the instrument to incompatible equipment.

There is no implied warranty for fitness of purpose.

Copyright © 1976 by Tektronix, Inc., Beaverton, Oregon. Printed in the United States of America. All rights reserved. Contents of this publication may not be reproduced in any form without permission of Tektronix, Inc.

U.S.A. and foreign TEKTRONIX products covered by U.S. and foreign patents and/or patents pending.

TEKTRONIX is a registered trademark of Tektronix, Inc.

TABLE OF CONTENTS

Section 1

GENERAL DESCRIPTION

INTRODUCTION

The Tektronix 4051R05 BINARY PROGRAM LOADER is an accessory to the Tektronix 4051 Graphic System. Four firmware routines are housed in a Read Only Memory device known as a ROM pack. The firmware routines are special sets of instructions for the 4051 microprocessor. The BINARY PROGRAM LOADER adds to the capabilities of the Graphic System without disturbing the normal operation of the system or occupying any RAM (Random Access Memory) storage space.

The BINARY PROGRAM LOADER gives the Graphic System the capabilities to work with program files stored in machine dependent binary code. Without the BINARY PROGRAM LOADER, the Graphic System can only work with data files stored in machine dependent binary code by using the READ and WRITE commands. Now, with the BINARY PROGRAM LOADER, the Graphic System has four binary program routines—BSAV, BOLD, BAPPEN, and LINK. With these routines, you can now save, retrieve, or append a binary program file, and you can also use binary program files as sub-routines by using the LINK routine.

When you use the BINARY PROGRAM LOADER, you may not notice anything different about your program. Your program is listed on the screen just like it always is. You do not see any special code. You may notice that you don't have to wait as long to store a program. The main function of the BINARY PROGRAM LOADER is storage format, which is something you never see.

MACHINE DEPENDENT BINARY CODE

The term "machine dependent binary code" refers to the binary format used by the

Graphic System to store BASIC programs and data in the internal Random Access Memory (RAM). This binary code is unique to the 4051 operating system.

Each BASIC command in an ASCII program file must be translated from ASCII code to machine dependent binary code in order to be ready for execution. Storing a program in machine dependent binary code saves time. For example, the BOLD routine loads a program into RAM from a peripheral storage device in approximately 1/4 the time that the OLD command takes to load the same program stored in ASCII code. This time is saved because the program does not go through a translation . After being loaded, the program is ready to be executed or edited just like a program stored in ASCII code.

From now on, this manual refers to machine dependent binary code as binary code.

SPECIFICATIONS

POWER REQUIREMENTS

The 4051R05 BINARY PROGRAM LOADER draws all necessary power from the 4051 power supplies. Connections to the power supplies are made through the backpack on the rear panel of the 4051 main chassis or from the ROM Expander Unit. The BINARY PROGRAM LOADER ROM pack must be inserted into a slot in the backpack, or into a ROM Expander Unit, before power is applied to the system.

Voltage Supplies	Typical Current	
	Operating	Non-operating*
+5 Vdc	56 mA	8 mA
+12 Vdc	20 mA	0 mA
−12 Vdc	4 mA	4 mA

*The ROM pack is inserted but the routines are not being used at this time.

TEMPERATURE

Non-operating:	$-40^{o}C$ to $+65^{o}C$
Operating:	$+10^{o}C$ to $+40^{o}C$

ALTITUDE

Non-operating:	50,000 feet maximum
Operating:	15,000 feet maximum

HUMIDITY

95% non-condensing (storage)
80% non-condensing (operating)

VIBRATION (NON-OPERATING)

0.015" DA-10-50-10

SHOCK (NON-OPERATING)

1/2 Sine 11 ms duration, 30 G's

PHYSICAL DIMENSIONS (INCLUDING EDGE BOARD CONNECTOR)

Length:	4.662"
Width:	2.620"
Depth:	0.875"

WEIGHT

8 oz.

