ST-2900 RAM-512 BOARD

Hardware and Software
User's Manual

(Preliminary Version)

Sardis Technologies
2261 East 11th Avenue
Vancouver, B.C.
Canada V5N 1Z7

- -

ABBREVIATIONS

68MJ - '68' Micro Journal magazine

TRADEMARKS

COPYRIGHT INFORMATION

DISCLAIMER

This manual last revised December 10, 1985

0.0 Table of Contents

1.0 Introduction

The ST-2900 RAM-512 board from Sardis Technologies greatly improves the overall performance of the ST-2900 system. The software provided allows you to use the 1/2 megabyte of memory as a high speed virtual disk (RAM-Disk) for the FLEX, STAR-DOS, or OS-9 operating systems.

The huge storage capacity -- 40% more than a double-sided, double-density 40 track 5" disk -- handles large sorts and compiles with room to spare. Compile times for Microware's OS-9 C compiler are dramatically slashed. Under FLEX, the Screditor III text editor loads in under 2 seconds!

Fully transparent refresh lets the CPU access the memory at full speed (no wait states), while at the same time protecting the integrity of your data by refreshing the memory well within manufacturer's specifications. The low power consumption minimizes heat buildup.

Unlike some other RAM-Disk software, ours stores a checksum for each "sector" of data and verifies that checksum every time a sector is read. Although the data stored in the RAM-Disk will rarely be corrupted, if it ever happens (caused by a wayward program, a faulty chip, or a glitch in the power supply), our software will tell you it happened, instead of crashing the system.

2.0 HARDWARE SECTION

2.1 Getting Started -- Fully Assembled and Tested RAM-512 Board (without RAM)

[] NOTE -- Static electricity can damage MOS integrated circuits. Even a static electricity charge too weak to feel can cause problems, and the damage, if it merely weakened the chip, might not cause the chip to fail until weeks or months later. While handling any MOS IC's or circuit boards containing MOS IC's you should ground your body and all tools and work areas that will touch the IC leads or the circuit board. Use a 1 Megohm resistor in series between you and ground to protect yourself against dangerous shocks. Remember -- just because the IC's have been installed in their sockets on the board doesn't make them immune to static electricity. Better safe than sorry!

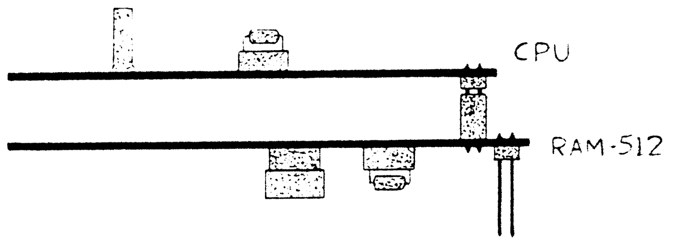
[] Before you proceed with the RAM-512 board, you should have the CPU board up and running by itself.

[] Remove ALL the protective anti-static foam pieces (they are black in colour) still stuck onto the top and bottom of the RAM-512 board. Make sure no small particles remain on the board as they are conductive and could cause the board to malfunction.

[] Put shorting blocks on the jumpers. See section 2.5 for a description of all jumpers.

[] Insert your 16 DRAM chips into their sockets (refer to section 2.9)

[] Make sure the power to the CPU board is turned off, wait several seconds until the capacitors are sufficiently discharged, then plug the RAM-512 board into the CPU board. Do not connect the FDC board at this time. Since there are 60 pins in the connectors, considerable pressure will be required, but be careful to apply the pressure evenly so as not to break anything.



[] Power the system up and verify that the CPU board still works after the addition of the RAM-512 board.

[] Use ST-MON's "T" command to test the memory at $E000-$EFFF. Use the "M" command to change location $FFB8 to $C5, then re-run the "T E000 EFFF" command. If any errors are displayed, refer to the trouble-shooting hints (section 2.3).

[] If you will also be using the ST-2900 FDC board, turn the power off. Now follow the instructions in the "Getting Started -- Fully Assembled and Tested FDC Board" section in the addendum to the FDC User Manual, except that you will be plugging the FDC board into the RAM-512 board instead of into the CPU board:

[picture #2]
(see below)

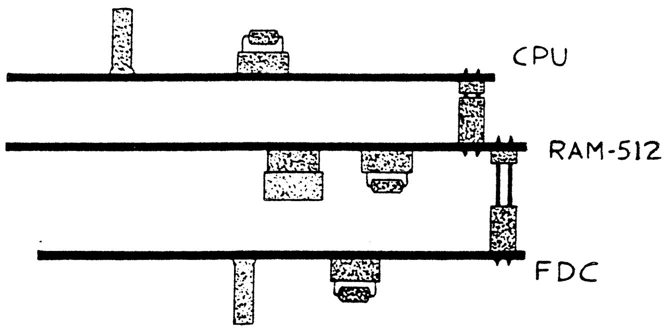
[] Run the supplied MEMTEST utility to check out the RAM-512 board. If you have the FDC board and a disk drive connected, insert the supplied "ST-2900 RAM-512 MEMTEST" diskette, then tell ST-MON to run it by typing "D F". Don't be alarmed if after some disk action nothing seems to happen -- approx. once every 52 seconds an asterisk will be printed to the screen to let you know another pass has been completed. If you don't have the FDC board, you will have to type in the object code of the MEMTEST program as shown in section 3.8. For more details, refer to section 3.7 (even if you only have the OS-9 software).

2.2 Theory of Operation

(not yet completed)

2.3 Trouble-shooting Hints

(not yet completed)



2.4 Parts List -- RAM-512 Board

	U1	74F02 quad dual input NOR
	U2	74F04 hex inverter
	U3	74LS10 triple three input NAND
	U4	74LS11 triple three input AND
	U5	74F20 dual four input NAND
	U6	74LS30 8 input NAND
	U7,U8	74F32 quad dual input OR
	U9,U10	74HCT244 octal bus buffer (3 state)
	U11	74HC4040 12 bit binary ripple counter
	U12	PAL16R8A-2 octal 16 input registered And-Or PAL
	U13	LPSLDM-200 200 nsec. low power Schottky logic delay module
	U14	74HCT245 octal bus transceiver (3 state)
	U15	74LS373 octal transparent latch (3 state)
	U16-U18	PAL16L8A-4 octal 16 input And-Or-Invert PAL
t	U19-U34	41256 256Kx1 dynamic RAM (150 nsec., tCAC <= 75 nsec.)
	U35	7407 hex buffer - open collector
*	C1-C3	10 uF tantalum capacitors 6.3 vdc
	C~~2~~4-C~~36~~37	.22 uF or .33 uF bypass capacitors (monolithic axial lead preferred)
	R1	4.7K ohm 1/4 watt resistor
	R2-R4	33 ohm " " "
	R5-R16	10 ohm " " "
	R17,R18	33 ohm " " "
	R19-R24	10 ohm " " "
	R25	470 ohm " " "
	R26	~~1.5K~~ 470 ohm " " "
*		10 - 14 pin IC sockets
*		17 - 16 pin IC sockets
*		8 - 20 pin IC sockets
t	P1	60 pin (2 row x 30) .100" center male header (wire-wrap)
	S1	60 pin (2 row x 30) .100" center female header (solder-tail)
	J1	2 pin .100" center male header
	J2-J4	3 pin .100" center male header

* these components must be oriented as marked on PCB
t see section 2.6 for optional configurations

or replace with a single female header with wire-wrap leads

2.5 Appendix A -- Jumpers

J1 - shorted: page latch address is $FFB8 - used for board #1
open: " " " " $FFBC - used for board #2

J2,J3 - short the center pin to the pin marked "64" for 64Kx1 chips (128K bytes)
- short the center pin to the pin marked "256" for 256Kx1 chips (512K bytes)

J4 - short the center pin to the pin marked "2" (the "1" setting is not currently used)

J5a/J5b - the design of this jumper pair will be simplified in the next version of the board. ~~For now, pin 2 of J5a should be connected to pin 2 of J5b, and pin 3 of J5a to pin 3 of J5b. Ignore pins 1 and 4~~.
* Refer to errata sheet

2.6 Appendix B -- Optional Configurations

1) For 128K bytes of RAM, use 16 64Kx1 dynamic RAMs (both 128 and 256 cycle refresh will work); for 512K bytes of RAM use 16 256Kx1 dynamic RAMs.

2) If you will not be plugging a third board (such as the ST-2900 FDC) into the "sandwich", connector P1 is not needed.

