OS/40

8540 Integration Unit

System Reference Booklet

Version 1

Tektronix

First printing Nov. 1981, in U.S.A. Information in this publication supersedes all previously published material. U.S.A. and Foreign Products of Tektronix, Inc. are covered by U.S.A. and Foreign Patents and/or Patents Pending.

070-3992-00
Product Group 61

@

CONTENTS

SECTION 1 OVERVIEW

Page

System Startup 1-2

SECTION 2 COMMAND DICTIONARY

Special Keys 2-1
Command Index 2-1
Command Line Syntax 2-3
  Strings 2-4
  Expressions 2-5
Notation Conventions 2-5
Commands 2-7

SECTION 3 SERVICE CALLS

Overview 3-1
SVC Function Code Descriptions 3-3

SECTION 4 INTERSYSTEM COMMUNICATION

Overview 4-1
COM Interface 4-1

SECTION 5 ERROR MESSAGES

SECTION 6 GLOSSARY

SECTION 7 INDEX

TABLES

Table No.		Page
1-1	Device Names, Jacks, and Functions	1-2
1-2	Emulation Modes	1-2
2-1	COM Command Options	2-11
2-2	Counter Source Parameters	2-13
2-3	Counter Gate Parameters	2-14
2-4	Counter Output Parameters	2-14
2-5	EVE Command Options	2-17
3-1	Summary of LAS and SAS Formats for SRBs	3-2
3-2	Device Identification and Type	3-5

ILLUSTRATIONS

Fig. No.		
1-1	8540 logical subsections	1-1
2-1	Sample syntax block	2-6
3-1	Service call memory layout, SAS format	3-2
3-2	Service call memory layout, LAS format	3-3

Section 1

OVERVIEW

The TEKTRONIX 8540 Integration Unit is a tool for performing hardware/software integration tasks. Figure 1-1 shows the components of a complete 8540 system.

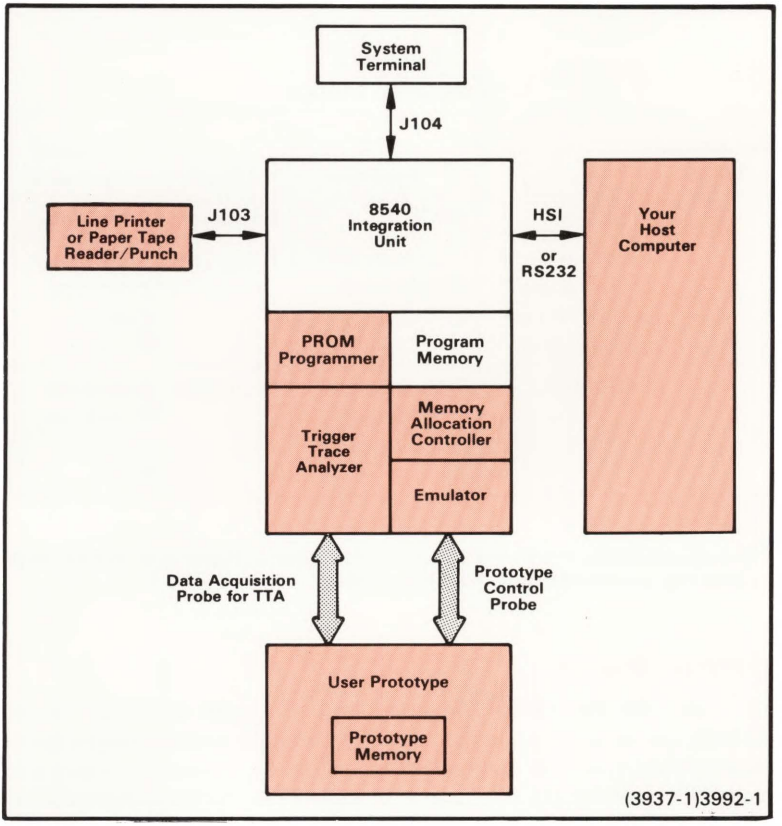


Fig. 1-1. 8540 logical subsections.

This functional diagram shows the parts of a complete 8540 system. Shaded areas indicate equipment that is not part of the minimum 8540 configuration.

Table 1-1
Device Names, Jacks, and Functions

Device Name	Jack Number	Function
CONI	J104	Console input
CONO	J104	Console output
LPT	J103	Line printer
PPTR	J103	Paper tape reader
PPTP	J103	Paper tape punch
REMI	J101 (DTE), J102 (DCE)	Remote input
REMO	J101 (DTE), J102 (DCE)	Remote output

Table 1-2
Emulation Modes

Emulation Mode	Memory	I/O Facilities	Clock	Required Hardware
0	Program	SVCs	Emulator	Emulator
1	Program and/or Prototype, depending on memory map	Prototype and (if selected) SVCs[a]	Prototype	Emulator, Prototype Control Probe
2	Prototype	Prototype and (if selected) SVCs[a]	Prototype	Emulator, Prototype Control Probe

[a] **Not all emulators support SVCs in Modes 1 and 2. Refer to your Emulator Specifics supplement for this information.**

System Startup

You can use the PERMSTR command to create a string (called $STARTUP) of commands that perform common system initialization tasks. If Switch S1100 on the 8540's System Controller board is in the closed (0) position, the commands in $STARTUP are executed as soon as you boot your 8540.

The 8540 boots in LOCAL mode. If you want to boot in TERM mode (as with an 8560), a CONFIG TERM command must be the **first** command in the $STARTUP string.

Section 2

COMMAND DICTIONARY

SPECIAL KEYS

CTRL-C—Interrupts command or program execution
CTRL-Q—Resumes console display without being echoed
CTRL-R—Displays the text in the type-ahead buffer
CTRL-S—Halts console display
CTRL-U—Deletes the text in the type-ahead buffer
CTRL-Z—Marks end of ASCII input
BACKSPACE—Deletes character
RUBOUT—Deletes character

COMMAND INDEX

System Commands

A—Aborts user program or command
CALC—Evaluates arithmetic expression
CO—Resumes execution of suspended command
LOG—Sends terminal input/output to a device
QUERY—Turns query mode on or off
PERMSTR—Places string in permanent storage area
ROMPATCH—Stores system patch in EEPROM
SEL—Selects target processor
STAT—Displays system status
STR—Displays or deletes current user strings
SUSP—Suspends command

Intersystem Communication Commands

COM—Sets up communication with host computer
CONFIG—Defines system configuration and host interface

Memory Management Commands

AL—Allocates memory to virtual memory map
D—Displays memory contents
DEAL—Deallocates memory from virtual memory map
EX—Displays or alters memory contents
F—Fills memory with data
LO—Loads program into memory
MAP—Sets or displays memory map assignments
MEM—Specifies prototype memory to be available to program
MEMSP—Defines memory space to be used by memory commands
MOV—Moves data between program and prototype memory
NOMEM—Specifies prototype memory to be unavailable to program
P—Alters memory contents
RH—Reads hexadecimal code into memory
SAV—Saves memory contents into host file
SEA—Searches memory for value or string
WH—Saves memory contents in hexadecimal format
X—Loads and executes program

Debugging and Emulation Commands

ADDS—Adds symbol to symbol table
AS—Assigns channel to device or host file
BK—Defines or displays breakpoint condition
CL—Disconnects channel from device
CLOCK—Controls program clock
DI—Translates object code to mnemonics
DS—Displays contents of emulator registers
EM—Selects emulation mode
G—Begins program execution
RD—Reads byte or word from emulator I/O port
REMS—Deletes symbol from symbol table
RESET—Reinitializes emulator
S—Assigns value to register or symbol
SVC—Controls execution of service calls from user program
SYMB—Finds symbolic equivalent of value
SYMD—Turns symbolic display on or off
SYMLO—Loads symbols and their values into memory
TRA—Controls display of executed instructions
WRT—Writes byte or word value to emulator I/O port

Trigger Trace Analyzer (TTA) Commands

ACQ—Selects bus transactions to be stored in Acquisition Memory
AD—Selects address portion of an event
BRE—Defines a breakpoint in terms of TTA events
BUS—Selects control signal portion of an event
CONS—Defines trigger in terms of consecutive events
COU—Selects counting units and output modes for TTA counters
CTR—Defines counter output portion of an event
DATA—Selects data portion of an event
DISP—Displays contents of Acquisition Memory
EVE—Defines an event in terms of input data
PRO—Defines probe signal portion of an event
QUA—Defines external input portion of an event
TCLR—Clears TTA trigger channels
TS—Displays status of TTA or selected triggers

PROM Programmer Commands

CPR—Compares PROM with memory
PSTAT—Displays PROM Programmer status
PTYPE—Displays PROM types
RPR—Reads data from PROM into memory
WPR—Writes data from memory to PROM

COMMAND LINE SYNTAX

- A command line consists of one or more commands or string definitions, separated by semicolons. A command consists of a command name plus any command modifiers or parameters.
- The maximum length of a command line is 80 characters, including the carriage return.
- A command modifier consists of a dash (-) followed by a single letter. Command modifiers can be grouped together: -A -B -C can be entered as -ABC.
- Parameters are separated from each other and from the command name by delimiters. A delimiter is a comma or one or more spaces.
- To omit an optional parameter, enter two commas in its place. To omit two consecutive parameters, enter three commas. Do not enter commas to omit a command modifier or the final parameter(s) in a command line.

- Capitalization is ignored for all elements of a command line except filenames and string names.
- A backslash (\) removes any special significance of the character following it. Each of the following characters has a special meaning, and must be immediately preceded by a backslash if you want to disable its special significance: \ < > ; * ? $ " ' Two consecutive backslashes are interpreted as a single backslash.
- You can redirect standard input and output from the system terminal to a specified host file or device. A left arrow (<) redirects input; a right arrow (>) redirects output. The redirection parameter can be inserted anywhere in a command line after the command name.

The following additional syntax conventions apply when you are in TERM mode with an 8560:

- Command names must be entered in lowercase.
- The use of commas as delimiters is restricted.
- Certain characters that have special meaning to the TNIX operating system must be preceded by a backslash (\) to disable their special significance.

For more information about TERM mode, refer to to your 8560 System Users Manual.

Strings

You may assign a name to a string of characters, and then refer to the string by name in a command line. To define a string, enter the string name, an equals sign, and the string.

A string name can be up to eight characters in length. It must begin with a letter, and must contain only letters and digits. Uppercase and lowercase letters are distinct.

If a string contains a delimiter such as a space, comma, or semicolon, the string must be enclosed in single or double quotes. If it contains a dollar sign, backslash, or double quote, the string must be enclosed in single quotes. Special characters in a quoted string need not be preceded by backslashes.

You can refer to a string anywhere in a command line by entering a dollar sign followed by the string name. When you use a string name in place of a full string, a simple character substitution is performed; thus, be careful of the 80-character limitation on command lines.

Strings are volatile and are lost when the 8540 is restarted or turned off. Use the PERMSTR command to place string names and values in the permanent string storage area.

Expressions

Many numeric parameters may be entered as expressions. An expression is a sequence of elements related by the operators + and −. An expression must not contain a space.

The following types of elements are permitted in an expression:

- Numbers. Each number is suffixed with a letter that indicates its radix: H—hexadecimal (default); T—decimal; O or Q—octal; Y—binary. A hexadecimal number cannot begin with a letter.
- Symbols. Any valid symbol, as defined in your Assembler Users Manual. To distinguish between identical symbols in different sections, you may prefix a symbol with a section name—"section:symbol".
- Don't-Cares. An "x" within a non-decimal expression signifies a don't-care value. Don't-cares are used primarily with TTA commands.
- Register names (microprocessor-specific).
- Memory space designators (microprocessor-specific).