STANDARD ACCESSORIES

1--Operators' Manual. 070-2171-00

INSTALLATION INSTRUCTIONS (For the 4051 ROM Backpack)

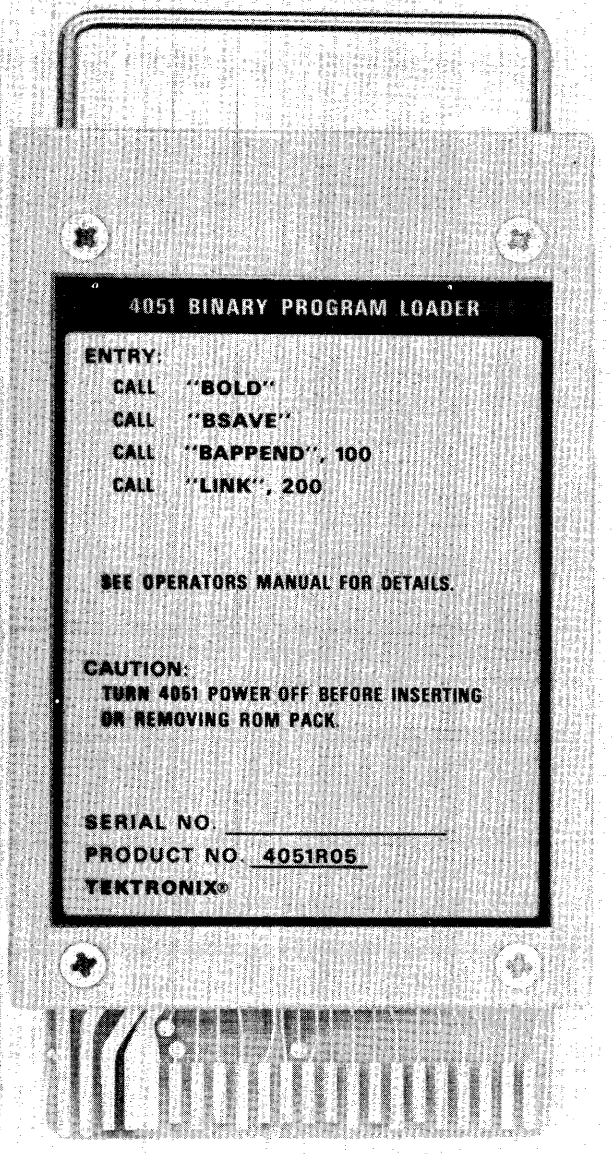
1. Flip the 4051 power switch to the OFF position.

CAUTION

Inserting any device into the backpack when the 4051 power is ON may cause the contents of RAM to be erased. Make sure that important information in RAM is stored on magnetic tape before turning the power OFF and proceeding.

2. With the power removed from the system, insert the BINARY PROGRAM LOADER ROM pack into a slot as shown in Fig. 1-2. Press down gently while rocking the plastic housing from side to side until the ROM pack edgeboard connector is firmly seated in the receptacle connector.

3. Flip the 4051 power switch to the ON position. After a few moments of warm-up, the system is ready to use.



2171-1

Fig. 1-1. 4051R05 BINARY PROGRAM LOADER ROM Pack.