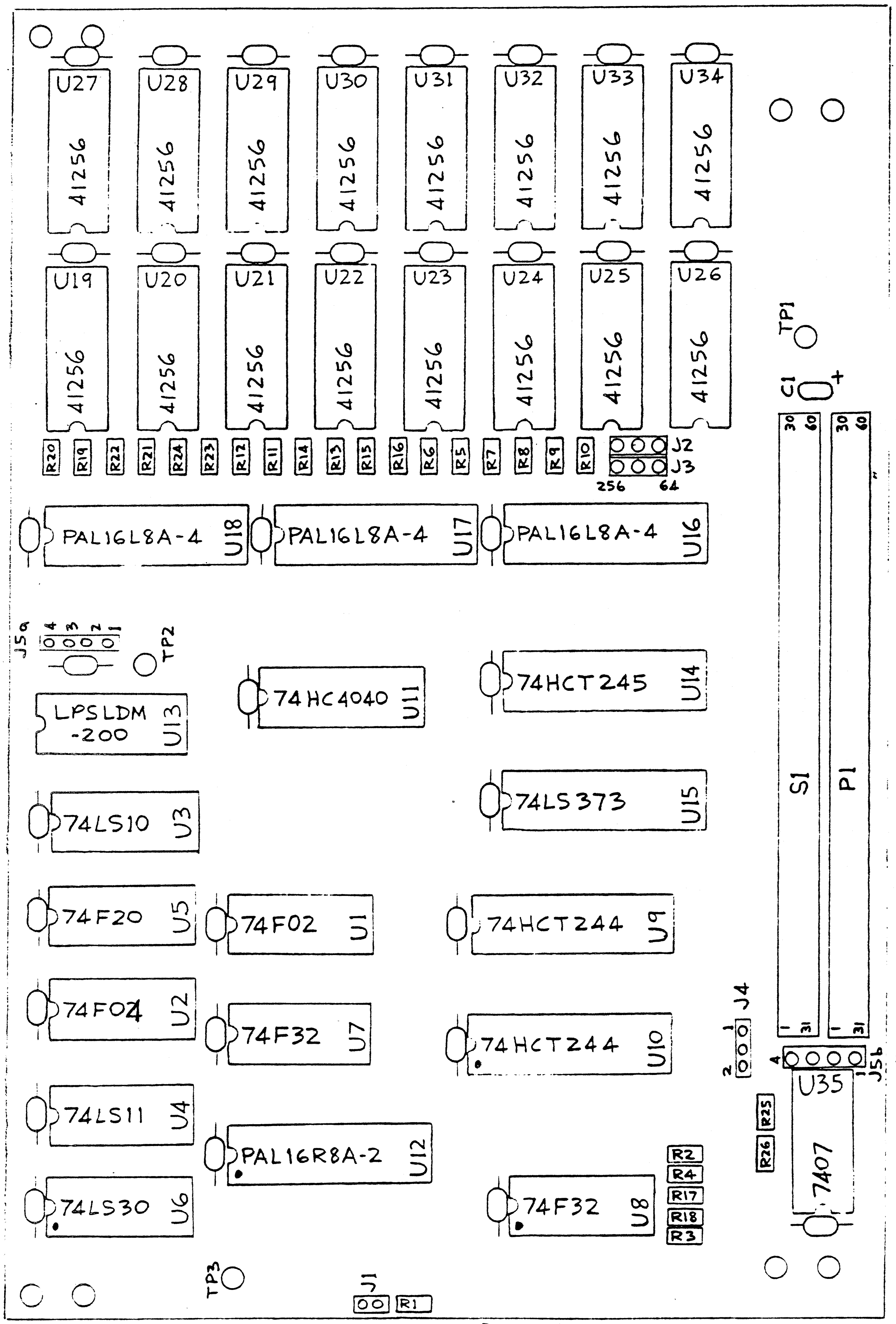
2.7 Appendix C -- Memory Map

0000-DFFF	(unchanged)
E000-EFFF	"window" where the currently selected page appears
F000-FFB7	(unchanged)
FFB8-FFBB	page latch for board #1
FFBC-FFBF	page latch for board #2
FFC0-FFFF	(unchanged)

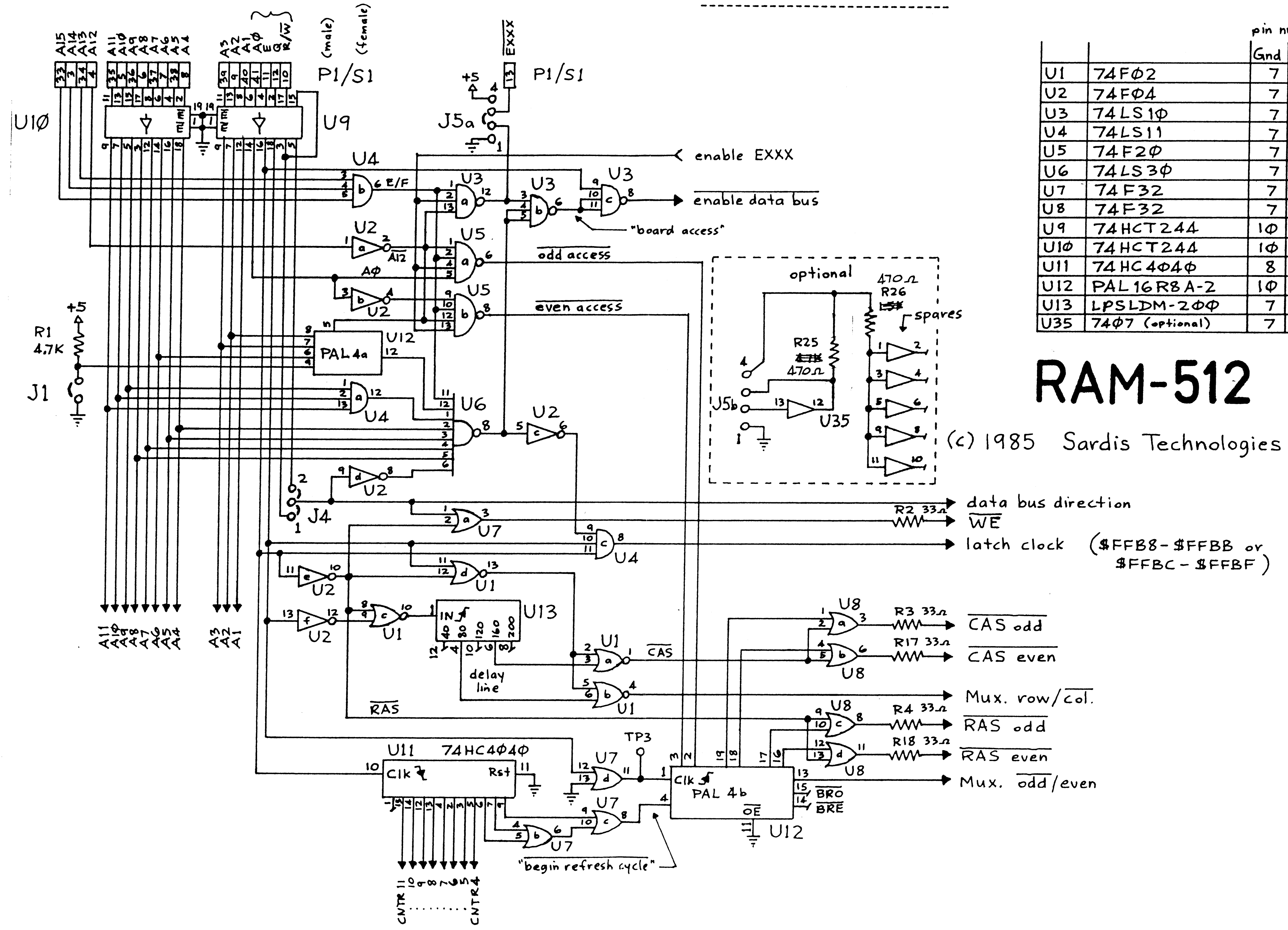
2.8 Appendix D -- Suggested Sources of Further Information

(not yet completed)