NOTATION CONVENTIONS

Figure 2-1 illustrates the syntax of a hypothetical command, **SAMPLE**, which contains the following elements:

a. a command name, **sample**

b. a command modifier, **−m**

c. a required first parameter, **addr1**

d. an optional second parameter, **addr2**

e. an optional third parameter, which is either PA or PB

f. a required fourth parameter, which is either an address or a string

g. optional subsequent parameters, each of which is an address or a string

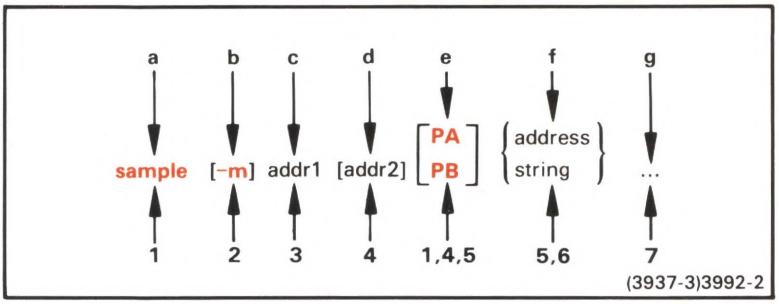


Fig. 2-1. Sample syntax block.

The following conventions are illustrated in Fig. 2-1:

1. Items shown in **colored** type must be entered exactly as shown.
2. Command modifiers are optional.
3. Items in black type identify the parameter type.
4. Optional parameters are enclosed in brackets [ ].
5. Stacked terms indicate a choice of parameters.
6. Groups of required parameters are enclosed in braces { }.
7. Trailing dots indicate that the previous parameter may be repeated.

The following command lines conform to the syntax in Fig. 2-1:

```
> SAMPLE myaddr,,,"A string"
> SAMPLE yraddr,,,100 101 102
> SAMPLE -M hisadd heradd,,0FFFF "Another string"
> SAMPLE here there PA "Yet another" 5 88 3 "and another"
```

For some parameters, an abbreviated form is allowed. The parameter appears in colored type with the abbreviation **underlined**.

COMMANDS

a { command / **-a** / **-u** }

Terminates execution of the specified command or program. Closes all channels assigned to the aborted process. **-A** aborts all active commands or programs. **-U** closes the channels of your program.
The following commands cannot be aborted: A, CO, COM, G, LO, SEL, SUSP, X.

acq { **ev4** / **all** } **for** n units

Selects bus transactions to be stored in TTA's Acquisition Memory. Defaults to the most recent 255 transactions.
The **for** clause specifies that acquisition is to stop after **n** units have been counted. **units** may be one of the options available to the S parameter of the COU (Count) command: time intervals, event or trigger occurrences, Acquisition Memory transactions, bus cycles, qualifier signals, and emulator clock signals.

acq { **ev4** / **all** } **for** n units **aftertrig4**

Stops acquisition **n** units after TTA trigger 4 occurs.

acq

Displays the current setting of the TTA's Acquisition Memory.

ad [**-n**] [**-c** / **-s**] evt address [address]

Defines an address or range of addresses as a TTA event. **-N** defines the event as any address **other than** the values specified in the command line.
address may include don't-cares. **evt** can be 1, 2, 3, or 4.
-S sets a breakpoint at the associated trigger. **-C** continues program execution after the break display.

ad {**all** / evt} **clr**

Clears previous address values for the specified TTA event.

adds [**-s**] symbol=value...

Places a user-defined symbol and its associated value in the user symbol table. **-S** indicates that the value is a scalar. **value** is an expression whose value is assigned to the symbol. An address may be of the form "section:label"; for a scalar, only the label part is allowed. If the symbol is associated with an absolute section, you must specify the section.

al [actual] [blocks] [virtual]

Reallocates an area of program memory to a virtual address range. The area to be reallocated is defined by its starting address (**actual**) and its size in 4K-byte blocks. **actual** defaults to 0. **blocks** defaults to all blocks between **actual** and the end of program memory. **virtual** defaults to **actual**. The target processor must be selected before you use AL. Entering AL with no parameters displays the memory allocation status.

al [loaddr [hiaddr]]

Used with the MAC option and (generally) with an emulator such as the Z8001/Z8002 or 68000. Declares a range of addresses to be used by your program. 4K-byte blocks of program memory are allocated. **loaddr** is truncated to a multiple of 1000H. **hiaddr** defaults to **loaddr**+0FFF. Memory spaces may be specified.

as {channel filespec} ...

Assigns the specified channel number(s) to the specified device(s) or host file(s). CONI, CONO, and host files may each be assigned to more than one channel. All other devices are limited to one channel at a time. Channels are numbered 0-9. Channels 8 and 9 are initially assigned to standard input and standard output, respectively. To assign channel 8 or 9 to some other device, you must first close that channel. The target processor must be selected before you use AS.

bk [**-c**] { **1** / **2** } expression [**rd** / **wt**]

Sets a breakpoint at the specified address. For most emulators, up to two breakpoints can be set at a time. If neither RD (read) nor WR (write) is specified, the break can occur on either type of access. Entering BK with no parameters displays the currently set breakpoints. The target processor must be selected before you use BK.
-C continues program execution after the break display. If -C is omitted, execution stops at the breakpoint.

bk [{ **1** / **2** / **all** } **clr**]

Clears the specified breakpoint(s).

bre [trig [**cont** / **stop** / **clr**]]

Sets or clears a breakpoint for the specified TTA trigger(s). **trig** can be 1, 2, 3, 4, or **all**.
stop halts program execution when the breakpoint is reached; **cont** continues program execution. **cont** and **stop** take precedence over any previous -C or -S parameter for the trigger channel.
BRE with only a trigger parameter sets the specified breakpoint, with a default of **stop**. Entering BRE with no parameters displays the currently selected BRE parameters.

bus [**-s** / **-c**] evt signal ...

Specifies bus/control signals that are recognized as a TTA event. **signal** is an emulator-specific symbol that represents a bus/control signal. Signals are ANDed together. **evt** can be 1, 2, 3, or 4.
-S sets a breakpoint at the associated trigger; **-C** continues program execution after the break display.

bus { **all** / evt } **clr**

Clears previous bus symbol parameters for the specified TTA event.

calc [-radix] expression

Displays the value of the specified expression.
-**radix** selects the number base you want the value displayed in: -H: hexadecimal (default); -T or -D: decimal; -Q or -O: octal; -Y or -B: binary.

cl channel ...

Closes the designated channel(s). The target processor must be selected before you use the CL command.

clock [**on** | **off** | value]

on causes the program clock to increment every 100 milliseconds during program execution. **off** disables the program clock. **value** sets the program clock to the specified value (expressed as a decimal integer between 0 and 65535). Entering CLOCK with no parameters displays the value in the program clock and the number of counts since the last G (Go) command. At system startup, the program clock is OFF and contains a value of 0. The target processor must be selected before you use CLOCK.

co {command | **-a**}

Continues execution of the specified command. -**A** continues all suspended commands.

com [option ...]

Initiates communication with a host computer, according to the options listed in Table 2-1. Options may be entered in the command line in any order.

Table 2-1
COM Command Options

Option Syntax	Meaning
e={r|l}	Selects remote (R) or local (L) echoing. Defaults to R (host echoes).
l={o|i}	Linefeed parameter. L=I: 8540 should output a linefeed with each carriage return. L=O: host supplies linefeed. Defaults to L=O.
p=prompt	Specifies prompt sequence used by host: an even number of up to 32 hex digits representing up to 16 ASCII characters. Defaults to no prompt.
t=delay	Turnaround delay parameter. **delay** is a 2-digit hex number representing 100-millisecond units. Defaults to 00.
m=parity	Selects parity option required by host. See the Intersystem Communication section of the 8540 System Users Manual for parity values.
c={t|i}	Error check parameter. C=T: COM execution stops if a communication error occurs on the remote port. C=I: execution continues. Defaults to C=T.
f=t	Specifies Standard Tekhex load module format. Defaults to Extended Tekhex format.
eol=hexstring	End-of-line parameter: an even number of up to 32 hex digits representing up to 16 ASCII characters.
hs=off	Handshaking parameter: eliminates the ACK/NAK response. Defaults to on.
subch1=ch2[/ch1=ch2]...	Specifies character substitution(s) performed on data transmitted to or from the host. **ch1** and **ch2** are hex representations of ASCII characters. **ch1** is the host character; **ch2** is the 8540 counterpart.

config term [e={r|l}] [l={o|i}] [i={h|r}] [m=parity] [t=factor]

Configures the 8540 to act as if the terminal were connected only to the host computer.
E=R specifies that the host provides echoing of characters entered from the system terminal. **E=L** specifies that the 8540 provides echoing. Echoing defaults to E=R (remote).
L=I specifies that the 8540 should output a linefeed with a carriage return. With **L=O**, the host supplies linefeeds. Defaults to L=I.
I=H specifies that the host interface is through the HSI port. **I=R** specifies the remote port (J101 or J102). Defaults to I=H.
M selects the parity option required by the host for data transmission. Defaults to 4. Values for M are given in the Intersystem Communication section of the 8540 System Users Manual.
factor is a two-digit hexadecimal number that represents a timeout multiplier. Defaults to 1.

config local [m=parity]

Places the 8540 in stand-alone mode. Characters entered from the terminal are processed directly by OS/40.

cons {emu|fet|cyc} {sequence}...

Causes a TTA trigger after a sequence has occurred on consecutive cycles: **cyc** specifies bus cycles, **fet** specifies fetch cycles, and **emu** specifies emulator-dependent bus cycles.
sequence is a 2-to-4 digit number that specifies the order of the events that form the sequence. Entering CONS with no parameters displays the current CONS settings for all events.

cons clr

Specifies that events are **not** linked.

cou [**-s** / **-c**] cntr { **r**=restart / **g**=gate / **o**=output / **s**=source / **v**=value }

Specifies counting units and output modes for the TTA's general purpose counters. **cntr** specifies the counter: 1, 2, 3, or 4. The counter is initialized to **value**.
source specifies the units to be counted. (See Table 2-2.)
gate controls **when** the source is counted. (See Table 2-3). **gate** is available only on counters 2–4.
restart can be set to **on** or **off**. If it is **on**, **value** is reloaded into the counter each time the specified **gate** condition becomes true.
output controls the output of the counter and whether the counter counts up or down. (See Table 2-4.)
-S sets a breakpoint at the associated trigger; **-C** continues program execution after the break display.