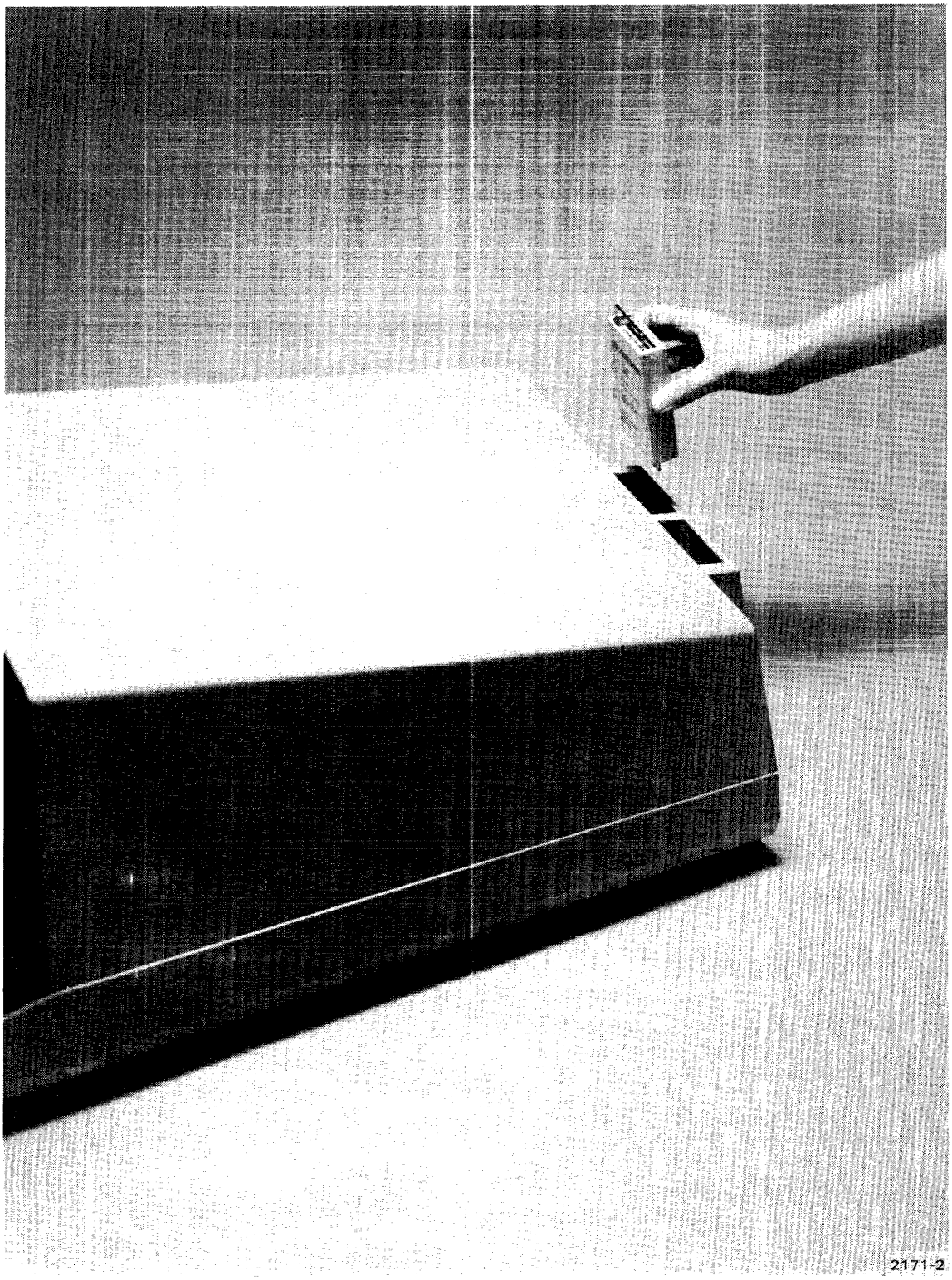


Fig. 1-2. Inserting the BINARY PROGRAM LOADER ROM Pack into a backpack slot.

Section 2

BINARY PROGRAM LOADER ROUTINES

INTRODUCTION

Each of the four firmware routines provided by the BINARY PROGRAM LOADER is explained in this section. If you are already familiar with the operation of the Tektronix 4051, then these routines are a natural extension of the Graphic System you know.

By using the Graphic System BASIC language CALL statement, you can specify the BINARY PROGRAM LOADER routine you want. The routine name can be specified as a string constant in the CALL statement, or the routine name can be assigned to a string variable and represented in the CALL statement by that string variable. Refer to the 4051 Graphic System Reference Manual for more detailed information on using the CALL statement.

Three of the routines (BSAVE, BOLD, BAPPEN) perform the same basic operation on binary program files as their counterpart commands (SAVE, OLD, APPEND) perform on ASCII program files. The syntax of the commands and the storage format of the files operated on may be different, but the same basic operation is performed. For example, the BSAVE routine stores a program on a peripheral device in binary code, while the SAVE command stores a program on a peripheral device in ASCII code. The following chart shows the relationships.