RAM-512 Component Layout



Revision A (c) 1985 Sardis Technologies



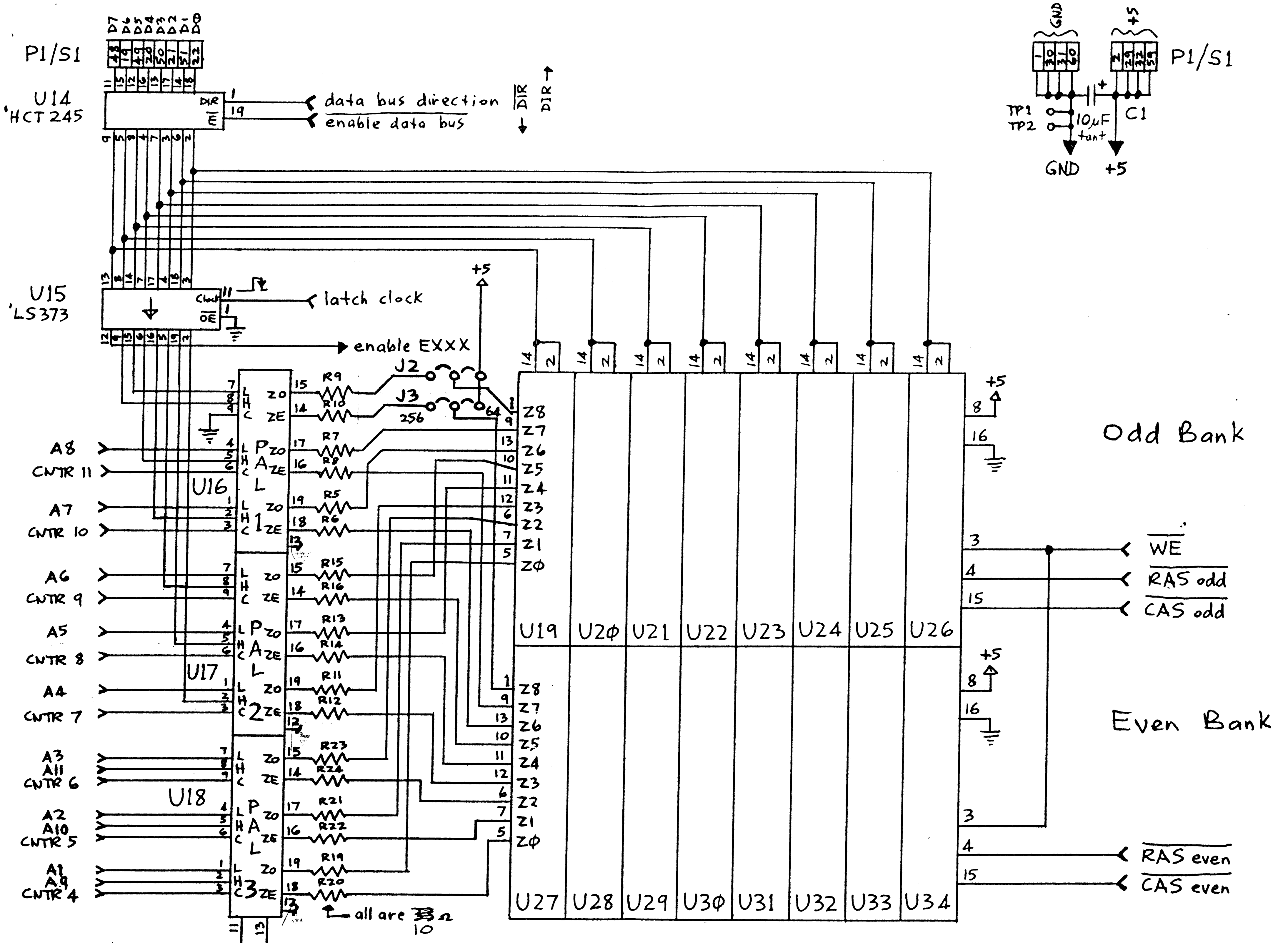
		pin numbers	
		Gnd	+5
U1	74F02	7	14
U2	74F04	7	14
U3	74LS10	7	14
U4	74LS11	7	14
U5	74F20	7	14
U6	74LS30	7	14
U7	74F32	7	14
U8	74F32	7	14
U9	74HCT244	10	20
U10	74HCT244	10	20
U11	74HC4040	8	16
U12	PAL16R8A-2	10	20
U13	LPSLDM-200	7	14
U35	7407 (optional)	7	14

RAM-512

(c) 1985 Sardis Technologies

Sheet 1 of 2

drawn Aug. 3/85 by DCW



~~.33μF~~
.22μF ~~.1μF~~ bypass capacitors
for each IC chip.

4164 or 41256 DRAM's
150 nsec. ($t_{CAC} \leq 75$ nsec.)
(128 or 256 cycle refresh)

		pin number	
		Gnd	+5
U14	74HCT245	10	20
U15	74LS373	10	20
U16	PAL16L8A-4	10	20
U17	PAL16L8A-4	10	20
U18	PAL16L8A-4	10	20

C2-C35

RAM-512

sheet 2 of 2

© 1985 Sardis Technologies

Feb. 27/85

2.11 Appendix G -- Pinout of PAL's and Delay Line

PAL®-Programmable Array Logic HAL®-Hard Array Logic

Features/Benefits

- Reduces SSI/MSI chip count greater than 5 to 1
- Saves space with SKINNYDIP® packages
- Reduces IC inventories substantially
- Expedites and simplifies prototyping and board layout
- PALASM™ silicon compiler provides auto routing and test vectors
- Security fuse reduces possibility of copying by competitors

Description

The PAL family utilizes an advanced Schottky TTL process and the Bipolar PROM fusible link technology to provide user programmable logic for replacing conventional SSI/MSI gates and flip-flops at reduced chip count.

The HAL family utilizes standard Low-Power Schottky TTL process and automated mask pattern generation directly from logic equations to provide a semi-custom gate array for replacing conventional SSI/MSI gates and flip-flops at reduced chip count.

There are four different speed/power families offered. Choose from either the standard, high speed, half power, or quarter power family to maximize design performance.

The PAL/HAL lets the systems engineer "design his own chip" by blowing fusible links to configure AND and OR gates to perform his desired logic function. Complex interconnections which previously required time-consuming layout are thus "lifted" from PC board etch and placed on silicon where they can be easily modified during prototype check-out or production.

The PAL transfer function is the familiar sum of products. Like the PROM, the PAL has a single array of fusible links. Unlike the PROM, the PAL is a programmable AND array driving a fixed OR array (the PROM is a fixed AND array driving a programmable OR array).

The HAL transfer function is the familiar sum of products. Like the ROM, the HAL has a single array of selectable gates. Unlike the ROM, the HAL is a selectable AND array driving a fixed OR array (the ROM is a fixed AND array driving a selectable OR array).

In addition the PAL/HAL provides these options:

- Variable input/output pin ratio
- Programmable three-state outputs
- Registers with feedback
- Arithmetic capability
- Exclusive-OR gates

PAL®, (Programmable Array Logic), PALASM®, HAL®, and SKINNYDIP® are registered trademarks and PMSI, and HMSI are trademarks of Monolithic Memories Inc.

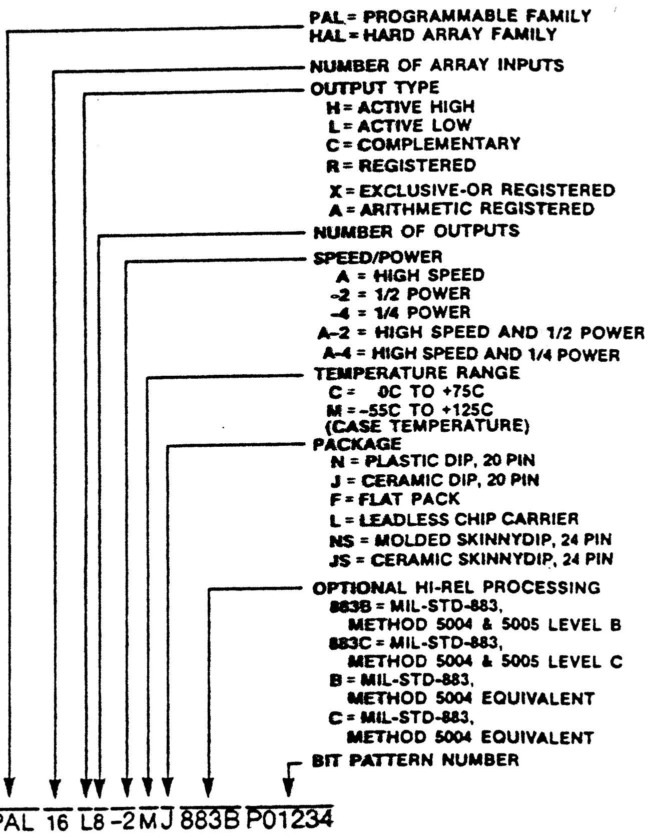
Unused inputs are tied directly to V_{CC} or GND. Product terms with all fuses blown assume the logical high state, and product terms connected to both true and complement of any single input assume the logical low state. Registers consist of D type flip-flops which are loaded on the low-to-high transition of the clock. PAL/HAL Logic Diagrams are shown with all fuses blown, enabling the designer to use the diagrams as coding sheets.

The entire PAL family is programmed using inexpensive conventional PROM programmers with appropriate personality and socket adapter cards. Once the PAL is programmed and verified, two additional fuses may be blown to defeat verification. This feature gives the user a proprietary circuit which is very difficult to copy.

To design a HAL, the user first programs and debugs a PAL using PALASM and the "PAL DESIGN SPECIFICATION" standard format. This specification is submitted to Monolithic Memories where it is computer processed and assigned a bit pattern number, e.g., P01234.