Table 2-2
Counter Source Parameters

Parameters	Signal Source for Counter N
s=200nsec	200 nanosecond intervals
s=2usec	2 microsecond intervals
s=20usec	20 microsecond intervals
s=200usec	200 microsecond intervals
s=2msec	2 millisecond intervals
s=evN	Occurrences of event 1, 2, 3, or 4
s=trigN	Occurrences of trigger 1, 2, 3, or 4
s=acq	Transactions stored in Acquisition Memory
s=cyc	All bus cycles
s=emuclk	Clock on emulator
s=qua	Low-to-high transitions on the Event Qualifier BNC input

Table 2-3
Counter Gate Parameters

Parameters	Explanation
g=off	Removes any other counter gate restrictions
g=ctr	Counts only when counter N-1 remains high
g=trigh	Counts only when trigger N-1 remains high
g=trigl	Counts only when trigger N-1 remains low
g=seqh	Starts counting after trigger N-1 becomes high
g=seql	Starts counting after trigger N-1 becomes low
g=self	Counts only when trigger N remains high

Table 2-4
Counter Output Parameters

Parameters	Output	Counter Action
o=arm	Remains high	Counts up
o=disarm	Remains low	Counts up
o=pulse	Is low while counting: pulses high when counter reaches 0, then returns to low	Counts down
o=delay	Is low while counting; becomes high after counter reaches 0	Counts down
o=timeout	Is high while counting; becomes low after counter reaches 0	Counts down

cou {**all** / cntr} **clr**

Clears the specified TTA counter(s).

cpr [**-a**] [**-d**] [**-m** / **-l**] [**-r**] memlo promtype promlo promhi

Compares the contents of PROM addresses **promlo** through **promhi** with the contents of program memory beginning at address **memlo**.
-A specifies that the PROM requires complemented addresses.
-D specifies that the PROM provides complemented data.
-M indicates that data being compared represents the most significant word halves; **-L** indicates least significant word halves.
-R specifies that bytes are to be arranged in "reverse" order, with the least significant byte first.

ctr [**-s** / **-c**] evt pattern

Defines a TTA event as a pattern of the four counter outputs. **evt** may be 1, 2, 3, or 4.
pattern is a 4-character "word" composed of 1's, 0's and x's; it represents the outputs of counters 1–4, respectively.
-S sets a breakpoint at the associated trigger; **-C** continues program execution after the break display.

ctr {**all** / evt} **clr**

Sets the TTA counter output pattern to all don't-cares.

d [**-b** / **-w**] loaddr [hiaddr]

Displays the contents of the specified program/prototype memory addresses in hexadecimal and ASCII formats on the standard output device. **hiaddr** defaults to **loaddr**+0F or to the high end of your microprocessor's memory space, whichever is less. **-B** specifies byte-oriented output. **-W** specifies word-oriented output. The default for -B/-W is emulator-specific.

data [**-n**] [**-s** / **-c**] evt {value [value]}

Defines a TTA event as the occurrence on the data bus of a value or one of a range of values. **value** is any valid numeric expression and may include don't-cares. **evt** is 1, 2, 3, or 4.
-N defines the event as the occurrence of any value **other than** the specified value or range of values.
-S sets a breakpoint at the associated trigger; **-C** continues program execution after the break display.

data {**all** / evt} **clr**

Clears previous data values for the specified TTA event(s).

deal {**-a** | loaddr [hiaddr]}

Deallocates memory previously allocated with the AL (allocate) command. Used only with the MAC option. The address parameters may include memory space designators. Memory is deallocated in 4K-byte blocks.

di [loaddr] [hiaddr] [lines]

Translates object code in program/prototype memory into assembly language mnemonics. The area of memory to be disassembled is defined by **loaddr** (lower boundary, defaults to 0) and either **hiaddr** (upper boundary) or **lines** (number of lines of assembly language). If neither **hiaddr** nor **lines** is specified, disassembly continues to the end of your microprocessor's memory space or until you press CTRL-C. The target processor must be selected before you use DI.

disp [**all** | n]

Displays contents of TTA's Acquisition Memory. **all** displays the latest 255 transactions. **n** displays the latest **n** transactions. Entering DISP with no parameters displays all transactions stored since the last G (Go) command.

ds [-l]

Displays the contents of the emulator registers. **-L** displays all registers. For some emulators, the default display lists only the principal registers. The target processor must be selected before you use DS.

em [mode]

Selects emulation mode 0, 1, or 2. If no mode is specified, the current mode is displayed. The target processor must be selected before you use EM.

eve [**-c** / **-s**] evt option...

Defines a TTA event. **option** is one or more of those shown in Table 2-5. **evt** is 1, 2, 3, or 4.
-S sets a breakpoint at the associated trigger; **-C** continues program execution after the break display.

Table 2-5
EVE Command Options

Option	Equivalent To
a=address [address]	AD command
an=address [address]	AD command with -N modifier
d=data [data]	DATA command
dn=data [data]	DATA command with -N modifier
b=symbol	BUS command
p=value	PRO command
c=pattern	CTR command
q=X	QUA command

eve {**all** / evt} **clr**

Clears previous parameters for the specified TTA event(s).

ex [**-b** / **-w**] [**-n**] address

Interactively modifies successive memory locations beginning at **address**.
-B specifies byte-by-byte modification. **-W** specifies word-by-word modification. The default for -B/-W is emulator-specific. **-N** suppresses the read-back check for the command.
The following keys have special meaning for the EX command:

Space bar: Displays the next address and its contents.
BACKSPACE: Displays the previous address and its contents.
LINEFEED or RUBOUT: Displays the current address and its contents.
RETURN or CTRL-C: Ends execution of EX command.

f [**-b** / **-w**] [**-n**] loaddr hiaddr { hexstring / **-a** string } ...

Fills memory locations **loaddr** through **hiaddr** with the byte sequence obtained by concatenating all specified hexadecimal and ASCII strings.
-A specifies an ASCII string.
-B specifies byte-by-byte filling: the number of digits in each hexstring must be even. **-W** specifies word-by-word filling: the number of digits in each hexstring must be divisible by 4, and the number of characters in each ASCII string must be even. The default for -B/-W is emulator-specific.
-N suppresses the read-back check.

g [**-r** / **-l**] [address]

Starts program execution at the specified address. If no address is given, execution starts at:

a. the address following the last instruction executed, if any; or

b. the transfer address, if it exists and the program has not already been started; or

c. address 0, if there is no transfer address and the program has not already been started.

-R displays a message when a break occurs and then repeats execution of the G command. **-L** repeats the G command, but suppresses trace lines and break messages. If -R and -L are omitted, execution stops at the first break.

lo <loadfile [parameters]
or
lo -o offset <loadfile [parameters]

Downloads object code from the specified host load file into program or prototype memory. Object code must be in A Series or B Series load module format. The offset, if specified, is added to the load address of each block of object code. The loaded program can use SVC 1C to access the optional user-defined parameters.

log device

Logs subsequent commands and system responses to the specified device. LOG CONO ends log activity. Not for use in TERM mode.

map option {loaddr [hiaddr]} ...

Changes the program/prototype memory designation and/or write protect status of the specified address range(s). Memory is mapped in 128-byte blocks for most microprocessors. **hiaddr** defaults to the upper boundary of the block that contains **loaddr**. Only program memory can be write protected by MAP. However, the protect status of an address block is independent of its program/prototype memory designation. The target processor must be selected before you use MAP.

MAP Option	Memory Designation	Protect status of Program Memory
P	program	unaffected
U	prototype	unaffected
PRW	program	read and write
PRO	program	read only
URW	prototype	read and write
URO	prototype	read only
RW	unaffected	read and write
RO	unaffected	read only

map [-m]

Displays the current memory map assignments in matrix format. Defaults to vertical tabular format.

mem [loaddr [hiaddr]]

Defines a block of prototype memory your program is allowed to access. Defaults to the entire memory range. For most emulators, the MEM command is not used, or is not available unless you also have the MAC option installed.
Entering MEM with no parameters displays the current status of memory.

memsp [**s** memspace | **m** memspace ...]

Specifies memory space to be used when no memory space is specified in a command line. **memspace** is a two-character symbol that represents a specific memory space. Entering MEMSP with no parameters displays the current default memory spaces.
For most emulators, the MEMSP command is not used or is not available unless you also have the MAC option installed.

The **M** parameter affects commands that accept more than one memory space command in the address expression. **S** affects commands that accept a single memory space in the address expression.

Commands Affected by M			Commands Affected by S			
AL	MAP	NOMEM	CPR	F	RPR	SVC
BK	MEM		D	MOV	S	WH
			DI	P	SAV	WPR
			EX	RD	SEA	WRT

mov { **uu** | **up** | **pu** | **pp** } loaddr hiaddr destaddr

Copies the contents of memory locations **loaddr** through **hiaddr** to the block beginning with **destaddr**.

MOV Option	Source Memory	Destination Memory
UU	prototype	prototype
UP	prototype	program
PU	program	prototype
PP	program	program

nomem [loaddr [hiaddr]]

Defines a block of prototype memory your program is **not** allowed to access. If NOMEM is not used, all memory is accessible.
For most emulators, NOMEM is not used and is not available unless you have the MAC option installed.
Entering NOMEM with no parameters displays the current status of memory.

p [**-b** | **-w**] [**-n**] address { hexstring | **-a** string } ...

Stores a sequence of bytes at the specified address in program/prototype memory. The byte sequence is obtained by concatenating all specified hexadecimal and ASCII strings.
-A specifies an ASCII string.

-B specifies byte-by-byte patching: the number of digits in each hexstring must be even. **-W** specifies word-by-word patching: the number of digits in each hexstring must be divisible by 4, and the number of characters in each ASCII string must be even. The default for -B/-W is emulator specific.

permstr stringname ...

Adds the specified string(s) to the permanent string storage area. An error message is displayed if you attempt to add a string that is already in the permanent string storage area, or if the permanent string storage area is full.

permstr -d stringname ...

Deletes the specified string(s) from the permanent string storage area.

permstr -l

Lists names and values of strings that are currently in the permanent string storage area.

pro [**-s** / **-c**] evt {value}

Defines a TTA event in terms of signals from the Data Acquisition Probe.
value can be a 2-digit hexadecimal number or eight individual bits, and may include don't-cares. Defaults to all don't-cares. The radix defaults to binary.
evt is 1, 2, 3, or 4.
-S sets a breakpoint at the associated trigger; **-C** continues program execution after the break display.

pro { **all** / evt } **clr**

Clears the previous probe value parameters for the specified TTA event(s).

pstat

Displays PROM Programmer status information.

ptype

Displays information about the types of PROMs supported by the currently inserted PROM Programmer characteristic module.

query [**on** | **off**]

Turns the system-wide query mode on or OFF. Entering QUERY with no parameters displays the current query status.

rd [**-s** | **-m**] [**-b** | **-w**] portnum ...

Reads a byte (**-B**) or a word (**-W**) from an I/O port on the emulator. Defaults to -B. Not available for most emulators.
-M specifies that **portnum** is a memory location. If -M is omitted, **portnum** is assumed to be a fixed port.
-S specifies special read. -S is not valid with -M.

rems [**-n** | **-q**] {sectionname:* | symbolspec} ...

Removes the specified symbol(s) from the symbol table. If **sectionname:*** is used, all symbols in the specified section are removed from the symbol table. If **symbolspec** does not include a section name, REMS searches for the symbol the current section, then in the entire user symbol table, and deletes the first occurrence of the symbol.
-Q directs OS/40 to query before removing the symbol. -N suppresses queries.

reset

Simulates a hardware reset signal to the emulating microprocessor. The target processor must be selected before you use RESET.

rh [**-i** | **-m** | **-t**] <loadfile [offset]

Downloads hexadecimal object code from the specified host load file into program/prototype memory. The offset, if specified, is added to the load address of each block of object code.

-I or **-M** specifies Intel or Motorola Load Module format, respectively.
-T specifies Standard Tekhex format. The default format is Extended Tekhex.

rompatch option ...

Patches the operating system. For information on command options, see the Command Dictionary section of your 8540 System Users Manual.

rompatch -d option ...

Deletes the most recent current patch from EEPROM.

rompatch -l

Displays a list of all patches currently saved in the EEPROM.

rpr [**-a**] [**-d**] [**-m** | **-l**] [**-r**] memlo promtype promlo promhi

Reads the contents of PROM addresses **promlo** through **promhi** into program memory beginning at address **memlo**.
-A specifies that the PROM requires complemented addresses.
-D specifies that the PROM provides complemented data.
-M specifies that data being read represents the most significant word halves; **-L** indicates least significant word halves.
-R specifies that bytes are to be arranged in "reverse" order, with the least significant byte first.

s symbolspec=expression ...

