EW

TopoSoft Detailed Design Specification Change List

The following is a brief list of the main changes made to the TopoSoft detailed design specification. These changes are reflected in the attached 0.904 version dated August 24, 1983.

1. All references to Apple II changed to Apple II+ and IIe.

2. MVP Forth editor, assembler and utilities will be contained in screen 72 to 120. (was 72 to 113)

3. Topo control words are in screen 121 to 139. (was 114 to 139)

4. "Androbot" is inserted into MVP Forth words COLD and TITLE.

5. MVP Forth word ABORT will be redefined to stop all Topos, reset the data link and check for serial card.

6. Private channels will be numbered 0 to 15 (was 1 to 16), default is channel 0.

7. Public channels will be P1-P4 instead of A-D.

8. TJOYSTICK automatically allocates public channel P2.

9. Changes in TopoSoft Commands:

   Added

   CLEAR-PUBLIC

Was	Has been changed to
SET-HEADSTOP	ENABLE-HEADSTOP DISABLE-HEADSTOP

10. Changes in TopoSoft Motion Commands:

    Added

    FWD BACK LEFT RIGHT

    Deleted

    TWAIT & TCANCEL

Was	Has been changed to
TSPEED	TGO
JOYSTICK	TJOYSTICK

TopoSoft for the Apple][+ and Apple][e
Detailed Design 0.904
August 24, 1983

Mike Saari
Ed Wischmeyer

0. Disclaimer

This design specification is subject to changes and/or additions. It is not intended for general distribution.

I. Introduction

This detailed design specifies the user commands and some of the useful utilities available under TopoSoft for the Apple][+ and][e. Because this software suite is a set of disjointed utilities added on to a pre-existing package, this detailed design is rather light on describing software structure and hierarchy...there is very little structure to implement. This document is based on the Preliminary TopoSoft Specifications (memo from EW to Peter Doctorow, dated 83July26, approved intact 83Aug1.)

II. Detailed Design

1. Media

TopoSoft will be a single, 5.25" single sided single density diskette, write protected bootable disk for the Apple II+ and IIe. The boot image will occupy the low tracks. Track 17, sector 0, will be left intact for non-Toposoft single disk copy routines. Tracks 18 and above (Forth screens 72-139) will contain source code only, specifically the MVP Forth editor, assembler, and utilities (screens 72-120) and the Topo control words (screens 121-139). These assignments may change as needed to fit on one diskette. (Note: a screen is 1024 bytes. The Apple disk has 35 tracks comprising 140 screens, or four screens per track.)

2. Language

TopoSoft will be based upon MVP Forth for the Apple][+ and][e. MVP Forth will be reassembled with one high res graphics page. All of the utilities included with MVP Forth V1.0103.03 will be included, with the following modifications to Forth words:

COLD	the MVP Forth logo will have "Androbot" inserted.
.S	utility, screen 3, will change the stack printing default to print the stack in normal order (top of stack on the right).
CALL	screen 11, will be rewritten so that it can load without the Forth assembler being memory resident.
TITLE	screen 9, will have "Androbot" inserted. The new text

11. Changes in TopoSoft Speech and Tone commands:

TONE command may or may not be implemented for first release.

Added

SET-MONOTONE

Was	Has been changed to
LOAD	SAY-LATER"
SET-VOICE-SPEED	SET-SPEECH-RATE

12. Changes in TopoSoft Status Fetch commands:

GET-SWITCH — now allows for up to 8 switches - 4 head switches and 4 unallocated switches.

Added

SPEECH-FULL? GET-QUEUE-LEFT TOPO-ON?

Was	Has been changed to
GET-QUEUE	GET-QUEUE-DEPTH

13. Changes in TopoSoft Utilities commands:

Added

MSG" GET-STATUS IS-QUEUE-FULL? T. TC.

Was	Has been changed to
BUTTON?	BUTTONS?
C-R	SAY-IT

will have to be checked for conformance with MVP's requirements for copyright.

ABORT will be redefined to check for a proper serial card, reset the data link (RESET-LINK), and stop all Topos.

The documentation for the assembler and editor is public domain and copyable. Elementary documentation is required for the disk utilities, such as copy and initialize. Many of the utilities will require little or no documentation, as the utilities are standard and covered in standard texts.

3. Public/Private Communication Channels

There are 16 "private" channels for addressing up to 16 unique Topos, numbered 0-15. On bootup, the software will be configured to communicate on channel 0. The number of a Topo is determined by hardware on the robot, such as 4 dip switches. We specifically disallow two active robots in the same room with the same number. Since every IR packet contains a channel number, any message addressed to a specific Topo will not be heard by any other Topo. Every private packet properly received by a Topo is acknowledged by an "ACK" being sent back to the host datalink.