ASCII Files	BINARY Files
SAVE	BSAVE
OLD	BOLD
APPEND	BAPPEN

The explanation of these BINARY PROGRAM LOADER routines includes comparisons to the related ASCII file commands. Use the 4051 Graphic System Reference Manual to find information on the ASCII file commands.

BSAVE ROUTINE

Syntax Form:

[Line number] CALL { "BSAVE" / string variable } (,I/O address)

Descriptive Form:

[Line number] CALL routine name [,I/O address]

PURPOSE

The BSAVE (Binary SAVE) routine stores the current BASIC program on the specified output device in binary code. If an output device is not specified, the internal magnetic tape unit is chosen by default.

EXPLANATION

The Random Access Memory (RAM) holds the current BASIC program in binary code. When executed, the BSAVE routine sends a copy of the current program to the output device in binary code. Like the SAVE statement, BSAVE does not alter assigned values of variables or system environmental conditions. The CALL "BSAVE" statement can be a step in the program being stored, or it can be executed directly from the Graphic System keyboard.

STORING A PROGRAM ON MAGNETIC TAPE IN BINARY CODE

To execute the BSAVE routine, the read/write head of the output device must be positioned at the beginning of a file marked BINARY or NEW. The MARK statement must allocate enough space to store the entire program.

Binary programs use more space on magnetic tape than ASCII programs. The amount depends on the size of the file. Because the SPACE function allocates the approximate space for an ASCII program, the SPACE function may or may not allocate enough space to hold the entire program in binary. If you try to execute the BSAVE routine and the selected file is not large enough to hold the current program, error message number 48 is displayed on the screen. To make sure enough space is allocated to hold the current program, you can use one of the following methods:

METHOD 1 Allocate all of the space the program uses in RAM. For example:

```
MARK 1, 32000 - MEM
```

The MEM function returns the number of bytes still available in RAM. By subtracting this amount from the total storage capacity of RAM, the remainder is the amount of space the current program occupies in RAM. This remainder is also the amount of space needed to store the program on tape.

METHOD 2 Allocate enough space to hold the entire contents of RAM. For example:

```
MARK 1,32000
```

This method may waste a lot of space on the tape if the current program is small.

NOTE

The actual storage capacity of RAM is approximately 32K or 32000 bytes. Of this 32000 bytes, about 2000 are reserved by the microprocessor for a work area. The MEM function

only considers the 30000 bytes that are free. By using 32000 as a constant, you can make sure that you consider all of the possible storage capacity of RAM.

Using the internal magnetic tape unit by default, the following example stores the current BASIC program in binary code.

```
FIND 0
MARK 1,1000
FIND 1
CALL "BSAVE"
```

Since this is the first file on the tape, the read/write head is positioned at the beginning of the tape by the FIND 0 statement. The MARK statement allocates space on the tape to hold 1000 bytes. The read/write head is then positioned to the start of the allocated space by the FIND 1 statement. The CALL "BSAVE" statement then transfers a binary copy of the current program to file 1.

If you now execute a TLIST statement, you can verify the action performed. Note the header of the internal tape for file 1 is marked BINARY PROGRAM.

```
TLIST
 1        BINARY  PROGRAM              1024
 2        LAST                         768
```

USING SECRET WITH BINARY PROGRAMS

The current BASIC program can be marked SECRET and also be stored in binary code. The rules for using SECRET are the same as with any ASCII file. For example:

```
SECRET
FIND 1
CALL "BSAVE"
```

These statements store the current program as a secret program in binary code in file 1. From a TLIST command, you can see that the file header is marked BINARY PROGRAM SECRET as shown below.

```
TLIST
 1        BINARY  PROGRAM   SECRET 1024
 2        LAST                    768
```

To mark the header BINARY PROGRAM SECRET when using an external tape drive, such as the 4924, you must specify the I/O address of the tape drive in a SECRET command. This allows the external device to recognize the program as secret and mark the header accordingly.