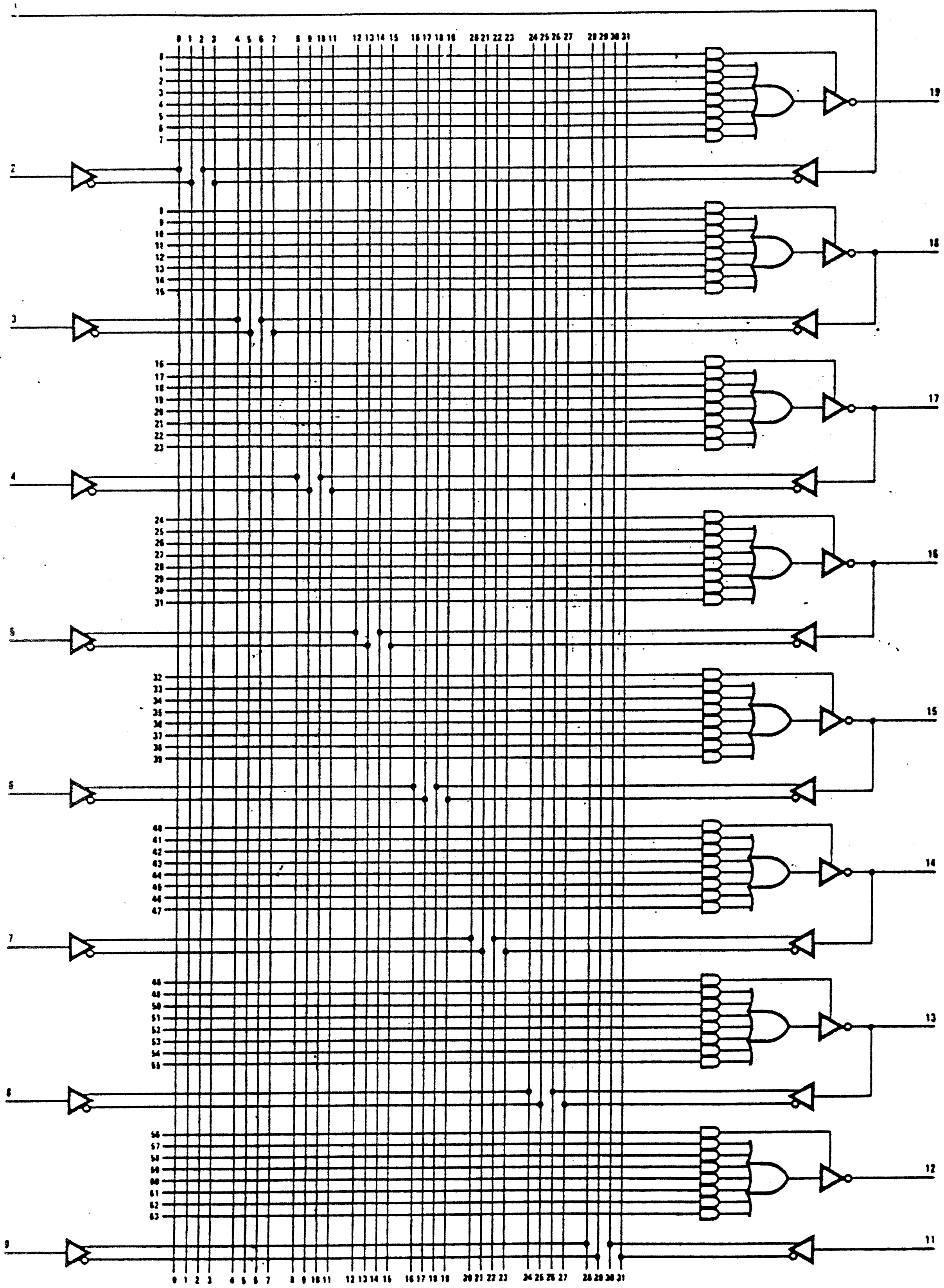
Monolithic Memories will provide a PAL sample for customer qualification. The user then submits a purchase order for a HAL of the specified bit pattern number, e.g., HAL18L4 P01234. See Ordering Information below.

Ordering Information

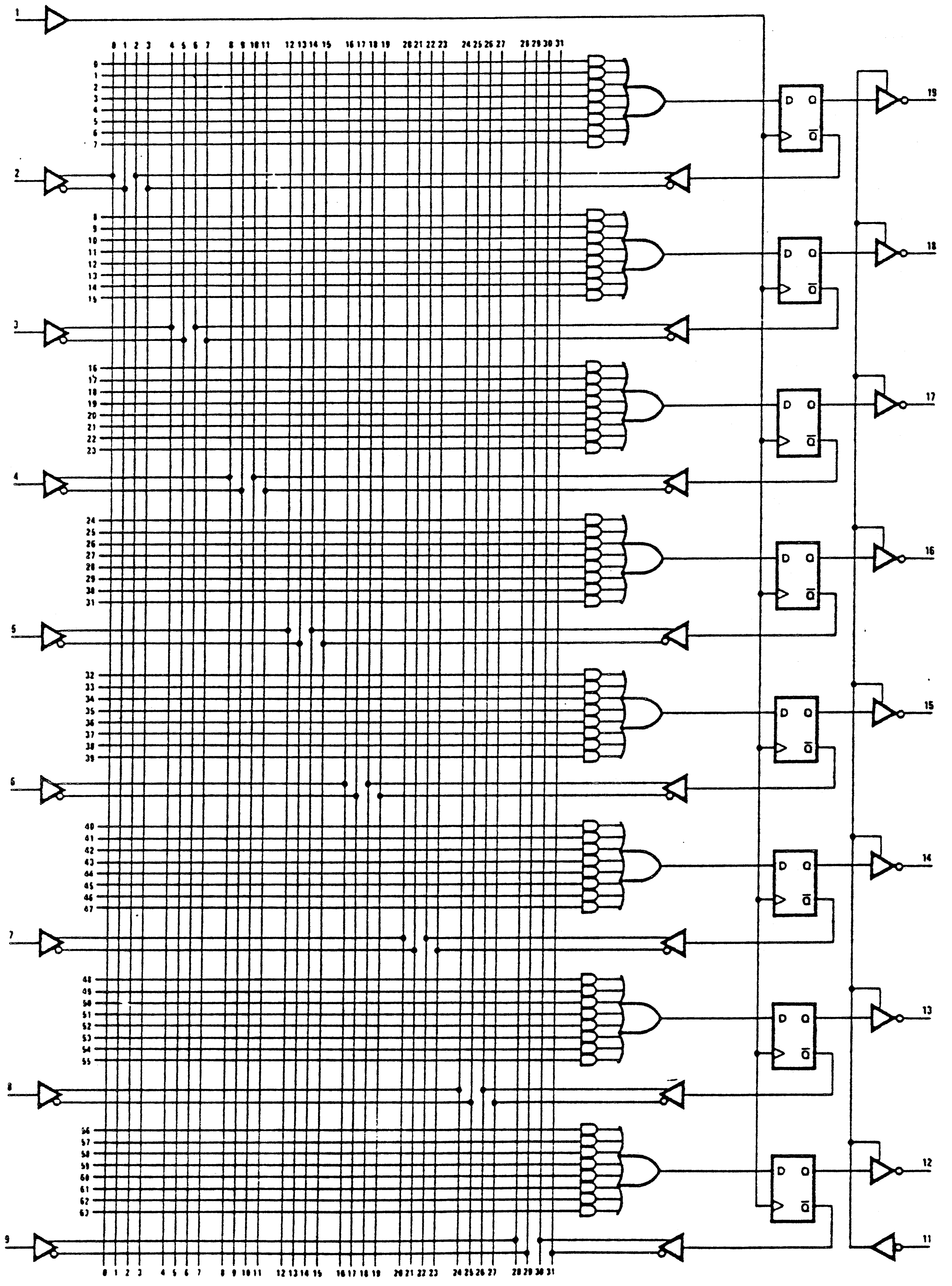

PAL = PROGRAMMABLE FAMILY
HAL = HARD ARRAY FAMILY
NUMBER OF ARRAY INPUTS
OUTPUT TYPE
H = ACTIVE HIGH
L = ACTIVE LOW
C = COMPLEMENTARY
R = REGISTERED
X = EXCLUSIVE-OR REGISTERED
A = ARITHMETIC REGISTERED
NUMBER OF OUTPUTS
SPEED/POWER
A = HIGH SPEED
-2 = 1/2 POWER
-4 = 1/4 POWER
A-2 = HIGH SPEED AND 1/2 POWER
A-4 = HIGH SPEED AND 1/4 POWER
TEMPERATURE RANGE
C = 0C TO +75C
M = -55C TO +125C
(CASE TEMPERATURE)
PACKAGE
N = PLASTIC DIP, 20 PIN
J = CERAMIC DIP, 20 PIN
F = FLAT PACK
L = LEADLESS CHIP CARRIER
NS = MOLDED SKINNYDIP, 24 PIN
JS = CERAMIC SKINNYDIP, 24 PIN
OPTIONAL HI-REL PROCESSING
883B = MIL-STD-883, METHOD 5004 & 5005 LEVEL B
883C = MIL-STD-883, METHOD 5004 & 5005 LEVEL C
B = MIL-STD-883, METHOD 5004 EQUIVALENT
C = MIL-STD-883, METHOD 5004 EQUIVALENT
BIT PATTERN NUMBER
PAL 16 L8 -2 M J 883B P01234

Monolithic Memories

16L8


1
2
3
4
5
6
7
8
9
19
18
17
16
15
14
13
12
11
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

16R8



1
2
3
4
5
6
7
8
9
11
12
13
14
15
16
17
18
19
D
Q
Q̄
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Operating Conditions

SYMBOL	PARAMETER			MILITARY MIN	MILITARY TYP	MILITARY MAX	COMMERCIAL MIN	COMMERCIAL TYP	COMMERCIAL MAX	UNI
V_{CC}	Supply voltage			4.5	5	5.5	4.75	5	5.25	V
t_w	Width of clock	16R8A-4 16R6A-4 16R4A-4	Low	40	20		30	20		ns
			High	40	20		30	20		
t_{su}	Set up time from input or feedback to clock	16R8A-4 16R6A-4 16R4A-4		90	45		60	45		ns
t_h	Hold time			0	−15		0	−15		ns
T_A	Operating free-air temperature			−55		125	0		75	°C