Assigns a value to the specified emulator register(s) or symbol(s).

sav [**-s** | **-l**] >loadfile {loaddr hiaddr} ... [transfer]

Writes the specified contents of program/prototype memory to the specified host file in A Series or B Series load module format. The transfer address defaults to **loaddr** of the first block.
-S specifies small address space (A Series) format. **-L** specifies large address space (B Series) format.

sea [**-b** | **-w**] [**-r**] loaddr [hiaddr] { value [precision] | **-a** string }

Searches program/prototype memory addresses **loaddr** through **hiaddr** for the specified value or ASCII string. **-A** specifies an ASCII string. **value** may be up to 4 bytes. Specify the precision if the value contains 1 to 3 bytes of leading zeros (or leading 1s, for negative values).
-B selects a byte-by-byte search; precision defaults to 1 byte. **-W** selects a word-by-word search; precision defaults to 2 bytes. The default for -B/-W is emulator-specific.
-R specifies that the search is repeated after each match is reported. If -R is omitted, the search stops after the first match.

sel [chip]

Selects the target processor, and automatically performs the following initialization commands:

```
AL 0 10 0          COU ALL CLR        RESET
BK ALL CLR         EM 0               SVC ON 40 0F0
BRE ALL CLR        EVE ALL CLR        SYMD -LS
CLOCK OFF          MAP PRW 0 OFFFF    TCLR -X
                                      TRA OFF
```

If you have the MAC option installed, SEL does not perform the AL initialization; instead it performs the commands **DEAL -A** and **MEM 0 himem**.
If the microprocessor name is omitted, SEL displays the name of the currently selected microprocessor.

stat

Displays the name of the currently selected microprocessor, the name of the last program loaded into program memory, and the I/O channel assignments.

str -l
or
str -d stringname ...

Displays or deletes a temporary string or strings. **-L** displays all currently defined temporary strings. **-D** deletes the specified temporary string(s).

susp { command / **-a** }

Suspends the specified command. The command remains suspended until you abort it (A command) or continue it (CO command). **-A** suspends all active commands.

svc [**on** / **off**] [address] [port]

Turns SVCs ON or OFF. **address** specifies the address of the SRB vector. **port** represents the lowest I/O port address used by your program to initiate SVCs. **port** must be a multiple of 10H and must not exceed 0F0H. If any parameter is omitted, the associated value is not changed. At system startup, the SRB vector address is 40H and the port range is F0-F7. EM 0 automatically turns SVCs ON; EM 1 or EM 2 automatically turns SVCs OFF. The target processor must be selected before you use SVC.

symb expression

Returns the symbol that matches the specified address. If no symbol corresponds to the expression value, SYMB returns the section name and byte offset of the value.

symd [-s] [-l] on

Turns on symbolic debug output. Symbols are substituted for their addresses.
-S enables offset substitution and turns label substitution off. **-L** enables label substitution and turns offset substitution off. **-SL** enables both. SYMD is automatically enabled as part of the SEL command.

symd off

Turns off symbolic debug output.

symlo [-**a**] [-**g**] [-**s**] <loadfile [sectionname] ...

Loads symbols from the specified load file into the symbol table, for use in symbolic debug. Input must be of the type produced by the TEKTRONIX A Series or B Series Linker.
-**A** appends new symbols to those already in the table. If -A is omitted, symbols already in the table are overwritten.
-**G** loads global symbols only.
-**S** loads scalars as well as addresses. Defaults to addresses only.
sectionname specifies the section of object code from which symbols are to be loaded. If no section names are specified, symbols from all sections are loaded.

tclr trig

Clears the EVE, COU, and BRE conditions for the specified TTA trigger(s). **trig** can be 1, 2, 3, 4, or **all**.

tclr -x

Resets the entire TTA: clears the EVE, COU, BRE, CONS, and ACQ conditions for all triggers. Does not alter the contents of Acquisition Memory.

tra [-**s**] [-**n** | -**l**] {**all** | **jmp** | **off**} [loaddr] [hiaddr]

Specifies the type of instructions to be traced in the specified address range. **loaddr** defaults to 0. **hiaddr** defaults to top of memory. Up to three TRA selections can be in effect at a time. The target processor must be selected before you use TRA.
-**S** stops execution after each trace line is displayed. If -S is omitted, execution continues after each trace display.
-**N** shows only the most important registers. -**L** displays all registers. If -L and -N are omitted, the default is to the most recent selection.
all displays all instructions; **jmp** displays only jump instructions; **off** disables trace display.

tra [-n / -l]

Changes the display format without affecting the trace selections.

tra

Displays the current TRA selections.

ts [-c / -e] trig ...

Displays the current status of the specified TTA trigger. Entering TS with no parameters displays the complete TTA status.
trig can be 1, 2, 3, or 4.
-C displays counter parameters only; **-E** displays event parameters only.

wh [-i / -m / -t] >loadfile {loaddr hiaddr} ... [transfer]

Writes the contents of the block(s) of program/prototype memory to the host load file in hexadecimal format. The transfer address defaults to 0.
-I or **-M** specifies Intel or Motorola Load Module format, respectively; **-T** specifies Standard Tekhex format. The default format is Extended Tekhex.

wpr [-a] [-d] [-m / -l] [-r] [-n] memlo promtype promlo promhi

Writes data from program memory, starting at **memlo**, into PROM addresses **promlo** through **promhi**.
-A specifies that the PROM requires complemented addresses.
-D specifies that the PROM requires complemented data.
-M specifies that data being written represents the most significant word halves; **-L** specifies least significant word halves.
-R specifies that bytes are to be arranged in "reverse" order: least significant byte first.
-N suppresses the erase check normally performed before writing.

Not supported in 280A

wrt [-s / -m] [-w / -b] portnum value

Writes a byte (-B) or word (-W) to an I/O port on the emulator. Defaults to -B. Not available for most emulators.
-**M** specifies that **portnum** is a memory address; if -M is omitted, **portnum** is assumed to be a fixed port.

x <loadfile [parameters]

Downloads the specified host load file into program or prototype memory and starts execution at the transfer address. The loaded program can use SVC 1C to access the optional user-defined parameters. The loadfile must be in A Series or B Series Load Module Format.

Section 3

SERVICE CALLS

OVERVIEW

A service call (SVC) is a request for OS/40 to perform a special service for an executing program. An SVC contains the following parts:

- An I/O instruction followed by one NOP instruction (in Modes 0 and 1), or by two NOPs (in Mode 2). The NOP allows time for OS/40 to perform the SVC. The port number of the I/O instruction selects one of eight Service Request Block (SRB) pointers. The standard SVC port range is F0–F7. You can use the SVC command to specify a different port range.
- An SRB pointer: the address of the first byte of an SRB. The SRB pointer is two bytes for SAS format, four bytes for LAS format. Eight SRB pointers form the SRB vector, which usually begins at address 40H. You can use the SVC command to place the SRB vector at a different location.
- A Service Request Block that contains the information OS/40 needs to perform the SVC. The SRB is eight bytes for SAS format, 12 bytes for LAS format. The first byte of the SRB contains the SVC function code that determines how OS/40 interprets the rest of the SRB.
- Buffers for data passed between OS/40 and the program. An SVC may use more than one buffer, or none at all, depending on the function.

Microprocessors that address a maximum of 64K bytes use Small Address Space (SAS) format for SRBs and SRB vectors. Microprocessors that address more than 64K use Large Address Space (LAS) format. Table 3-1 summarizes the differences in SRB formats. Figures 3-1 and 3-2 illustrate SAS and LAS memory layout for SVCs.

In SVCs, multi-byte values are stored with the most significant byte first and the least significant byte last. Any byte in an SRB not specifically designated to pass data may contain garbage when the SVC is completed.

Table 3-1

Summary of LAS and SAS Formats for SRBs

SRB Field Name	Bytes Used	
	Small Address Space Format (SAS)	Large Address Space Format (LAS)
Function	1	1
Channel	2	2
Status	3	3
Fourth Byte	4	4
Byte Count	5	5-6
Buffer Length	6	7-8
Buffer Pointer	7-8	9-12

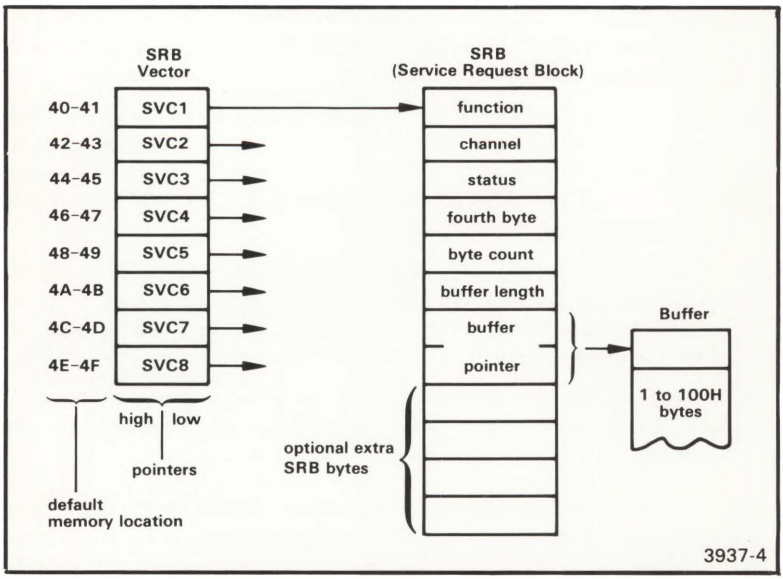


Fig. 3-1. Service call memory layout, SAS format.

An instruction sequence within your program determines which of the eight SRB pointers is used. The SRB addressed by the selected pointer contains the parameters needed to perform the SVC. Depending on the function specified in the first field of the SRB, an I/O buffer may be needed; in that case, the buffer length and buffer pointer fields indicate the length and location of the buffer.

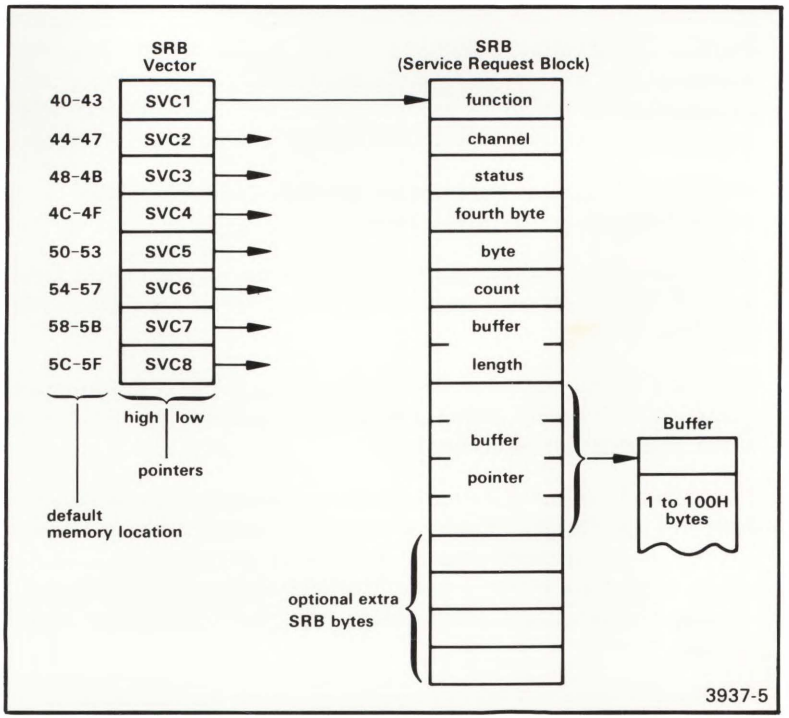


Fig. 3-2. Service call memory layout, LAS format.