For example, if the 4924 tape drive is assigned to device number 2 on the General Purpose Interface Bus, the following program stores the current program as secret in binary code.

```
SECRET @ 2:
SECRET
FIND @ 2:1
CALL "BSAVE",2
```

USING AUTO LOAD WITH BINARY FILES

The AUTO LOAD key finds the first file on the magnetic tape and then executes an OLD command automatically. The OLD command loads and executes the first program on file. Because the OLD command does not work with binary program files, the first program on tape must be an ASCII program, if you use the AUTO LOAD key.

By storing an ASCII program that finds and loads a binary program in file 1, you can use the AUTO LOAD key to automatically find and load a binary program. The following example shows how this can be done.

Using a TLIST command, you can see the programs stored on the magnetic tape.

```
TLIST
 1        ASCII    PROGRAM          768
 2        BINARY   PROGRAM          1792
 3        LAST                      768
```

File 1, an ASCII program file, contains this program:

```
100 FIND 2
110 CALL "BOLD"
```

When the AUTO LOAD key is pressed, the program in file 1 is loaded and executed. This program in turn finds and loads the desired binary program in file 2. Since the CALL "BOLD" statement is executed under program control, the binary program in file 2 is executed after being loaded.

SPECIFYING AN OUTPUT DEVICE

The current program can be stored in binary code on any peripheral device in the system by specifying the desired primary address in the CALL "BSAVE" statement. For example:

```
290 FIND @ 19:5
300 CALL "BSAVE",19
```

This statement sends the current program in binary code to file number 5 on device 19 on the General Purpose Interface Bus.

CAUTION

Unless the internal tape file is closed when executing a BSAVE to an external device, data may be inadvertently written on the internal magnetic tape. Ensure the internal tape file is closed before executing a BSAVE to an external device.

BOLD ROUTINE

Syntax Form:

[Line number] CALL { "BOLD" / string variable } [,I/O address]

Descriptive Form:

[Line number] CALL routine name [, I/O address]

PURPOSE

The BOLD (Binary OLD) routine loads a copy of the program stored in binary code into the Random Access Memory (RAM) from the specified input source. If an input source is not specified, the internal magnetic tape unit is chosen by default.

EXPLANATION

Any program that is in the machine dependent binary code of the 4051 Graphic System can be loaded into RAM by using the BOLD routine. A program stored by the BSAVE routine is usually the program retrieved by the BOLD routine. BOLD also loads BINARY SECRET programs, and the retrieved programs remain secret.

LOADING A BINARY PROGRAM FROM MAGNETIC TAPE

To retrieve a binary program from magnetic tape, the read/write head of the output device must first be positioned at the beginning of a binary program file. For example, using the internal magnetic tape unit by default, the statements:

```
FIND 1
CALL "BOLD"
```

load the binary program from file 1 into RAM. The FIND statement locates the beginning of file 1. The BOLD routine erases everything currently in RAM, then transfers the binary file into RAM. The loaded program is ready to be executed by a RUN statement or edited. Like the OLD command, if the BOLD routine is executed under program control, a RUN statement is automatically executed after the program is loaded into RAM.

SPECIFYING AN INPUT DEVICE

Any peripheral device holding a binary program can be specified as the input source for the BOLD routine by designating the appropriate I/O address. The following statement specifies file number 6 on device number 2 on the General Purpose Interface Bus.

```
290 FIND @ 2:6
300 CALL "BOLD",2
```

BAPPEN ROUTINE

Syntax Form:

[Line number] CALL { "BAPPEN", / string variable, } [I/O address;] line number [,increment]

Descriptive Form:

[Line number] CALL routine name, [I/O address;] target line number in current program [line number [,increment]]

PURPOSE

The BAPPEN (Binary APPEND) routine inputs BASIC statements stored in binary code on the specified peripheral device and attaches those statements to the program currently in RAM. The internal magnetic tape unit is selected by default if an input device is not specified.

EXPLANATION

The BAPPEN routine follows the same procedure as the APPEND statement. First, the read/write head of the input device must be positioned to the beginning of a binary program file by using the FIND command. Then, when the BAPPEN routine is executed, the given target statement is overwritten by the first statement coming from the peripheral device. The newly appended statements and any statements that originally followed the target statement are renumbered, stating with the target line number. The line numbers are incremented by either the given increment, or by the default of 10 if an increment is not specified.