There are also 4 "public" channels P1-P4 which allow sending commands simultaneously to multiple Topos. Channel P1 is always listened to by all Topos. The other channels are inactive on power-up. Topos can be commanded to listen on any or all of public channels P2-P4. The disadvantage of public channels is that no acknowledges are ever sent back for public messages, because of potential IR collisions by various Topos. Thus there is no assurrance that the message was received. The TJOYSTICK command orders all active Topos to listen to joystick commands on Channel P2, and then clears P2 on exit from TJOYSTICK.

The command OPEN-CHANNEL controls the channel number for all subsequent commands, until changed. All other Topo commands are assumed to only apply to the addressed Topo, or equally to multiple Topos if a public channel is used. The only exceptions are Topo status fetch commands which will not work on public channels, since Topos are not allowed to make any response to a public message.

4. Command Buffering

Topo has a command queue to hold several additional motion commands while one is executing. Buffered commands are TMOVE and TGO. Means are provided to determine the current queue depth used and remaining, cancel the current command, or flush the queue. If the Apple wishes to send another command to a full queue, it must simply wait until there is room in the queue. There are no provisions currently for additional automatic queueing on the Apple.

5. TopoSoft Commands

Every listed TopoSoft command includes a brief explanation of what is needed to implement that command. The phrase "direct command" means that the command being described has a single exact counterpart command which is sent over the IR link to Topo, so implementation consists of simply sending the counterpart command over the IR system. Similarly, "direct fetch" means a single status fetch command to Topo. More complex commands sketch out the building blocks needed.

TopoSoft commands in general do not check for valid parameters on the stack. If the user needs error protection, the user can easily incorporate this into his own application. Error conditions for erroneous parameters are unspecified.

a. TopoSoft System Words

RESET-TOPO (;) - Reset Topo to default settings, the same as a power-on reset. Includes the functions of RESET-MOTION and of RESET-SPEECH, also resets public channel settings, headstop, and headswitch readings.
Implementation: Direct command.

RESET-LINK (;) - Reset the host datalink. Any message in process is aborted.
Implementation: Direct command.

OPEN-CHANNEL (;n = -4 - +15) - Send all subsequent transmissions on the channel being opened. Channels 0 - 15 are private channels, used to control individual Topos. Channels -1 to -4 are the public channels, P1 - P4. P1 is the all-listen public channel that all Topos listen to, and P2 is the public channel which is used by the TJOYSTICK command. P1 - P4 are defined internally as constants.
Implementation: Direct datalink command.

SET-PUBLIC (;n=P2-P4) - Enable Topo to listen to the indicated public channel.
Implementation: Direct command.

CLEAR-PUBLIC (;n=P2-P4) - Disable Topos from listening to the indicated public channel. Power-on value is P2-P4 clear.
Implementation: Direct command.

ENABLE-HEADSTOP (;) - Enable Topo to stop automatically after any head switch is pressed. Power-on default condition is true. (Note: ENABLE- and DISABLE- words take no parameters, but SET- and CLEAR- words do take parameters.)
Implementation: Direct command.

DISABLE-HEADSTOP (;) - Topo will not stop upon headswitch actuation.
Implementation: Direct command.

b. TopoSoft Motion Words

For all motion-related commands, positive linear values are for forward motion, and positive angular values are for counter-clockwise turns. Negative values mean the opposite.

TMOVE (;distance,angle) - Move Topo over the given distance (in cm) and angle (in degrees). Both values are 16 bit signed integers.
Implementation: Repeat IS-QUEUE-FULL?
until room available
Direct command

FWD (;distance) - Move Topo straight forward over the given distance (in cm). Uses a 16 bit signed integer. A negative value causes a backward motion.
Implementation: distance 0 TMOVE

BACK (;distance) - Move Topo straight backward over the given distance (in cm). Uses a 16 bit signed integer. A negative value causes a forward motion.
Implementation: distance (negate) 0 TMOVE

LEFT (;angle) - Turn Topo in place over the given angle (in degrees). Uses a 16 bit signed integer. A negative value causes a right turn.
Implementation: 0 angle TMOVE

RIGHT (;angle) - Turn Topo in place over the given angle (in degrees). Uses a 16 bit signed integer. A negative value causes a left turn.
Implementation: 0 angle (negate) TMOVE

SET-V (;speed) - Set the centerline target speed (in cm/sec) which Topo will attempt to maintain during a TMOVE. Power-on default is ( ). For motion commands with only rotation, the units are in degrees/sec.
Implementation: Direct command.

SET-RAMP (;n=0-255) - Set the acceleration/deceleration ramp characteristics (in cm/sec/sec) during a TMOVE. Power-on default is ( ).
Implementation: Direct command.