An instruction sequence within your program determines which of the eight SRB pointers is used. The SRB addressed by the selected pointer contains the parameters needed to perform the SVC. Depending on the function specified in the first field of the SRB, an I/O buffer may be needed; in that case, the buffer length and buffer pointer fields indicate the length and location of the buffer.

SVC FUNCTION CODE DESCRIPTIONS

Byte 1 of an SRB always contains the SVC function code. The roles of the remaining bytes are described here for each function type.

01—Read ASCII and Wait. Reads an ASCII line from the designated channel. **Parameters passed:** Channel; Buffer length (maximum number of characters to read, including the RETURN character); Buffer pointer. **Information returned:** Byte count (number of characters read); Buffer (the ASCII line read, ending with RETURN character).

02—Write ASCII and Wait. Writes an ASCII line to the designated channel. **Parameters passed:** Channel; Buffer length (maximum number of characters to write, including the RETURN character); Buffer pointer; Buffer (line to be written, ending with RETURN character). **Information returned:** Byte count (number of characters written).

03—Close Channel. **Parameters passed:** Channel number to be closed. **Information returned:** none.

04—Rewind File. Moves the file pointer to the beginning of a file on the host. **Parameters passed:** Channel number of file. **Information returned:** none.

05—Delete File. Unlinks the filespec assigned to the designated channel and closes the channel. **Parameters passed:** Channel number of file. **Information returned:** none.

07—Special Function. Performs a device-dependent operation on the device or file assigned to the designated channel. The only special function code currently implemented is 14 (same as SVC 14, Get Device Type). **Parameters passed:** Channel; SAS byte 5, LAS bytes 5-6 (special function code). **Information returned:** depends on special function.

10—Assign Channel. Assigns a device or file (on the host) to the designated channel. If the specified file does not exist, it is created and status code 01 is returned. **Parameters passed:** Channel to be assigned; Buffer pointer; Buffer (filespec of file or device, ending with RETURN character). **Information returned:** none.

11—Read Program Clock. Reads the value in the 100-millisecond program clock. **Parameters passed:** none. **Information returned:** SAS bytes 4–5, LAS bytes 5–6 (clock value).

13—Get Command Line Parameter. Returns the ASCII representation of a selected parameter from the LO (Load) or X (Execute) command line that loaded the current program. Parameter number 0 is the filespec of the load file. Subsequent parameters are numbered with consecutive integers. **Parameters passed:** SAS byte 4, LAS bytes 5–6 (parameter number); Buffer length (maximum length expected); Buffer pointer. **Information returned:** Buffer (parameter, ending with RETURN character). For SAS format, the byte count is also returned.

@

14—Get Device Type. Returns two device-dependent values that define the type and general capabilities of the device or file assigned to the designated channel. These values are defined in Table 3-2. **Parameters passed:** Channel. **Information returned:** SAS byte 4, LAS byte 7 (device identification number); SAS byte 5, LAS byte 8 (device type code).

Table 3-2
Device Identification and Type

Name	Description	Device Identification	Type Code	Type Description
CONI	Console input	01	01	ASCII read
CONO	Console output	02	02	ASCII write
LPT	Line printer	03	02	ASCII write
PPTR	Paper tape reader	08	01	ASCII read
PPTP	Paper tape punch	09	02	ASCII write
REMI	Remote input	0A	01	ASCII read
REMO	Remote output	0B	02	ASCII write
(file)	File	FF	43	Binary read/write

16—Get Last CONI Character. Returns the last character typed at the system terminal. **Parameters passed:** none. **Information returned:** SAS byte 4, LAS byte 6 (character typed).

19—Suspend Program. Suspends the currently executing program. Use the G (Go) command to resume execution. **Parameters passed:** none. **Information returned:** none.

1A—Exit Program. Same as SVC 19 (Suspend Program).

1C—Get Execution Line Parameter. Same as SVC 13 (Get Command Line Parameter).

1F—Abort Program. Terminates the currently running program and closes all open channels. **Parameters passed:** none. **Information returned:** none.

22—Overwrite ASCII and Wait. Same as SVC 02 (Write ASCII and Wait).

24—Seek Relative to Byte in File. Moves the file pointer the specified number of bytes relative to its current position. If the number is negative, the file pointer is moved toward the beginning of the file. **Parameters passed:** Channel; SAS bytes 5–8, LAS bytes 9–12 (number of bytes to move the file pointer: a signed four-byte value). **Information returned:** SAS bytes 5–8, LAS bytes 9–12 (new position of file pointer).

30—Open for Read. Assigns a channel to the device or file (on the host) and designates the channel as read-only. **Parameters passed:** Channel; Buffer pointer; Buffer (filespec of file or device, ending with RETURN character). **Information returned:** none.

41—Read Binary and Wait. Reads binary data from the designated channel. **Parameters passed:** Channel; Buffer length (number of bytes to read); Buffer pointer. **Information returned:** Byte count (number of bytes read); Buffer (data read).

42—Write Binary and Wait. Writes binary data to the designated channel. **Parameters passed:** Channel; Buffer length (number of bytes to write); Buffer pointer; Buffer (data to be written). **Information returned:** Byte count (number of bytes written).

44—Seek to Byte in File. Moves the file pointer to the specified byte of a file on the host computer. Byte number is given as a signed 4-byte value. Byte 0 is the first byte in the file. **Parameters passed:** Channel number of file; SAS bytes 5–8, LAS bytes 9–12 (byte position to seek). **Information returned:** SAS bytes 5–8, LAS bytes 9–12 (new byte position).

4D—Unlink. Deletes a single filespec on the host. **Parameters passed:** Buffer pointer; Buffer (filespec to be deleted, ending with RETURN character). **Information returned:** none.

50—Open for Write. Assigns a channel to the device or file (on the host) and designates the channel as write-only. **Parameters passed:** Channel; Buffer pointer; Buffer (filespec, ending with RETURN character). **Information returned:** none.

62—Overwrite Binary and Wait. Same as SVC 42 (Write Binary and Wait).

64—Seek to Byte in File Relative to EOF. Positions the file pointer the specified number of bytes (zero or negative) from the end of the file (on the host). **Parameters passed:** Channel number of file; SAS bytes 5–8, LAS bytes 9–12 (a signed four-byte value that specifies the desired position of file pointer, relative to EOF). **Information returned:** SAS bytes 5–8, LAS bytes 9–12 (new byte position of file).

70—Open for Read and Write. Assigns an existing device or file (on the host) to the designated channel. **Parameters passed:** Buffer (filespec, terminated by a RETURN character); Channel; Buffer pointer. **Information returned:** none.

81—Read ASCII and Proceed. Same as SVC 01 (Read ASCII and Wait).

82—Write ASCII and Proceed. Same as SVC 02 (Write ASCII and Wait).

90—Create File. Creates an empty file on the host and assigns a channel to the new file. If the specified file already exists, its contents are deleted. **Parameters passed:** Channel to be assigned; Buffer pointer; Buffer (filespec of file to be created, ending with RETURN character). **Information returned:** none.

A2—Overwrite ASCII and Proceed. Same as SVC 02 (Write ASCII and Wait).

C1—Read Binary and Proceed. Same as SVC 41 (Read Binary and Wait).

C2—Write Binary and Proceed. Same as SVC 42 (Write Binary and Wait).

E2—Overwrite Binary and Proceed. Same as SVC 42 (Write Binary and Wait).

Section 4

INTERSYSTEM COMMUNICATION

Overview

Intersystem communication is the general process of communication between an 8540 and an external host computer.

A TERM interface is a mode of communication between an 8540 and a TEKTRONIX 8560 Multi-User Software Development Unit. A TERM interface provides simultaneous access to the 8540 and 8560, and allow you to execute both 8540 and the 8560 commands. You initiate a TERM interface by entering the OS/40 command CONFIG TERM, along with the desired parameters. For more information on a TERM interface, refer to the 8560 System Users Manual.

A COM interface is a mode of communication between an 8540 and a host computer. With a COM interface, you can execute host commands from the 8540 system terminal and transfer Tekhex-formatted object code between the 8540 and the host. You initiate a COM interface by entering the OS/40 COM command, along with the COM parameters that specify the protocol that is appropriate for your host.

COM Interface

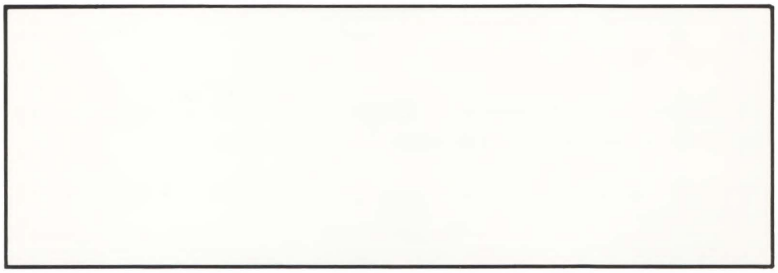
Setup

You can use this space to document the setup procedure for your host computer.

Establishing Communication

Enter the COM command on the 8540's system terminal. Then log on to the host computer.

You can use this space to document the COM parameters and log-on commands required by your host.

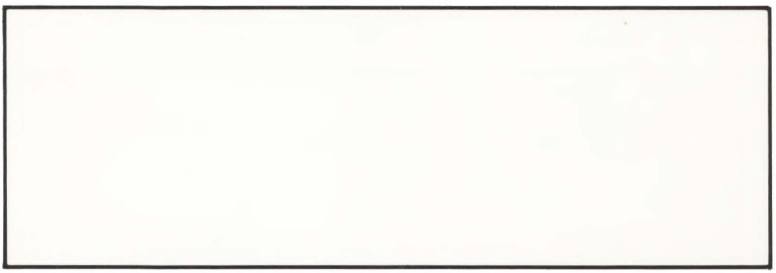


Performing a Transfer

Download. To download a Tekhex load module from the host to the 8540, enter a command line with the following elements:

1. the host command that executes the program you've written to download a Tekhex load module
2. the null character (CTRL-@ on most terminals)
3. a carriage return.

The object code in the load module is loaded into 8540 program memory, and the program symbols are placed in the symbol table in 8540 system memory.



Upload. To upload one or more blocks of 8540 program memory to a file on the host, enter a command line with the following elements:

1. the host command that executes the program you have written to upload Tekhex message blocks
2. a null character (CTRL-@ on most terminals)
3. one or more pairs of addresses representing blocks of program memory to be uploaded. The first address in each pair represents the low end of the block, and the second address marks the high end.
4. (optional) the transfer address of the program being uploaded
5. a carriage return.

The object code is uploaded in Tekhex message blocks. Program symbols are not uploaded.

Terminating Communication

1. Press the BREAK key to abort any data transfer that is taking place.
2. Log off from the host computer.
3. Enter (null)(esc) to exit from the COM subsystem.

You can use this space to document the log-off procedure required by your host.

Section 5

ERROR MESSAGES

NOTE

Any of these codes may be returned as an SRB status code after an SVC. Refer to the Service Calls section of this manual for information regarding SRB status codes.

02—**Illegal channel number**. The channel number used was outside the range 0–9.

03—**I/O channel not open**. An I/O operation has been attempted on a channel that has not been assigned to a file or device.

05—**Illegal function for device**. An attempt has been made to perform an illegal function, such as reading from the line printer or writing to the paper tape reader.

06—**Short or unterminated read**. The number of bytes read was less than the number of bytes requested, or a carriage return was not detected. This is the normal status when the last (short) block of a file is read.

07—**Short or unterminated write**. A carriage return was not detected in the specified number of characters to be written.

0A—**Device not operational**. Power to the device is on, but for some reason it cannot function properly. For example, the line printer may be out of paper.

0C—**Device not ready**. Power to the device you wish to access is not on.

0D—**Device in use**. An attempt has been made to assign a non-shareable device on a second channel. Only CONI and CONO are shareable.