Electrical Characteristics Over Operating Conditions

SYMBOL	PARAMETER	TEST CONDITIONS			MIN	TYP	MAX	UNI
V_{IL}*	Low-level input voltage						0.8	V
V_{IH}*	High-level input voltage				2			V
V_{IC}	Input clamp voltage	V_{CC} = MIN		I_I = −18mA		−0.8	−1.5	V
I_{IL}	Low-level input current †	V_{CC} = MAX		V_I = 0.4V		−0.02	−0.25	mA
I_{IH}	High-level input current †	V_{CC} = MAX		V_I = 2.4V			25	μA
I_I	Maximum input current	V_{CC} = MAX		V_I = 5.5V			1	mA
V_{OL}	Low-level output voltage	V_{CC} = MIN	MIL	I_{OL} = 4mA		0.3	0.5	V
			COM	I_{OL} = 8mA				
V_{OH}	High-level output voltage	V_{CC} = MIN	MIL	I_{OH} = −1mA	2.4	2.8		V
			COM	I_{OH} = −1 mA				
I_{OZL}	Output short-circuit current**	V_{CC} = MAX		V_O = 0.4V			−100	μA
I_{OZH}				V_O = 2.4V			100	μA
I_{OS}	Output short-circuit current	V_{CC} = 5V		V_O = 0V	−30	−70	−130	mA
I_{CC}	Supply current	V_{CC} = MAX	16R4A-4 16R6A-4 16R8A-4 16L8A-4			30	50	mA

Switching Characteristics Over Operating Conditions

SYMBOL	PARAMETER		TEST	MILITARY MIN	MILITARY TYP	MILITARY MAX	COMMERCIAL MIN	COMMERCIAL TYP	COMMERCIAL MAX	UNIT
t_{PD}	Input or feed-back to output	16R6A-4 16R4A-4 16L8A-4	R_T = 800Ω R_2 = 1.56kΩ		35	75		35	55	ns
t_{CLK}	Clock to output or feedback				20	45		20	35	ns
$t_{PXZ/ZX}$	Pin 11 to output disable/enable—except 16L8A-4				15	40		15	30	ns
t_{PZX}	Input to output enable	16R6A-4 16R4A-4 16L8A-4			30	65		30	50	ns
t_{PXZ}	Input to output disable	16R6A-4 16R4A-4 16L8A-4			30	65		30	50	ns
f_{MAX}	Maximum frequency	16R8A-4 16R6A-4 16R4A-4		8	18		11	18		MHz

Operating Conditions

SYMBOL	PARAMETER		MILITARY MIN	MILITARY TYP	MILITARY MAX	COMMERCIAL MIN	COMMERCIAL TYP	COMMERCIAL MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	4.75	5	5.25	V
t_w	Width of clock	Low	25	10		25	10		ns
		High	25	10		25	10		
t_{su}	Set up time from input or feedback to clock	16R6A-2 16R4A-2 16R8A-2	50	25		35	25		ns
t_h	Hold time		0	-15		0	-15		ns
T_A	Operating free-air temperature		-55		125	0		75	°C

Electrical Characteristics Over Operating Conditions

SYMBOL	PARAMETER	TEST CONDITIONS			MIN	TYP	MAX	UNIT
V_{IL}*	Low-level input voltage						0.8	V
V_{IH}*	High-level input voltage				2			V
V_{IC}	Input clamp voltage	V_{CC} = MIN		I_I = −18mA		−0.8	−1.5	V
I_{IL}	Low-level input current †	V_{CC} = MAX		V_I = 0.4V		−0.02	−0.25	mA
I_{IH}	High-level input current †	V_{CC} = MAX		V_I = 2.4V			25	μA
I_I	Maximum input current	V_{CC} = MAX		V_I = 5.5V			1	mA
V_{OL}	Low-level output voltage	V_{CC} = MIN	MIL	I_{OL} = 12mA		0.3	0.5	V
			COM	I_{OL} = 24mA				
V_{OH}	High-level output voltage	V_{CC} = MIN	MIL	I_{OH} = -2mA	2.4	2.8		V
			COM	I_{OH} = -3.2mA				
I_{OZL}	Off-state output current †	V_{CC} = MAX		V_O = 0.4V			−100	μA
I_{OZH}				V_O = 2.4V			100	μA
I_{OS}	Output short-circuit current **	V_{CC} = 5V		V_O = 0V	−30	-70	−130	mA
I_{CC}	Supply current	V_{CC} = MAX				60	90	mA

Switching Characteristics Over Operating Conditions

SYMBOL	PARAMETER		TEST CONDITIONS	MILITARY MIN	MILITARY TYP	MILITARY MAX	COMMERCIAL MIN	COMMERCIAL TYP	COMMERCIAL MAX	UNIT
t_{PD}	Input or feed-back to output	16L8A-2 16R6A-2 16R4A-2	R_1 = 200Ω R_2 = 390Ω		25	50		25	35	ns
t_{CLK}	Clock to output or feedback				15	25		15	25	ns
$t_{PXZ/ZX}$	Pin 11 to output disable/enable except 16L8A-2				15	25		15	25	ns
t_{PZX}	Input to output enable	16L8A-2 16R6A-2 16R4A-2			25	45		25	35	ns
t_{PXZ}	Input to output disable	16R8A-2 16R6A-2 16R4A-2			25	45		25	35	ns
f_{MAX}	Maximum frequency	16R8A-2 16R6A-2 16R4A-2		14	25		16	25		MHz

PAL 1,2,3 (PAL16L8A-4)

Name	Pin No.	Type	Description
AHA	2	input	Address upper A
ALA	1	input	Address lower A
CTRA	3	input	Counter A
AHB	5	input	Address upper B
ALB	4	input	Address lower B
CTRB	6	input	Counter B
AHC	8	input	Address upper C
ALC	7	input	Address lower C
CTRC	9	input	Counter C
MXE	13	input	multiplex $\overline{\text{odd}}$/even
MXR	11	input	multiplex row/$\overline{\text{column}}$
/ZOA	19	output	$\overline{\text{multiplexed address odd A}}$
/ZEA	18	output	$\overline{\text{multiplexed address even A}}$
/ZOB	17	output	$\overline{\text{multiplexed address odd B}}$
/ZEB	16	output	$\overline{\text{multiplexed address even B}}$
/ZOC	15	output	$\overline{\text{multiplexed address odd C}}$
/ZEC	14	output	$\overline{\text{multiplexed address even C}}$
NC	12	--	(not used)
VCC	20	power	Vcc
GND	10	ground	ground

PAL 4 (PAL16R8A-2)

Name	Pin No.	Type	Description
/ACE	3	input	$\overline{\text{even access}}$
/ACO	2	input	$\overline{\text{odd access}}$
/BRC	4	input	$\overline{\text{begin refresh cycle}}$
CLK	1	clock input	clock to flip-flops
/A12	5	input	$\overline{\text{A12}}$
A6	6	input	A6
A3	7	input	A3
A2	8	input	A2
A2SEL	9	input	A2 select
/OC	11	input	$\overline{\text{output enable}}$
/REE	16	registered output	$\overline{\text{RAS even enable}}$
/REO	17	registered output	$\overline{\text{RAS odd enable}}$
/CEE	18	registered output	$\overline{\text{CAS even enable}}$
/CEO	19	registered output	$\overline{\text{CAS odd enable}}$
/BRE	14	registered output	$\overline{\text{blocked even refresh}}$
/BRO	15	registered output	$\overline{\text{blocked odd refresh}}$
/MXO	13	registered output	multiplex $\overline{\text{odd}}$/even
LSEL	12	registered output	latch select - partial decoding
VCC	20	power	Vcc
GND	10	ground	ground

Sardis Technologies DRAM Refresh Control Mar. 12 '85

FUSE PATTERN:

PAL20 V1.7D - PAL16R8 - RFSHCTL-1

```
                    11 1111 1111 2222 2222 2233
   0123 4567 8901 2345 6789 0123 4567 8901

 0 -X-- ---- ---- ---- ---- ---- ---- ---- ACO
 1 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 2 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 3 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 4 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 5 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 6 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 7 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 8 ---- -X-- ---- ---- ---- ---- ---- ---- ACE
 9 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
10 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
11 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
12 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
13 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
14 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
15 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

16 X--- X--- X--- ---- ---X ---- ---- ---- /ACO*/ACE*/BRC*BRO
17 X--- -X-- X--- ---- ---X ---- ---- ---- /ACO*ACE*/BRC*BRO
18 X--- -X-- -X-- ---- ---- ---- ---- ---- /ACO*ACE*BRC
19 -X-- X--- ---- ---- ---- ---- ---- ---- ACO*/ACE
20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
21 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
22 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
23 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

24 X--- X--- X--- ---- --X- ---X ---- ---- /ACO*/ACE*/BRC*/BRO*BRE
25 X--- X--- -X-- ---- ---- ---- ---- ---- /ACO*/ACE*BRC
26 X--- -X-- ---- ---- ---- ---- ---- ---- /ACO*ACE
27 -X-- X--- X--- ---- ---- ---X ---- ---- ACO*/ACE*/BRC*BRE
28 -X-- X--- -X-- ---- ---- ---- ---- ---- ACO*/ACE*BRC
29 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
30 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

32 X--- X--- -X-- ---- ---- ---- ---- ---- /ACO*/ACE*BRC
33 -X-- X--- X--- ---- ---X ---- ---- ---- ACO*/ACE*/BRC*BRO
34 -X-- X--- -X-- ---- ---- ---- ---- ---- ACO*/ACE*BRC
35 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
36 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
37 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
38 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
39 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

Sardis Technologies DRAM Refresh Control Mar. 12 '85

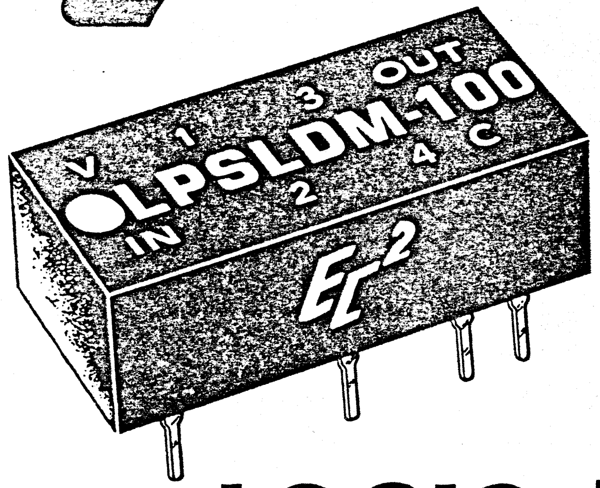
FUSE PATTERN:

PAL20 V1.7D - PAL16R8 - RFSHCTL-1