For example:

```
.
.
.
200 REM APPEND STATEMENTS FROM FILE 2 HERE
    FIND 2
    CALL "BAPPEN",200
```

When these statements are executed, the FIND statement positions the read/write head at the beginning of File 2. File 2 must be a binary program file. (Error message number 55 is printed if file 2 is not binary). The BAPPEN routine replaces line 200 with the first statement in file 2. The remaining statements in file 2 are then appended with line numbers incremented by 10.

BAPPEN WITH SECRET PROGRAMS

If the program you want to append is stored as a SECRET BINARY program on the peripheral device, after executing the BAPPEN routine, the entire contents of RAM becomes secret. This includes all original statements and the statements appended.

SPECIFYING AN INPUT DEVICE

Any peripheral device storing a binary program can be specified as the input device. The device is specified like this:

```
FIND @ 15:3
CALL "BAPPEN",15;650,50
```

This statement selects file number 3 on device number 15 on the General Purpose Interface Bus as the input device. Line numer 650 is the target statement in the current program stored in RAM. The renumber increment is 50.

LINK ROUTINE

Syntax Form:

[Line number] CALL { "LINK", / string variable, } [I/O address;] line number

Descriptive Form:

[Line number] CALL routine name, [I/O address;] line number of entry point

PURPOSE

The LINK routine transfers a stored binary program to RAM from the specified peripheral device without disturbing variables and associated values stored in RAM. If an input device is not selected, the internal magnetic tape unit is selected by default.

EXPLANATION

The LINK routine erases the program currently in RAM, but retains all variables and their currently assigned values. Then, a binary program is loaded into RAM from the specified input device. The BASIC line counter is set to the starting line number and execution starts at that point.

These features allow you to break a large BASIC program into subroutines, store them on tape, and then structure the flow of execution by using the LINK routine. By using the LINK routine, the size of a program is no longer limited by the storage capacity of RAM. If RAM cannot hold an entire program, you can break the program into sections and use the LINK routine to load and execute the program sections.

LOCATING THE LINK POINT

To use the LINK routine, the read/write head of the peripheral device must first be positioned at the beginning of a binary program file.

To locate the beginning of a binary program file, use the FIND statement. After the LINK routine is executed, variables and values are kept and the current program is erased. The entire binary program (previously located by the FIND statement) is then loaded into RAM.

If the LINK routine is executed in immediate mode, execution stops after the program is transferred. The loaded program can then be executed by entering a RUN statement and pressing the RETURN key. Execution begins with the starting line number.

If the LINK routine is executed under program control, then after the binary program is transferred, execution automatically begins with the starting line number.

USING LINK WITH SECRET BINARY PROGRAMS

If a LINK routine brings a SECRET BINARY program into RAM, then the entire contents of RAM are made secret. As always, secret programs can only be executed.

SAMPLE PROGRAMS

To use the LINK routine a BASIC program must be stored in binary code. As an example, the following program is stored in file 1 on the internal magnetic tape unit. Since this file is stored in binary code, the file is retrieved by using the BOLD routine. A LIST statement is executed to show the program.

```
FIND 1
CALL "BOLD"
LIST
100 INIT
110 A$="CHANGE"
120 PRINT "SOME THINGS NEVER ";A$
```

Now, the program is executed.

```
RUN
SOME THINGS NEVER CHANGE
```

This binary program is used in the following examples by the LINK routine.

```
100 A$="STAY THE SAME"
110 FIND 1
120 CALL "LINK", 120
```

In line 100, A$ is given the value "STAY THE SAME". The binary file is then located by the FIND statement. The LINK routine directs execution to begin in line 120 of the binary program. So. . .

```
RUN
SOME THINGS NEVER STAY THE SAME
```

The stored binary program is now in memory. No statements have been changed. A$ retains the value "STAY THE SAME" because execution started in line 120.