10—**Error reading disk bit map**. An error in reading a disk bit map has been detected. Host-dependent SVC error.

11—I/O error or access violation on read. An error on REMI, CONI, or PPTR has been detected, or you do not have read access to the directory or file on the host.

12—I/O error or access violation on write. An error on REMO, CONO, or PPTP has been detected, or you do not have write access to the directory or file on the host.

13—Command not found. OS/40 does not recognize the command name. Be sure that the proper emulator software has been installed and selected.

16—Illegal file specification. A filespec may be an incorrect length or contain illegal characters. May be host dependent.

17—Illegal SVC function code. An SVC code has been specified that does not exist or is not available to the user.

18—Channel already open. Self-explanatory.

19—Volume or disk not found. A file has been specified on a volume that is not mounted on the host. Host-dependent SVC error.

1B—Checksum error. A checksum error in a load file has been detected by the LO or RH command.

1C—Command area in use. Commands may not be executed simultaneously.

1D—File not found. An attempt has been made to open a file for reading or writing, but the file either does not exist, or exists in a protected directory. Host-dependent SVC error.

1E—Invalid parameter. OS/40 does not recognize one of the parameters entered in a command line. Check the required format and parameters in the Command Dictionary.

1F—No header on load file. An attempt has been made to load a file that is not in load module format.

20—Invalid input parameter. An invalid input parameter has been entered. Refer to the Command Dictionary for the correct syntax.

21—Invalid output parameter. An invalid output parameter has been entered. Refer to the Command Dictionary for the correct syntax.

22—No transfer address. This status code is returned when you use SVC 17 or 57 to load a program that does not have a transfer address.

23—Command buffer overflow. The command line entered (or the resulting line after all string and parameter substitution) must be less than or equal to 80 characters, including the carriage return.

24—Symbol table full. An attempt has been made to load more symbols into the symbol table (with COM, SYMLO, or ADDS) than allowed. The number of symbols possible is about 1000, depending on symbol length. Use REMS to remove symbols.

25—Data format error. The input data is not in the format expected by the command.

26—No emulator in system. The emulator hardware is not installed.

27—Numeric parameter out of range. Either the clock count or the number of lines to be printed has been set to a value that is outside the range 0–64K.

28—System interface error. Serious problems have been detected in the host interface. Check your 8560 or host cable connections and interface baud rate. If this problem continues, contact your Tektronix service representative.

29—Seek error on file I/O. An attempt has been made to seek past the end (or before the beginning) of a file. The seek stops at the end (or beginning) of the file.

2A—Parameter required. A command line has been entered that requires another parameter. Check the required parameters in the Command Dictionary.

2B—Too many parameters. A command line has been entered that contains more parameters than required. Check the required format in the Command Dictionary.

2C—Invalid address parameter. An address parameter may contain numbers, register names, symbols, memory space designators, don't-care values, and the operators + and −. Refer to the Command Dictionary for more information regarding valid address expressions.

2E—System must be idle to SELect. The system must not have any active commands or programs. SEL destroys the current program debugging status.

2F—Buffer overflow on HSI operation. This error indicates serious hardware or software problems. The 8540 Installation Guide contains verification procedures.

30—Invalid address range. An invalid address range has been entered. The high address must be greater than or equal to the low address.

32—Too many trace ranges. An attempt has been made to set a fourth TRA command. Only three TRA commands may be active at one time. Check the Command Dictionary for correct parameters.

34—Command not active. An attempt has been made to abort or suspend an inactive command.

35—Command not suspended. An attempt has been made to continue a command that is executing or that has finished executing.

37—Invalid hexadecimal string. The commands EX, F, and P accept a hexadecimal string. Refer to the Command Dictionary for the requirements of a hexadecimal string.

38—Missing close quotation mark. A quoted string must have opening and closing quotation marks.

39—No emulator selected. An attempt has been made to execute the G or X command before an emulator has been selected.

3A—Bad acknowledge. The valid acknowledgments are "0" (ASCII 30H) for ACK and "7" (ASCII 37H) for NAK.

3B—Transfer aborted. A data transfer between the 8540 and the host was incomplete, so the transfer was aborted. Check cable connections, and power switches and connections. If the problem persists, run the system verification described in the 8540 Installation Guide and/or have a qualified service representative run diagnosics on the systems communications hardware as described in the 8540 Service Manual.

3C—Memory write error. Data written to memory could not be read back. The memory may be read-only, or there may be no RAM for the addresses specified.

3E—Invalid memory space designator. A memory space designator (such as SC:) has been incorrectly entered.

3F—Illegal use of don't-care expression. A don't-care expression has been used where a unique value is required.

40—Memory space designator illegal in expr. A memory space designator has been used in a parameter that does not allow memory space designators. For example, in a pair of parameters that represent an address range, only the first may contain a memory space designator.

41—Word not filled. The string entered with the F or P command did not fill an even number of bytes.

42—Invalid use of multiple memory spaces. Multiple memory spaces can only be used with the commands listed under MEMSP in the Command Dictionary.

43—No Trigger Trace Analyzer in system. The TTA hardware has not been installed.

44—Emulator clock missing. Make sure the prototype power switch is on and that the prototype clock is functioning correctly.

45—Emulator faulted. The emulator is malfunctioning. Error code 5F may also occur under this condition. Have a qualified service representative check the hardware and fuses, and clean the board contacts. Make sure that the prototype power switch is on, that the prototype clock is functioning properly, that the entered commands are valid, and that the instructions being executed are legal.

46—Odd word address. An attempt has been made to modify an area of memory that does not begin on a word boundary.

47—Byte not filled. An uneven number of hexadecimal digits has been entered in an attempt to modify memory with the F or P command. You cannot modify a half byte.

48—Port has no carrier signal. No carrier signal has been detected at the REMI/REMO port.

49—Port parity error. A parity error has been detected at the REMI/REMO port.

4A—Port framing error. A framing error has been detected at the REMI/REMO port. That is, the number of start, stop, and data bits received was not what was expected. The COM -M command can be used to set the desired framing.

4B—All job streams active. The system is busy. Commands cannot be entered until one of the currently executing commands is finished.

4D—**PROM power failure.** The power to the PROM Programmer has failed.

52—**Command busy.** An attempt was made to enter a command again when it was already executing.

53—**Symbol not found.** An attempt has been made to access a symbol that does not exist or has been entered incorrectly. Valid symbols include register names and any program symbols you have placed in the symbol table. Be sure to include a leading zero on any hexadecimal number that starts with a letter, and on any number that begins with an X (don't-care).

54—**Invalid symbol.** An invalid character has been detected in a symbol, or the symbol is too long.

55—**Symbol value not alterable.** The symbol to which you are attempting to assign a value with the S command is not alterable.

56—**Truncation error.** An attempt has been made to assign too large a value to a symbol or register, or the hex string specified in the F (Fill) command does not evenly fill the specified memory area.

57—**Invalid arithmetic operator.** The only valid arithmetic operators are + and - .

58—**Invalid term in expression.** An invalid character or character string has been detected in an expression.

59—**Overflow in expression.** The resulting value of an expression (or some intermediate value obtained in evaluating it) is too large. Expressions are evaluated using 32-bit arithmetic.

5A—**Invalid dash modifier.** Dash modifiers must be letters only. Refer to the Command Dictionary for the dash modifiers accepted by each command.

5C—**Too many files open.** No more than eight channels to files can be simultaneously open by commands and user programs. Enter the command **A -A** to close all channels.

5D—**Bad character in number.** The valid digits are 0-9, A-F, and the standard suffixes: H (hexadecimal), T (decimal), O or Q (octal), and Y (binary).

5F—Emulator halted. The emulator halted while a program was executing. Error code 45 (Emulator faulted) may also occur under this condition. Make sure that the appropriate emulator board is installed and the instructions entered are valid.

60—Emulator SVC synchronization error. A serious emulator software error has been detected. Reboot and reselect the emulator. If the problem persists, contact your Tektronix service representative.

61—Disk protected against writing. A disk on the host is write-protected. If you are writing to the disk, place a tab over the write-protect slot. Host-dependent SVC error.

63—Disk not formatted. A disk on the host does not have the proper sector or timing marks. Format the disk to prepare it for use. Host-dependent SVC error.

64—Disk CRC error. A bad block (parity error) has been detected on a disk on the host. Host-dependent SVC error.

65—Disk full. There are no blocks available for allocation on this host disk, or there are no free files. Host-dependent SVC error.

66—System synchronization error. The 8540 and the host have serious interface problems. Reboot your system and try again. Contact your Tektronix service representative if problems continue.

67—Exclusive access conflict. An attempt has been made to open a host file or device that is already being used exclusively by another process. Host-dependent SVC error.

69—Too many channels open. Serious hardware and/or software problems have been detected. Reboot your system and try again. Contact your Tektronix service representative if problems continue.

6A—Disk structure corrupt. Serious problems have been detected in the file structure of your host's disk. Host-dependent SVC error.

6B—Current user is not file owner. An attempt has been made to access a file on the host to which you do not have access privileges. Host-dependent SVC error.

6E—Directory alteration invalid. An attempt has been made to create, delete, or rename a file on the host in a directory to which you do not have write access, or to create a duplicate name within a directory. Host-dependent SVC error.

6F—**Invalid file linkage attempt**. Host-dependent SVC error.

70—**File full**. The file structure of a volume on the host is full. Host-dependent SVC error.

71—**String already exists**. Using the PERMSTR command, an attempt has been made to store a permanent string in EEPROM when a string by that name is already stored.

74—**Program memory jumpered incorrectly**. Using the 68000 or Z8001, the SELect command cannot set up the MAC board properly since program memory has been strapped so that addresses do not have a unique location.

76—**ASCII read to CONI terminated by CTRL-C**. CTRL-C was typed while OS/40 was performing a read operation. The data read is not valid.

77—**Emulator SVC request outstanding**. A prior SVC is still in progress. Your current request will be filled when the first SVC has finished.

78—**No more program memory available**. An attempt has been made to ALlocate program memory when there is none available. Use DEAL to deallocate memory.

79—**Program memory address not allocated**. In order for you to access the specified address range, memory must be allocated to it using the AL command.

7A—**Program memory address already allocated**. An attempt has been made to ALlocate an address that has already been allocated.

7B—**String not found.** OS/40 does not recognize the string name entered. Check that you have spelled the name correctly. Use the STR -L command to list temporary strings and/or the PERMSTR -L command to list permanent strings.

7C—**String area full**. The capacity of either the temporary or the permanent string area has been exceeded. Use the STR or the PERMSTR command to delete unused strings.

7D—**System memory parity error**. Reboot the system. If the problem persists, contact your Tektronix service representative.

7E—**Error in command execution**. A command has been executed that detected errors but continued.

80—PROM type not supported. The characteristic module currently installed does not support the PROM type entered.

81—Maximum PROM address exceeded. The address given is negative, or exceeds the maximum for the PROM type.

82—PROM Programmer PTYPE data error. System error. The PROM type specified in a RPR, WPR, or CPR command is not supported by the characteristic module currently installed.

83—Modifier required. The command entered needs a dash modifier.

84—PROM Programmer hardware I/O error. System hardware error. Contact your Tektronix service representative if the error persists.

85—Allocation hardware disabled. Program memory allocation hardware error. Contact your Tektronix service representative if the problem continues.

87—Invalid trigger number. There are four TTA triggers, numbered 1-4.

88—Signals cannot occur simultaneously. Using either the TTA, or a Z8001/Z8002, 8086, or 68000 emulator, an attempt has been made to set an event or breakpoint on bus signals that are mutually exclusive (such as a read and a write on the same line).

8A—Invalid event linkage (wraparound). Using CONS, an attempt has been made to completely link together all events. Thus, no trigger can occur.