TGO (;velocity,turnrate) - Cause Topo to start moving with the indicated linear velocity (in cm/sec) and angular velocity (in deg/sec). Motion will continue until another command is received, or until carrier loss is detected onboard Topo.
Implementation: Repeat IS-QUEUE-FULL?
until room available
Direct command

TJOYSTICK (;) - Starts with 3 warning ticks and a beep, then initiates real-time analog control of Topo. Exit by pressing either joystick button, or any key on the keyboard (except shift or control). When a key is pressed, the keystroke is used to

start the next command. Automatically uses public channel P2 for better real-time response.
Implementation: P2 SET-PUBLIC (current Topo listen to P2)
Save current channel in memory
P2 OPEN-CHANNEL
Sound warning beeps
Repeat
GET-JOYSTICK, convert numeric data, TSPEED
Until keystroke or button
Then 0 0 TGO (stop Topo)
Using the saved "current channel," OPEN CHANNEL
P2 CLEAR-PUBLIC (current Topo don't listen P2)

TPARK (;) - Cancel the current motion command and discard any remaining motion commands in the queue.
Implementation: Direct command.

RESET-MOTION (;) - TPARK, and set all motion parameters to their power-on defaults.
Implementation: Direct command.

c. TopoSoft Speech and Tone Commands

SAY" (;) - Command Topo's speech module to say the following text, delimited by double quotes. Missing trailing quotes are handled as in MVP Forth.
Implementation: speech-code# START-MSG
(.") C-R
END-MSG

PHON" (;) - Command Topo's speech module to say the following phoneme codes, delimited by quotes. Missing trailing quotes are handled as in MVP Forth.
Implementation: speech-code# START-MSG
CTRL-V (.") C-R
END-MSG

SAY-LATER" (;) - Load the following text (without saying it yet) to Topo's speech module. Text is delimited by quotes. This speech will be initiated by SAY", PHON", SET-PITCH, SET-MONOTONE, SET-SPEECH-RATE, or SET-VOLUME.
Implementation: speech-code# START-MSG
(:")
END-MSG

SET-PITCH (;n=1-63,) - Set speaking pitch for the given value if n. Every increase or decrease of 20 is about one octave. Power on reset value is 24.
Implementation: Save pitch=n
speech-code# START-MSG
CTRL-E

```
                Output "value of n"
                If flatflag=true, then output "F" else output "P"
                END-MSG
```

SET-MONOTONE (;on/off) - Set Echo speech to be monotonic if the flag is true, otherwise set to inflected speech. Power on value is inflected.

```
Implementation: Save flatflag=flat? value
                speech-code# START-MSG
                CTRL-E
                Output "pitch value"
                If flat?, then output "F" else output "P".
                END-MSG
```

SET-SPEECH-RATE (;fast?) - Set speaking rate to fast or slow. The power on reset value is slow.

```
Implementation: speech-code# START-MSG
                CTRL-E
                If fast, then output "C" else output "E"
                END-MSG
```

SET-VOLUME (;n=0-15) - Command Topo's speech module volume to the indicated value. Power on reset value is 12.

```
Implementation: speech-code# START-MSG
                CTRL-E
                Output "value of n", "V"
                END-MSG
```

SET-SPEECH (;) - Abort current speech and flush the speech ffer on Topo's speech module. Sets speech-rate to slow, volume to 12, and pitch to 24 inflected.

```
Implemented: Direct command.
```

TONE (;freq,duration) - Enable Topo's tone generator to sound for the specified pitch and duration. Tone commands do not queue up, so any new tone command will override an ongoing command. This may or may not be implemented in the initial release.

```
Implementation: Direct command.
```

d. TopoSoft Status Fetch Commands

Note: All status commands will be delayed by the IR communications system by about 100 ms, so the status commands will not necessarily be totally accurate.

SPEECH-FULL? (;-) flag) Returns true if Topo's speech buffer is full.

MOVING? (;->flag) - Returns true if Topo is currently moving its wheels.

```
Implementation: Direct fetch and mask.
```

TALKING? (;->flag) - Returns true if Topo's speech module is eaking.

```
plementation: Direct fetch and mask.
```

TONING? (;->flag) - Returns true if Topo's tone generator is active.
mplementation: Direct fetch and mask.

RECENT-ON? (;->flag) - Returns true if Topo has been powered on since the last RECENT-ON? request.
Implementation: Get power-on flag (direct fetch and mask)
Reset power-on flag (direct command).

GET-DISTANCE (;->distance) - Read the distance covered so far, in cm, either by the currently executing motion command, if there is one, or by the last motion command executed, if there was one since the last reset, or a 0.
Implementation: Direct fetch.

GET-ANGLE (;->angle) - Read the angle covered so far, in degrees, either by the currently executing motion command, if there is one, or by the last motion command executed, if there was one since the last reset, or a 0.
Implementation: Direct fetch.