Notice the INIT statement in line 100 of the binary program. If the LINK routine is specified by

```
CALL "LINK",100
```

then execution would begin with line 100. The INIT statement puts all variables in an undefined state, including any variables retained by the LINK routine. A$ would be undefined. Then in line 110 of the binary program, A$ would be set to "CHANGE".

If the LINK routine is specified by

```
CALL "LINK",110
```

then A$ is retained with the value "STAY THE SAME", but is set to "CHANGE" in line 110 of the binary program.

From these examples, you can see how execution and program flow can be controlled by using the LINK routine.

SPECIFYING THE INPUT DEVICE

The LINK routine uses any peripheral device in the Graphic System. You can specify the desired peripheral device like this:

```
FIND @ 2:8
CALL "LINK",2;150
```

In this case, file number 8 on device number 2 is selected as the input device on the General Purpose Interface Bus. Execution begins with line 150.

Appendix A

SUMMARY

1. The BINARY PROGRAM LOADER adds four routines to the Tektronix 4051 Graphic System. Three of the routines (BSAVE, BOLD, BAPPEN) perform the same basic operation on binary files as the commands (SAVE, OLD, APPEND) perform on ASCII program files. The following chart sumarizes the four routines and their relationships to other statements.

BINARY	ASCII
BSAVE	SAVE
BOLD	OLD
BAPPEN	APPEND
LINK	

2. All routines are specified by using a CALL statement.

3. All routines access binary program files.

4. The read/write head of the peripheral device must be positioned to the beginning of the binary program file before executing a BINARY PROGRAM LOADER routine. Use the FIND statement to locate the beginning of a file.

5. Binary programs may be also declared SECRET. The SECRET BINARY program remains secret when used by a BINARY PROGRAM LOADER routine.

Appendix B

ERROR MESSAGES

MESSAGE NUMBER	ERROR MESSAGE ——— PROBABLE CAUSE AND SOLUTION
14	INVALID COMMAND ARGUMENT ——— A parameter in the CALL statement for the BAPPEN routine is invalid. Check the syntax of the CALL statement for the BAPPEN routine.
15	INVALID COMMAND ARGUMENT ——— The line number specified in the CALL statement for the BAPPEN routine does not exist. List the program to find the correct line number.
32	CALL NAME INVALID ——— The routine name is misspelled. Check the spelling of the selected routine name. ——— A string variable appears in a CALL statement with a value of no recognizable routine name. Check the spelling of the value assigned to the string variable. ——— The 4051R05 ROM pack is not installed in the machine. See the installation instructions included in this manual.
38	PROGRAM IS SECRET ——— You attempted to modify or output a secret program. Secret programs can only be executed after being retrieved.

48	EOF ON UNIT O IN IMMEDIATE LINE ——— You tried to store a program in a file that is not large enough to hold the entire program. Either allocate more space for that file or locate or make a file with more space.
51	INVALID LINE NUMBER ——— The line number specified cannot be found or is invalid. List the program to find the correct line number.
52	MAG TAPE FILE NOT FOUND ——— Either the specified magnetic tape file does not exist or an attempt has just been make to KILL the LAST (dummy) file. Issue a TLIST to find the desired file number.
55	MAG TAPE ERROR ——— The read/write head is not at the beginning of a file before accessing that file. Execute a FIND statement to position the read/write head before using the file. ——— You tried to use an ASCII program file with one of the BINARY PROGRAM LOADER routines. The BINARY PROGRAM LOADER routines only operate on a BINARY PROGRAM file. ——— You tried to use an ASCII program file command with a BINARY PROGRAM file. BINARY PROGRAM files can only be used by the BINARY PROGRAM LOADER routines.
57	MAG TAPE CARTRIDGE REQUIRED ——— A magnetic tape cartridge is not inserted into the magnetic tape unit. Insert a magnetic tape into the tape unit.

61	INVALID OPERATION ON AN OPEN FILE ——— A MARK statement was executed and the read/write head of the peripheral device was not positioned at the beginning of a file. Use the FIND statement to position the read/write head.