8B—Restart requires gate option. In the COU command, the restart option must be used in conjunction with a gate option.

8C—Restart/gate not available on trigger 1. The trigger 1 hardware has no prior trigger channel from which to gate.

8D—No section contains specified address. The ADDS command requires that any address you add be contained within a previously defined program section. Use the COM or SYMLO command to download the section definition information into the symbol table from your load file.

8E—Segmentation trap pending. Either your program or a system program has attempted to access memory which is invalid according to the Z8000 Memory Management Unit. Refer to the Z8001/Z8002 Emulator Specifics supplement for further information.

8F—User memory declared non-existent. An attempt has been made to access memory which was declared non-existent with the NOMEM command. Check memory declarations with the MEM or NOMEM command. If the problem persists after checking your program, check your MAC board.

90—Invalid arming mode. The -A arming modifier needs two programmed breakpoints, but only one is currently programmed. This error occurs only when using an emulator such as the Z8001/Z8002, 8086, or 68000.

91—Invalid initial value for counter. Zero is an invalid initial value for decrementing counter modes.

92—No such label or scalar. An attempt has been made to remove a symbol which is non-existent or is a section name.

93—Invalid symbol specification. In a symbol specification of the form section:label, either the section name is too long, the label name is too long, or the specification contains an illegal character.

94—No prototype control probe attached. An attempt was made to change from emulation mode 0 to mode 1 or 2 while the prototype control probe was not attached. This error contains an illegal character.

95—Prototype not ready. When the prototype microprocessor attempted to access prototype memory, the prototype held the READY line "not ready" for too many wait states. Check the prototype. The number of wait states allowed is jumper selected. This error occurs only with an 8086 emulator.

96—Prototype bus hang. The prototype has held the bus for an inordinate length of time. Check the prototype. Detection of this condition is jumper selectable; you may hold the bus for any length of time if the jumper is not used. This error occurs only with an 8086 emulator.

D4—Internal parse error. Serious software errors have been detected. Contact your Tektronix service representative if this problem continues.

D7—Internal term error. Serious software errors have been detected. Contact your Tektronix service representative if this problem continues.

E1—**Emulator double fault or odd stack pointer**. On the 68000, the emulator has halted during a user job. Possible causes are a double address or bus error, or an odd system stack pointer. Reset the registers and check the program and prototype.

E2—**Processor registers changed**. Following a 68000 processor halt, the emulator had to reset the PC, SSP, and SR registers before all the registers were saved.

E4—**Emulator system error**. Unknown emulator error. Reboot and reselect.

E6—**No MAC board in system**. No Memory Allocation Controller board has been installed.

E7—**System error on MAC board**. Unknown system error. Reboot and reselect. If the problem persists, contact your Tektronix service representative.

FE—**Process aborted**. This message is returned when the A command is used.

FF—**End of file**. Returned on a Read SVC if the file was at end of file before the read occurred. May be host dependent.

Section 6

GLOSSARY

Acquisition Memory. The buffer in the TTA that holds the 255 bus transactions most recently captured during program execution. This buffer stores up to 62 bits of information from each bus transaction.

ASCII Transfer. A type of data transfer performed with SVCs, in which the unit to be transferred is a line of ASCII characters terminated by a carriage return. See also **Binary Transfer**.

Assign. To associate a channel (using an SVC) with a device or host file.

Binary Transfer. A type of data transfer performed with SVCs, in which the unit to be transferred is a block of binary data. See also **ASCII Transfer**.

Break. A suspension of program execution by OS/40, accompanied by a display of the status of the emulator. A break may be set using a OS/40 BK command or one of several TTA commands. A break may also result from a special action, such as attempting to write to protected memory or typing CTRL-C.

Breakpoint. A program instruction at which a break is set. You can set breakpoints by using the TTA or the OS/40 BK (Breakpoint) command. The TRA -S command causes a break to occur each time a trace line is displayed.

Buffer. An area of memory where a block of data may be stored.

Bus Operation. A transfer of information between a microprocessor and a memory or I/O device. The four basic types of bus operations are memory read, memory write, I/O read, and I/O write. Some emulators recognize other types of bus operations.

Channel. The logical link between a device or a host file and the 8540 operating system. Channels are numbered 0–9. Channels 0–7 are user-defined; channels 8 and 9 are assigned to standard input and standard output, respectively. The AS (Assign) command assigns a channel for use by a file or device.

Characteristic Module. A circuit card that configures the PROM Programmer for a particular set of PROM devices.

Close. To disassociate a channel from a device or host file.

COM Interface. A communications interface between an 8540 and a host computer, established via the OS/40 COM command. Tekhex-formatted object code can be transferred between the host and the 8540, and host commands are executed from the 8540 system terminal as if the 8540 were not present.

Control Character. A character whose ASCII code is in the range 00 to 1F hexadecimal. RUBOUT (ASCII code 7F) is also a control character. Some control characters are entered using special keys, such as TAB or RETURN. Others are entered by pressing the CTRL key and some other key at the same time.

Counter. See **General Purpose Counter**.

Disassemble. To translate machine language back into assembly language mnemonics. The DI command performs disassembly.

Download. To transfer data from a host to an 8540.

EEPROM. Electronically erasable PROM. EEPROMs on the 8540 contain user-defined strings saved with the PERMSTR command, and system patch information created with the ROMPATCH command.

Emulating Microprocessor. The microprocessor on which your program executes during emulation. The emulating microprocessor resides on the emulator board or in the prototype control probe, and is usually the same type of microprocessor as the one being emulated.

Emulation. Performing the functions of a microprocessor, at or near the microprocessor's execution speed, with controllable hardware that facilitates debugging and testing of the microprocessor software in the prototype hardware. Emulation features of the 8540 include breakpoints, tracing, and event timing.

Emulation Mode. An 8540 operating mode in which the 8540 can provide some of the hardware functions needed by the microprocessor-based program. The hardware functions that can be provided are memory, a clock, and I/O facilities. The three modes are:

- Mode 0: System mode. Your program uses program memory and the emulator clock and uses SVCs for I/O. Until the prototype is built and connected to the emulator hardware, the program may execute only in mode 0.
- Mode 1: Partial emulation mode. Your program uses the prototype's clock, and may access both program and prototype memory: the memory map determines whether a particular address refers to program or prototype memory. Some emulators may also use SVCs.
- Mode 2: Full emulation mode. Your program uses the prototype's memory, clock, and I/O facilities. Some emulators may also use SVCs.

Emulator (or Emulator Processor). A circuit board in the 8540 that emulates the microprocessor that will drive your prototype hardware. You may use the emulation and debugging features of the emulator to test the software that will run on the prototype and to integrate the software and hardware components of the prototype.

Event. The simultaneous occurrence of one or more specified conditions that can be detected by the TTA. An event is defined in terms of values on the address bus, data bus, and certain other lines connected to the emulating microprocessor or prototype.

Extended Tekhex. See **Tekhex**.

File Pointer. A logical position in a file, maintained by OS/40 for use in processing SVCs. All reads from the file and writes to the file are performed beginning at the current file pointer. The file pointer is updated to the end of the data item read or written. The file pointer may also be moved using Seek SVCs. An independent file pointer is maintained for each channel.

Filespec. A sequence of characters that specifies a host file or an 8540 device.

Full Emulation Mode. See **Emulation Mode**.

General Purpose Counter. One of four registers in the TTA that increments or decrements while the emulator is running. You may use these counters to measure the execution time of a program segment, to measure the time between two specified events, or to count the occurrences of an event.

Host. A separate computer system that is used to prepare and maintain programs that are tested and debugged on an 8540.

Instruction. A **machine** instruction is a sequence of bytes that directs a microprocessor to perform an elementary operation such as load, store, add, or branch. An **assembly language** instruction is an alphanumeric representation of a machine instruction. The assembler translates an assembly language instruction into the corresponding machine instruction.

Instruction Fetch. A bus operation during which the first byte of the next instruction to be executed appears on the data bus.

Intersystem Communication. A process by which the 8540 exchanges information with a host computer system (such as an 8550 or 8560) via cable or phone line.

K. 1024 bytes (400 hexadecimal).

Load. To copy an executable program from a file into memory.

Load Module. A collection of executable object code suitable for loading into program memory. A load module may be in binary format, as produced by a linker or the SAV command, or it may be in a hexadecimal format. The Tables section of your 8540 System Users Manual presents information on hexadecimal formats recognized by OS/40.

LOCAL Mode. Stand-alone mode for an 8540, in which each command entered in interpreted by the 8540 itself. LOCAL is the usual operating mode. See also **TERM Interface**.

MAC. see **Memory Allocation Controller.**

Memory Allocation Controller. A hardware option that permits the expanded addressing capabilities of emulators such as the Z8001/Z8002 and 68000 to operate within the confines of 8540 program memory. For some emulators, you must have the MAC option installed in order to use the AL (ALlocate), DEAL (DEALlocate), MEM (MEMory), and NOMEM (NOMEMory) commands.

Memory Map. An internal table maintained by OS/40 that indicates which portions of memory used by the emulator are in program memory and which are in prototype memory. The memory map also indicates which parts of program memory are protected from write operations during program execution. Memory may be mapped in blocks as small as 128 bytes. Use the MAP command to change or display the memory map.

OS/40. The ROM-based operating system of the 8540 Integration Unit.

Partial Emulation Mode. See **Emulation Mode.**

Patch. To alter a program by changing the executable object code rather than the source code.

Program Clock. A counter in the 8540 that increments every 100 milliseconds while the emulator is running. You may use the CLOCK command to initialize or display the value in the program clock. This value is also accessible through the "Read Program Clock" SVC.

Program Memory. Memory in the 8301 that is used as a substitute for prototype memory in the early stages of prototype development (emulation modes 0 and 1).

Program/Prototype Memory. This term refers either to program memory or to prototype memory, depending on the current emulation mode and memory map. In mode 0, this term always refers to program memory. In mode 2, this term always refers to prototype memory. In mode 1, the memory map dictates which address ranges are in program memory and which are in prototype memory.

PROM. Programmable Read-Only Memory. Nonvolatile read-only memory that is blank when it is manufactured and stores whatever information is written to it by a PROM Programmer.

PROM Programmer. An optional device that writes data from memory to a PROM chip or reads data from a PROM chip into memory. You must have a PROM Programmer in order to use the following commands: CPR, RPR, WPR, PSTAT, and PTYPE.

Prototype. The microprocessor-based device that you are developing using the 8540.

Prototype Control Probe. A probe that takes the place of the prototype microprocessor and connects the prototype to the appropriate emulator hardware in the 8540.

Prototype Memory. Memory that resides in the prototype being developed.

Rewind. To position a file pointer to the beginning of a file, so that information in the file can be re-processed. See also **Seek**.

Seek. To position a file pointer to a given location in a file. Your program can use Seek SVCs to select any position in the file at which to read or write data. See also **Rewind**.

Service Call (SVC). A request for OS/40 to perform a specified I/O or maintenance function for an executing program. System programs use SVCs, as do most user programs that do not rely entirely on prototype I/O. The Service Calls section of your 8540 System Users Manual explains how to set up and initiate a service call, and describes each type of service call.

Slave Computer. An MDL that is connected to a host computer.

Standard Input. The file or device from which a command takes its input. Defaults to the system terminal unless you redirect input by using a <**filespec** parameter in the command line.

Standard Output. The file or device to which a command directs its output. Defaults to the system terminal unless you redirect output by using a >**filespec** parameter in the command line.

Standard Tekhex. See **Tekhex**.

String. A sequence of ASCII characters. OS/40 allows you to assign a name to a string and then refer to the string by name in a command line. If a string contains a delimiter such as a space, comma, or semicolon, the string should be enclosed in single or double quotes. If it contains a dollar sign, backslash, or double quote, the string should be enclosed in single quotes.