GET-VELOCITY (;->velocity) - Read Topo's current instantaneous velocity controller target speed, in cm/sec. This speed is the desired speed that the controller is trying to move Topo, and is time varying due to steady state commanded speeds and ramping. It is equivalent to the actual current motor speed, except for any overshoot, undershoot, or time delays inherent to actually riving the motors.
mplementation: Direct fetch and mask.

GET-TURNRATE (;->turnrate) - Read Topo's current velocity controller target turning rate, in degrees/sec. This is analogous to GET-VELOCITY.
Implementation: Direct fetch and mask.

GET-SWITCH (;->n=0-255) - Returns a 8-bit code for Topo's 4 head switches and 4 other unimplemented switches. 1=pressed. Bit 0=switch#0. Switches 0-3 are latched, but the other switches, as yet unimplemented, may not be latched. The latched switches are reset when a GET-SWITCH is executed, or when the Topo is reset.
Implementation: Latch head switch (direct command), then
Direct fetch and mask.

GET-QUEUE-DEPTH (;->n) - Returns the count of motion commands which have not completed execution, including the currently executing command.
Implementation: Direct fetch and mask.

GET-QUEUE-LEFT (;-> n) - Returns the number of empty positions left in the motion queue.
Implementation: Direct fetch and mask.

ET-VERSION# (;->n = 1-255) - Returns Topo's onboard version

number.
Implementation: Direct fetch and mask.

TOPO-ON? (;->flag) - Return a true if there is a Topo answering on the currently active channel.
Implementation: request status (direct command)
repeat get-linkstatus
until error or message waiting
error returns false, msg waiting returns true

e. TopoSoft Utilities

These are a few underlying words used by the above commands.

GET-JOYSTICK (;->x,y) - Reads the joystick and returns x and y values, each 0-255. X=0 is extreme left, Y=0 is extreme top.

BUTTONS? (;->bits) - Returns true if either joystick button is depressed. The bits can be decoded to specify which button was pushed. Bit#0=button 0, Bit#1=button 1.

START-MSG (;cmd#) - This word revectors the Forth word EMIT to use TEMIT, which sends character output to the host datalink via a serial card in slot 2 of the Apple, and also handles the handshaking between the Apple and the datalink. Then it sends a "start-message" code to the data link, followed by the command number. All further text output by the Apple goes to the host datalink until an END-MSG is encountered.
Implementation: Standard Apple/datalink message header.

END-MSG (;) - Send the datalink an "end-message" code, and then re-vector EMIT to the value it had before the START-MSG.
Implementation: Standard Apple/datalink message terminator.

MSG" (;) - Send the following character string, delimited by quotes, to the current Topo.
Implementation: START-MSG
(.")
END-MSG

GET-STATUS (;cmd# -> char3, char2, char1, char0) - Send the given status command number to Topo, and return the 4 character response from Topo. For returning a 32-bit value, each character is a hex digit. Char0 is the least significant.
Implementation: START-MSG
END-MSG
Repeat GET-LINKSTATUS and test
until message received
Read message and put value on stack

GET-LINKSTATUS (;->byte) - Returns a byte giving the status of the datalink and of the last message sent.
Implementation: Standard Apple/datalink "Query" command.

SAY-IT (;) - Output the ASCII code for carriage return (no linefeed). Used to activate Topo's speech module. SAY-IT must be enclosed in a message.
Implementation: cr-code EMIT.

T. (;value) - Take a value off the stack, convert it to ASCII characters in the current base, and emit these ASCII characters without leading or trailing blanks. This utility is used to ship a 16 bit number to Topo.

TC. (;byte) - Like T. , but TC ships out a only byte.

IS-QUEUE-FULL? (;-> flag) This returns a flag to indicate if the motion queue is full or has space available for another motion command. This word maintains a counter so that Topo is not queried for status every request, but only when the number of empty spaces determined at the last status request have been used up in subsequent motion commands.
Implementation: Get count of known empty spaces and decrement
If none, poll Topo with GET-STATUS-LEFT
Save new value
Return false if value >0

6. TopoTeach Compatibility

TopoSoft and TopoTeach share the same basic syntax structure, so source code compatibility is accomplished easily through straightforward translation utilities. These utilities should only be released concurrently with TopoTeach release or later, since TopoTeach is still subject to minor change.

7. Demos

There will be at least a simple functionality test. Other demos may be included if marketing writes any.

8. Required Activities outside of Engineering

1) Inform MVP and the authors (Haydon & Kunze) what we are up to, and double check that proper acknowledgements are given.
2) Arrange appropriate compensation.
3) Documentation!! Simple reference information, and some simple examples of TopoSoft words, plus documentation on the assembler and the editor.