```
                   11 1111 1111 2222 2222 2233
     0123 4567 8901 2345 6789 0123 4567 8901

  40 X--- X--- X--- ---- ---X ---X ---- ---- /ACO*/ACE*/BRC*BRO*BRE
  41 X--- -X-- X--- ---- ---- ---X ---- ---- /ACO*ACE*/BRC*BRE
  42 X--- -X-- -X-- ---- ---- ---- ---- ---- /ACO*ACE*BRC
  43 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  44 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  45 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  46 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  47 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

  48 X--- X--- X--- ---- --X- ---- ---- ---- /ACO*/ACE*/BRC*/BRO
  49 X--- X--- -X-- ---- ---- ---- ---- ---- /ACO*/ACE*BRC
  50 -X-- X--- ---- ---- ---- ---- ---- ---- ACO*/ACE
  51 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  52 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  53 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  54 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  55 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

  56 ---- ---- ---- X--- ---- ---- ---- ---- /A12
  57 ---- ---- ---- ---- X--- ---- ---- ---- A6
  58 ---- ---- ---- ---- ---- -X-- ---- ---- /A3
  59 ---- ---- ---- ---- ---- ---- X--- -X-- A2*/A2SEL
  60 ---- ---- ---- ---- ---- ---- -X-- X--- /A2*A2SEL
  61 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  62 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  63 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

OUTPUT POLARITY WORD XXXXXXXX

LEGEND:  X : FUSE NOT BLOWN (L,N,0)   - : FUSE BLOWN   (H,P,1)

NUMBER OF FUSES BLOWN =  730

SECURITY FUSE XX

DATA I/O checksum = 5F50
         device code = 22/24
```

```
Sardis Technologies DRAM Multiplexer March 12, 1985

FUSE PATTERN:

PAL20 V1.7D - PAL16L8  - MUX32-1

                    11 1111 1111 2222 2222 2233
    0123 4567 8901 2345 6789 0123 4567 8901

  0 ---- ---- ---- ---- ---- ---- ---- ----
  1 X--- ---- ---- ---- ---- ---- ---X --X- AHA*MXR*/MXE
  2 --X- ---- ---- ---- ---- ---- ---X ---X ALA*/MXR*/MXE
  3 ---- X--- ---- ---- ---- ---- --X- ---- CTRA*MXE
  4 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  5 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  6 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
  7 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

  8 ---- ---- ---- ---- ---- ---- ---- ----
  9 X--- ---- ---- ---- ---- ---- --X- --X- AHA*MXR*MXE
 10 --X- ---- ---- ---- ---- ---- --X- ---X ALA*/MXR*MXE
 11 ---- X--- ---- ---- ---- ---- ---X ---- CTRA*/MXE
 12 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 13 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 14 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 15 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 16 ---- ---- ---- ---- ---- ---- ---- ----
 17 ---- ---- ---- X--- ---- ---- ---X --X- AHB*MXR*/MXE
 18 ---- ---- X--- ---- ---- ---- ---X ---X ALB*/MXR*/MXE
 19 ---- ---- ---- ---- X--- ---- --X- ---- CTRB*MXE
 20 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 21 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 22 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 23 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 24 ---- ---- ---- ---- ---- ---- ---- ----
 25 ---- ---- ---- X--- ---- ---- --X- --X- AHB*MXR*MXE
 26 ---- ---- X--- ---- ---- ---- --X- ---X ALB*/MXR*MXE
 27 ---- ---- ---- ---- X--- ---- ---X ---- CTRB*/MXE
 28 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 29 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 30 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 31 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

 32 ---- ---- ---- ---- ---- ---- ---- ----
 33 ---- ---- ---- ---- ---- ---- X--X --X- AHC*MXR*/MXE
 34 ---- ---- ---- ---- ---- X--- ---X ---X ALC*/MXR*/MXE
 35 ---- ---- ---- ---- ---- ---- --X- X--- CTRC*MXE
 36 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 37 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 38 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
 39 XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
```

```
Sardis Technologies DRAM Multiplexer March 12, 1985

FUSE PATTERN:

PAL20 V1.7D - PAL16L8  - MUX32-1

                    11 1111 1111 2222 2222 2233
    0123 4567 8901 2345 6789 0123 4567 8901

40  ---- ---- ---- ---- ---- ---- ---- ----
41  ---- ---- ---- ---- ---- ---- X-X- --X- AHC*MXR*MXE
42  ---- ---- ---- ---- ---- X--- --X- ---X ALC*/MXR*MXE
43  ---- ---- ---- ---- ---- ---- ---X X--- CTRC*/MXE
44  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
45  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
46  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
47  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

48  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
49  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
50  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
51  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
52  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
53  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
54  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
55  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

56  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
57  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
58  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
59  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
60  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
61  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
62  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX
63  XXXX XXXX XXXX XXXX XXXX XXXX XXXX XXXX

OUTPUT POLARITY WORD XXXXXXXX

LEGEND:  X : FUSE NOT BLOWN (L,N,0)   - : FUSE BLOWN   (H,P,1)