Permanent strings, created with the OS/40 PERMSTR command, are stored in the 8540's EEPROM storage area and thus are available whenever the system in powered up or restarted.

SVC. See **Service Call.**

Symbol. A string of up to 16 characters that begins with a letter and contains only letters, digits, periods, underscores, or dollar signs. In an assembly language program, predefined symbols include assembler directives and functions, mnemonics, and register names; user-defined symbols represent addresses, data items, variables, macros, sections, or modules.

Symbol Table. A table in system memory that contains program symbols and their values. This table is used in symbolic debug. Use the SYMLO or COM command to place symbols from a load file in the symbol table. Use the ADDS command to create other symbols.

Symbolic Debug. The use of symbols in place of expressions or hexadecimal numbers during debugging. You can use symbols as parameters in any OS/40 command that accepts expressions as parameters. During output, each hexadecimal address is replaced with either a symbol or an offset relative to the start of a program section. Some emulators do not support symbolic debug: refer to your Emulator Specifics supplement for this information.

System Memory. Memory in the 8540 that is not accessible to the user. Most OS/40 commands execute in system memory.

System Mode. See **Emulation Mode**.

System Terminal. The CRT terminal or other RS-232-C-compatible I/O device through which you communicate with the 8540. Device names for input and output through the system terminal are CONI and CONO, respectively.

Target Processor. The microprocessor that the 8540 is to emulate.

Tekhex. Tektronix Hexadecimal Format: a format for representing the contents of a block of memory as an ASCII sequence of hexadecimal digits. Checksums in the Tekhex format permit verification of Tekhex data transmitted from one computer to another. The WH command can be used to write memory contents to a file or device in Tekhex format; similarly, RH can be used to read Tekhex data from a file or device into memory. There are two forms of Tekhex: Standard Tekhex and Extended Tekhex. Both forms of Tekhex are described in the Intersystem Communication section of your 8540 System Users Manual.

TERM Interface. A mode of communication between an 8540 and an 8560 Multi-User Software Development Unit. In TERM mode, you can intermix OS/40 commands and TNIX commands, execute OS/40 commands from TNIX command files, and transmit data between the 8540 and 8560.

TNIX. The operating system of the TEKTRONIX 8560 Multi-User Software Development Unit.

Trace. To monitor the execution of a program by displaying the processor status each time a specified type of instruction is executed. The TRA command specifies the type and/or range of instructions to be displayed.

Transfer Address. The address of the first machine instruction to be executed in a program.

Trigger Trace Analyzer (TTA). An optional hardware device that enhances the 8540's debuging capabilities. The TTA allows you to capture and store up to 255 bus transactions that precede or follow a selected event in the executing program. The TTA includes four general purpose counters and four triggers that can break program execution, start or stop counters, or signal other instruments.

You must have a TTA installed in your 8540 in order to use any of the following commands: ACQ, AD, BRE, BUS, CONS, COU, CTR, DATA, DISP, EVE, PRO, QUA, TCLR, TS.

Upload. To transfer data from the 8540 to a host.

Section 7

INDEX

A

A command, 2-7
abbreviation of a parameter, 2-6
abortable commands, 2-7
ACQ command, 2-7
Acquisition Memory, 6-1
AD command, 2-7–2-8
ADDS command, 2-8
AL command, 2-8
apostrophe, 2-4
arrows (< >), 2-4
AS command, 2-8
ASCII transfer, 3-3, 3-4, 6-1

B

Backslash (\), 2-4
BACKSPACE key, 2-1
binary transfer, 3-6
BK command, 2-9
braces ({ }), 2-6
brackets ([ ]), 2-6
BRE command, 2-9
break, 6-1
buffer (for an SVC), 3-1–3-3
BUS command, 2-9

C

CALC command, 2-10
capitalization, 2-4
channels. See I/O channels

characters, special:
apostrophe, 2-4
arrows (< >), 2-4
backslash (\), 2-4
braces ({ }), 2-6
brackets ([ ]), 2-6
dash (-), 2-3
dollar sign ($), 2-4
double quote ("), 2-4
null, 4-2, 4-3
periods (...), 2-6
question mark (?), 2-4
semicolon (;), 2-3
single quote ('), 2-4
See also Control characters
CL command, 2-10
CLOCK command, 2-10
Close SVC, 3-4
CO command, 2-10
COM command, 4-1
COM interface, 4-1
commands, 2-1
abortable commands, 2-7
command line, 2-3
command modifiers, 2-3
continuing a suspended command, 2-10
parameters, 2-3
suspending a command, 2-25
syntax notation conventions, 2-5–2-6
CONFIG command, 2-12
CONI, 1-2, 3-5
CONO, 1-2, 3-5
CONS command, 2-12
control characters, 2-1
null, 4-2, 4-3
correcting a typing mistake, 2-1
COU command, 2-13–2-14
CPR command, 2-14
CTR command, 2-15
CTRL (control) characters. See Control characters

D

D command, 2-15
dash (-), 2-3
Data Acquisition Probe, 2-21
DATA command, 2-15
data transfers. See Intersystem Communication
DEAL command, 2-16
deleting text being entered, 2-1

delimiters, 2-3
devices. See I/O devices
DI command, 2-16
DISP command, 2-16
displaying:
    contents of Acquisition Memory, 2-16
    contents of memory, 2-15
    contents of registers, 2-16
    PROM Programmer status, 2-21
    system information, 2-24
    TTA status, 2-27
dollar sign ($), 2-4
don't-care values, 2-5
double quote ("), 2-4
download, 4-2
DS command, 2-16

E

EEPROM, 2-23, 6-2
EM command, 2-16
emulation, 6-3
    commands, 2-2
emulation modes, 1-2, 6-3
emulator, 6-3
    reading from emulator port, 2-22
    writing to emulator port, 2-28
end of file, 2-1
escape character (CTRL-C), 2-1
EVE command, 2-17
event:
    clearing parameters, 2-17
    consecutive, 2-12
    defining, 2-17
EX command, 2-17
executing a program, 2-28
expressions, 2-5
    calculating the value of, 2-10
Extended Tekhex. See Tekhex

F

F command, 2-18
formatted transfers. See Intersystem communication

G

G command, 2-18
general purpose counter, 2-13, 6-4

H

host computer. See Intersystem communication

I

intersystem communication:
    COM command parameters, 2-11
        establishing communication, 4-2
        exiting from COM, 4-3
        formatted transfers:
            Tekhex download, 4-2
            Tekhex upload, 4-3
I/O buffer (for an SVC), 3-1–3-3
I/O channels, 6-2
    assigning, 2-8, 3-4
    closing, 2-10, 3-4
    displaying assignments, 2-24
I/O devices:
    identification and type codes, 3-5
    jack numbers, 1-2
I/O redirection, 2-4

J

jack numbers, 1-2

K

keys, special, 2-1

L

line printer, 1-1, 1-2
LO command, 2-18
load module, 6-4
    A Series and B Series linker output:
        copying to a file from memory, 2-23
        loading into memory, 2-28
    hexadecimal:
        copying to a file from memory, 2-27
        loading into memory, 2-22

LOCAL mode, 2-12
LOG command, 2-18
lowercase. See Capitalization
LPT, 1-2

M

M parameter (of COM and CONFIG commands), 2-11, 2-12
M parameter (of MEMSP command), 2-20
MAC. See Memory Allocation Controller
MAP command, 2-19
MEM command, 2-19
memory:
    allocation, 2-8
    deallocation, 2-16
    filling, 2-18
    layout for SVCs, 3-2, 3-3
    memory management commands, 2-2
    memory map, 2-19
    memory spaces, 2-19
    program/prototype memory, 6-5
        changing contents, 2-17, 2-20
        copying data between program and prototype memory, 2-20
        displaying contents, 2-15
        writing contents to a file, 2-23
    prototype memory, 6-6
    read-only memory (ROM):
        detecting a write to ROM:
            MAP command, 2-19
            NOMEM command, 2-20
        patching operating system, 2-23
Memory Allocation Controller, 6-5
MEMSP command, 2-19
mode. See Emulation modes
MOV command, 2-20
multiple command lines, 2-3

N

NOMEM command, 2-20
notation conventions, 2-5–2-6
null character, 4-2, 4-3
numbers, syntax of, 2-5

O

object code:
    disassembling, 2-16
    downloading, 2-22
operators in numeric expressions, 2-5

P

P command, 2-20
paper tape reader/punch, 1-1, 1-2
parameters, 2-3
periods (...) in syntax blocks, 2-6
peripherals. See I/O devices
PPTP, 1-2, 3-5
PPTR, 1-2, 3-5
PRO command, 2-21
program (user's):
    aborting, 2-17, 3-5
    continuing execution of, 2-10
    executing, 2-18
    halting execution, 2-9
    loading, 2-18
    modifying, 2-17, 2-20
    monitoring, 2-26
    suspending, 3-5
program clock, 2-10, 3-4
program memory. See Memory
PROM Programmer, 6-6
    characteristic module, 6-2
    commands, 2-3
prototype control probe, 6-6
prototype memory. See Memory
PSTAT command, 2-21
PTYPE command, 2-22

Q

QUERY command, 2-22
quote, double ("), 2-4
quote, single ('), 2-4

R

radix selector letters, 2-5
RD command, 2-22
read-only memory (ROM). See Memory

@

redirecting I/O, 2-4
registers:
    changing contents of, 2-23
    displaying contents of, 2-16
REMI, 1-2, 3-5
REMO, 1-2, 3-5
REMS command, 2-22
RESET command, 2-22
resuming display, 2-1
RH command, 2-22
ROMPATCH command, 2-23
RPR command, 2-23
RUBOUT key, 2-1

S

S command, 2-23
S parameter (of MEMSP command), 2-20
SAV command, 2-23
SEA command, 2-24
SEL command, 2-24
semicolon (;), 2-3
service calls. See SVCs
service request block. See SRB
single quote ('), 2-4
special characters. See Characters, special
special keys. See Keys, special
SRB, 3-1, 3-2
SRB vector, 3-1
standard input, 2-4
standard output, 2-4
STARTUP string, 1-2
STAT command, 2-24
status of SVC, 3-2
STR command, 2-24
string (of characters), 2-4, 2-24
    permanent storage of, 2-21
SUSP command, 2-25
suspendable (abortable) commands, 2-7
suspending display, 2-1
SVC command, 2-25
SVCs, section 3
    in modes 1 and 2, 1-2
    I/O instruction, 3-1
    LAS memory layout, 3-3
    SAS memory layout, 3-2
SYMB command, 2-25

symbolic debug, 2-25
symbols, 2-25
    adding, 2-8
    assigning value to, 2-23
    loading, 2-26
    removing, 2-22
SYMD command, 2-25
SYMLO command, 2-26
syntax notation conventions, 2-5–2-6
system memory, 6-7
system terminal, 1-1, 6-7

T

target processor:
    selecting, 2-24
TCLR command, 2-26
Tekhex (Tektronix Hexadecimal Format), 6-8
TERM mode, 2-4, 2-12, 4-1
TNIX, 6-8
TRA command, 2-26–2-27
Trigger Trace Analyzer. See TTA
TS command, 2-27
TTA:
    Acquisition Memory, 6-1
    clearing, 2-26
    commands, 2-3
    Data Acquisition Probe, 2-21
    displaying status of, 2-27
    event, 2-17
        consecutive, 2-12

U

underlined characters, 2-6
UNIX. See TNIX
upload, 4-2
uppercase. See Capitalization

W

WH command, 2-27
WPR command, 2-27
WRT command, 2-28

X

X command, 2-28