NUMBER OF FUSES BLOWN =  720

SECURITY FUSE XX

DATA I/O checksum = 5A24
         device code = 22/17
```



low profile

T^2L COMPATIBLE LOGIC DELAY MODULE (LOW POWER SCHOTTKY)

- T^2L input and outputs
- Delays stable and precise
- 14-pin DIP package (.250 high)
- Available in delays from 50 to 500ns
- 20% taps – each isolated and with 20 low power Schottky T^2L fan-out capacity
- Rise time 8ns maximum

design notes

The "DIP Series" Low Power Schottky Logic Delay Modules developed by Engineered Components Company have been designed to provide precise tapped delays with required driving and pick-off circuitry contained in a single 14-pin DIP package. These logic delay modules are of hybrid construction utilizing the proven technologies of active integrated circuitry and of passive networks utilizing capacitive, inductive and resistive elements. The ICs utilized in these modules are burned-in to Level B of MIL-STD-883 to ensure a high MTBF. The MTBF on these modules, when calculated per MIL-HDBK-217B for a 50°C ground fixed environment, is in excess of 3 million hours. Module design includes compensation for propogation delays and incorporates internal termination at the output; no additional external components are needed to obtain the desired delay.

The LPSLDM is offered in 22 delays from 50ns to 500ns with each module incorporating taps at 20% increments of total delay. Delay tolerances are maintained as shown in the accompanying part number table, when tested under the "Test Conditions" shown. Delay time is measured at the +1.3V level on the leading edge. Rise time for all modules is 8ns maximum when measured from 0.8V to 2.0V. Temperature coefficient of delay is approximately +500ppm/°C over the operating temperature range of 0 to +70°C.

These modules accept either logic "1" or logic "0" inputs and reproduce the logic at the selected output tap without inversion. The delay modules are intended primarily for use with positive going pulses and are calibrated to the tolerances shown in the table on rising edge delay; where best accuracy is desired in applications using falling edge timing, it is recommended that a special unit be calibrated for the specific application. Each module has the capability of driving up to 20 low power Schottky loads at each tap.

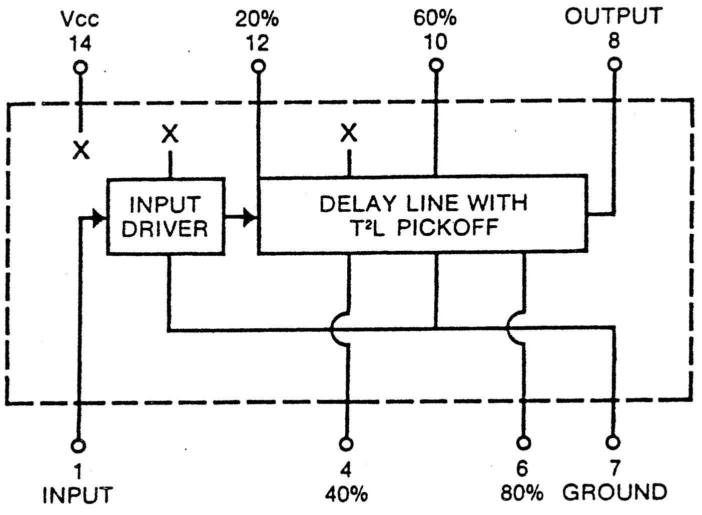
These "DIP Series" modules are packaged in a 14-pin DIP housing, molded of flame-proof Diallyl Phthalate per MIL-M-14, type SDG-F, and are fully encapsulated in epoxy resin. Flat metal leads meet the solderability requirements of MIL-STD-202, Method 208. Leads provide positive stand off from the printed circuit board to permit solder-fillet formation and flush cleaning of solderflux residues for improved reliability.

EC2 engineered components company
3580 Sacramento Drive, San Luis Obispo, California 93401
Phone: (805) 544-3800

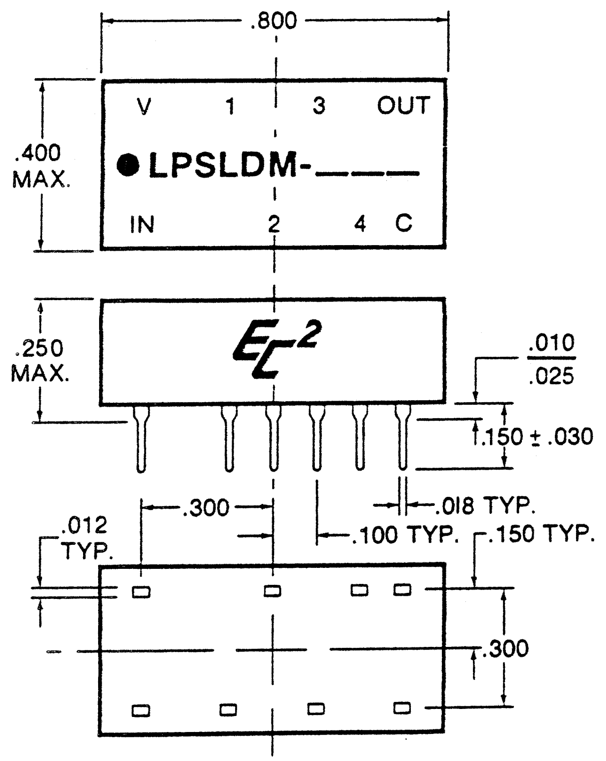
ESIGN NOTES (continued)

Marking consists of manufacturer's name, logo (EC2), part number, terminal identification and date code of manufacture. All marking is applied by silk screen process using white epoxy paint in accordance with MIL-STD-130, to meet the permanency of identification required by MIL-STD-202, Method 215.

BLOCK DIAGRAM IS SHOWN BELOW



MECHANICAL DETAIL IS SHOWN BELOW



EST CONDITIONS

1. All measurements are made at 25°C.
2. V_{CC} supply voltage is maintained at 5.0V DC.
3. All units are tested using a low power Schottky toggle-type positive input pulse and one low power Schottky T^2L load at the output being tested.
4. Input pulse width used is 5 to 10ns longer than full delay of module under test; spacing between pulses (falling edge to rising edge) is three times the pulse width used.

OPERATING SPECIFICATIONS

* V_{CC} supply voltage:	4.75 to 5.25V DC
V_{CC} supply current:	
Constant "0" in	16ma typical
Constant "1" in	3ma typical
Logic 1 input:	
Voltage	2V min.; 5.5V max.
Current	2.4V = 20ua max.
	5.5V = .1ma max.
Logic 0 input:	
Voltage	.8V max.
Current	-.4ma max. (@.4V in)
Logic 1 Voltage out:	2.7V min.
Logic 0 Voltage out:	.5V max.
Operating temperature range:	0 to 70°C.
Storage temperature:	-55 to +125°C.

* Delays increase or decrease approximately 2% for a respective increase or decrease of 5% in supply voltage. and an additional ±2% at 70°C vs 25°C

PART NUMBER TABLE

	ϕ DELAYS AND TOLERANCES (in ns)				
PART NO.	Tap 1	Tap 2	Tap 3	Tap 4	OUTPUT
LPSLDM-50	10 ±1	20 ±1	30 ±1.5	40 ±2	50 ±2
LPSLDM-55	11 ±1	22 ±1	33 ±1.5	44 ±2	55 ±2
LPSLDM-60	12 ±1	24 ±1.5	36 ±1.5	48 ±2	60 ±2
LPSLDM-65	13 ±1	26 ±1.5	39 ±1.5	52 ±2	65 ±2
LPSLDM-70	14 ±1	28 ±1.5	42 ±2	56 ±2	70 ±2.5
LPSLDM-75	15 ±1	30 ±1.5	45 ±2	60 ±2	75 ±2.5
LPSLDM-80	16 ±1	32 ±1.5	48 ±2	64 ±2	80 ±2.5
LPSLDM-85	17 ±1	34 ±1.5	51 ±2	68 ±2	85 ±2.5
LPSLDM-90	18 ±1	36 ±1.5	54 ±2	72 ±2.5	90 ±3
LPSLDM-95	19 ±1	38 ±1.5	57 ±2	76 ±2.5	95 ±3
LPSLDM-100	20 ±1	40 ±1.5	60 ±2	80 ±3	100 ±3
LPSLDM-125	25 ±1	50 ±2	75 ±2.5	100 ±3	125 ±4
LPSLDM-150	30 ±1.5	60 ±2	90 ±3	120 ±4	150 ±5
LPSLDM-175	35 ±1.5	70 ±2.5	105 ±4	140 ±5	175 ±5
LPSLDM-200	40 ±1.5	80 ±2.5	120 ±4	160 ±5	200 ±6
LPSLDM-225	45 ±2	90 ±3	135 ±4	180 ±6	225 ±7
LPSLDM-250	50 ±2	100 ±3	150 ±4.5	200 ±6	250 ±8
LPSLDM-300	60 ±2	120 ±4	180 ±5	240 ±7	300 ±9
LPSLDM-350	70 ±2	140 ±4.5	210 ±7	280 ±9	350 ±11
LPSLDM-400	80 ±3	160 ±5	240 ±7	320 ±10	400 ±12
LPSLDM-450	90 ±3	180 ±6	270 ±8	360 ±11	450 ±14
LPSLDM-500	100 ±3	200 ±6	300 ±9	400 ±12	500 ±15

ϕ All modules can be operated with a minimum input pulse width of 40% of full delay and pulse period approaching square wave; since delay accuracies may be somewhat degraded, it is suggested that the module be evaluated under the intended specific operating conditions. Special modules can be readily manufactured to improve accuracies and/or provide customer specified random delay times for specific applications.

3.0 FLEX SOFTWARE SECTON

3.1 Contents of the "ST-2900 RAM-512 FLEX RAM-DISK" disk

RAMDSK29.CMD	program to install the RAM-Disk driver
VERIFYRD.CMD	utility to check the entire RAM-Disk for checksum errors
FCOPY.CMD	fast file copy program
MEMTEST.BIN	program to test entire RAM-512 memory
RAMDSK29.TXT	source code of above } optional at extra cost
VERIFYRD.TXT	" " " "
FCOPY.TXT	" " " "
MEMTEST.TXT	" " " "

3.2 Installing the ST-2900 FLEX RAM-Disk software

1) Make a copy of the "ST-2900 FLEX RAM-DISK" diskette, then put the original into a safe place and use only the copy from now on.

2) Run the RAMDSK29 program (see section 3.4) with an "FM=Y" parameter. Running the DSKSET program (without parameters) should now show that drive #3 is valid.

3) Run the VERIFYRD program (using all defaults -- don't specify any parameters) to verify the integrity of the data in the newly formatted "disk".

4) The RAM-Disk drive (usually #3) can now be accessed just like any other disk drive by FLEX utilities and applications programs.

5) To automatically install the RAM-Disk software every time you re-boot FLEX:
 a) copy RAMDSK29.CMD onto the system disks you boot from
 b) modify the STARTUP file on those system disks to run RAMDSK29

6) ** VERY IMPORTANT ** The information stored in the RAM-Disk is very volatile, meaning that such things as a momentary "blink" in the power being supplied to the system can result in the data being instantly "erased". To prevent the loss of important data, you should frequently back up from RAM-Disk to floppy disk those files that are new or have been modified.

3.3 RAM-Disk Structure

Each "track" on the RAM-Disk is defined to contain 16 256-byte "sectors", and fits exactly into one 4K page on the RAM-512 board. The RAM-512 board contains 128 such pages, but since page 0 is used to replace the $E000-$EFFF memory on the CPU board, and page 1 is reserved for storing checksums, only 126 "tracks" are allowed, for a total of 2016 sectors.

There is a lot of wasted space on track zero of a FLEX format diskette. Only two sectors are normally needed: sector 03 containing the System Information Record (SIR), and sector 05 containing the first sector of the directory. The ST-2900 RAM-Disk drivers for FLEX (& STAR-DOS) completely eliminate track zero, and remap those two sectors to track 1, resulting in no wasted sectors.

This remapping is transparent to the user, and almost all FLEX programs will run just fine. The types of programs that might have problems are some disk diagnostics, and some high-speed copy programs such as MIRROR or BACKUP, but these usually cannot be run with a RAM-Disk anyways for other reasons.

FLEX will automatically extend the directory if it is full. Allocating an initial directory size of only 1 sector results in additional directory sectors being scattered all over the RAM-Disk, but without any degradation in speed as there would be with a "real" disk.

Re-mapping Table (track/sector)

Sector requested	Sector accessed
00/01 to 00/04	all four are mapped to the same sector (01/01), but only the 00/03 address (SIR) is normally used
00/05	directory sector, mapped to 01/02
00/06 to end of track 0	invalid
01/01 to 01/02	invalid (track zero sectors are mapped here)
01/03 to end of disk	no change

Track 1 sector 01 is stored at $E000-$E0FF on page 2, all the way through to track 126 sector 16 stored at $EF00-$EFFF on page 127.

Each sector has a 2 byte checksum associated with it, and since there are 2016 sectors (126 pages @ 16 sectors), all the checksums for the entire "disk" fit nicely into one 4K page (2016 x 2 < 4096 bytes). By storing the checksums on a different page from their associated sectors, all sizes are convenient powers of 2 and fit without waste -- 256 bytes per sector, 16 sectors per track, one 4K page per track, etc.

The overhead to calculate, store, and compare checksums reduces throughput so slightly that the confidence and peace of mind regarding data integrity afforded by the checksums makes their use very worthwhile.

3.4 RAMDSK29

The RAMDSK29 command is used to install and activate the RAM-Disk device drivers (and optionally re-format the RAM-Disk contents) after the FLEX or STAR-DOS operating system has been booted. Its syntax is:

```
   RAMDSK29[,<parameter list>]
```

where <parameter list> is a list of 2 character parameter codes, each followed by an equals sign "=" and by the value being assigned. Each parameter should be separated by a comma. If any parameter is omitted, a default value will be assigned to it. Some examples:

```
   +++RAMDSK29
   +++RAMDSK29 FM=Y
   +++RAMDSK29 FM=N,TK=30,DR=2
```

The first example installs the drivers using all defaults. Drive #3 will be assigned, 126 tracks will be allocated, and the "disk" will only be re-formatted if any checksum errors are found. The second example is similar to the first, except that the "disk" will be re-formatted without first testing for checksum errors. The third example only allocates 30 tracks, assigns drive #2 instead, and does not re-format the "disk" (checksum errors are not tested for).

Here is a list of all RAMDSK29 parameters:

FM=x   re-ForMat the "disk"   (default = E)

The alphabetic character "x" can be one of 3 values:
  "N" - No, do not re-format the "disk" (and don't test for checksum errors)
  "Y" - Yes, re-format the "disk" (but don't test for checksum errors)
  "E" - only re-format the "disk" if any checksum Error is discovered

TK=ddd   number of tracks to be allocated   (default = 126)

The decimal value "ddd" specifies the number of tracks to be allocated to the RAM-Disk. Each "track" is 16 256-byte "sectors" and occupies one 4K page on the RAM-512 board. For a 128K board you can specify any value from 1 to 30, while a 512K board lets you allocate 1 to 126 "tracks". Allocating less than the maximum number of tracks allows you to reserve some 4K pages at the top of the RAM-512's space if you want to use them for other purposes such as a printer buffer.

DR=d   DRive number to be assigned   (default = 3)

The decimal value "d" specifies which drive number (1, 2 or 3) FLEX is to use when accessing the RAM-Disk. The drive number must not already be allocated for a floppy disk.

The RAMDSK29 command also patches FLEX to greatly speed up the LOADCODE ($CD30) system call. As Leo Taylor has previously pointed out (68MJ 85Oct p.39, 68MJ 82Apr p.25), FLEX's FMS ($D406) system call involves a lot of overhead. The original LOADCODE routine calls the FMS routine for each byte of the program being loaded. After being patched, LOADCODE only calls FMS for the first and last byte of each sector, and directly accesses the sector buffer for the middle 250 bytes of each sector. The performance improvement is significant -- a 20K program that would otherwise take approx. 5 seconds to load from RAM-Disk now loads in approx. 1.3 seconds (@ 1 MHz). Load times from a hard disk would be similarily improved. Yet unlike the FLEX version of VDISK (another commercially available RAM-Disk package), the binary file doesn't need to be converted into any special format to achieve this speedup.

Both the RAM-Disk driver routines and the replacement LOADCODE routine are loaded below MEMEND; MEMEND is then reduced by the approx. 430 bytes that they occupy.

Note -- every time FLEX (or STAR-DOS) is re-booted, RAMDSK29 must be run to re-install the RAM-Disk drivers. However, depending on whether or not the RAM-Disk contents are still intact, re-formatting of the RAM-Disk may not be necessary (use FM=E).

If you need ~~For~~ more information, ~~study the supplied~~ source code is available on disk.

3.5 VERIFYRD

The VERIFYRD command is used to test EVERY sector in the RAM-Disk for checksum errors, and to optionally re-format the "disk". Its syntax is:

```
   VERIFYRD[,<parameter list>]
```

where <parameter list> is a list of 2 character parameter codes, each followed by an equals sign "=" and by the value being assigned. Each parameter should be separated by a comma. If any parameter is omitted, a default value will be assigned to it. Some examples:

```
   +++VERIFYRD
   +++VERIFYRD DR=2,FM=E
```

The first example assumes the RAM-Disk is installed as drive #3, and reports any checksum errors found, but does not re-format the "disk" even if any errors are found. In the second example, VERIFYRD is pointed to drive #2 (because the RAM-Disk was, for whatever reason, assigned to that drive when installed by RAMDSK29), all checksum errors found are reported, and if any errors are found, the "disk" is automatically re-formatted.

Here is a list of all VERIFYRD parameters:

DR=d DRive number to check   (default = 3)

The decimal value "d" specifies which drive number (1, 2 or 3) FLEX is to check. The drive should be a RAM-Disk drive, not a floppy disk.

FM=x   re-ForMat the "disk"   (default = N)

The alphabetic character "x" can be one of 3 values:
  "N" - No, do not re-format the "disk" even if checksum errors are found
  "Y" - Yes, re-format the "disk" even if no checksum errors are found
  "E" - only re-format the "disk" if any checksum Error is discovered

Testing all 126 "tracks" only takes approx. 8 seconds, so if you typically leave your system powered up for days on end, we suggest you run VERIFYRD every morning.

If you need ~~For~~ more information, ~~study the supplied~~ source code is available on disk.

3.6 FCOPY

The FCOPY command is used to make a copy of a file, either to the same , or onto another drive. Its main advantage over the COPY command supplied with FLEX (or STAR-DOS) is that it is significantly faster when copying large files, where the source and/or destination drive is a RAM-Disk or hard disk. For example, to do a RAM-Disk to RAM-Disk copy of a 50 sector file takes 7 seconds with TSC's COPY, but only 2 seconds with FCOPY (@ 1 MHz). Its limitation is in only being able to copy one file per run. The syntax of FCOPY is:

```
    FCOPY,<file spec>,<file spec>[,+<option list>]
```

where the first <file spec> is the name of the file to be copied (default extension is .TXT), the second <file spec> is the name of the file to be created (the default extension is .TXT), and the optional option list consists of the letter "D" and/or "S". Some examples:

```
   +++FCOPY BOOK1,BOOK2
   +++FCOPY 1.CHAPTER.OUT,CHAPTERX.BAK.3,+DS
```

In the first example, the file BOOK1.TXT on the default working drive (refer to the FLEX "ASN" command) is copied to a new file called BOOK2.TXT, also on the working drive. The file creation date for BOOK2 is copied from BOOK1. In the second example, the file CHAPTER.OUT on drive 1 is copied to a new file called CHAPTERX.BAK on drive 3. The current system date is used as the file creation date, and a small buffer is used so user memory from $0000 to MEMEND is not disturbed.

Here is a list of the available options:

D tells FCOPY to use the current system date as the creation date of the output file. If this option is omitted, the creation date of the output file remains identical to that of the input file.

(cont'd)

S- tells FCOPY to use a small (1 sector) buffer so that the entire program will fit into FLEX's Utility Command Area ($C100-$C6FF) without affecting any user memory ($0000 to MEMEND). If this option is omitted, FCOPY uses all of user memory as a buffer. The use of this option lets you call FCOPY from programs such as TSC's XBASIC without clobbering the calling program.

If you need ~~For~~ more information, ~~study the supplied~~ source code is available on disk.

3.7 MEMTEST

MEMTEST is a program to test all 512K bytes on an ST-2900 RAM-512 board. It uses ST-MON I/O routines, so does not need an operating system in order to run. It is stored as a binary file on a FLEX format disk, and has been "linked" by FLEX's LINK command, so can be loaded and run with ST-MON's "D F" command -- even if you only have an OS-9 system.

If you don't have an ST-2900 FDC board connected (or it isn't working), you can type in the object code of MEMTEST from the listing in section 3.8.

MEMTEST performs up to 65,536 passes over all 512K bytes. An asterisk "*" is displayed to the screen at the end of each pass (which takes approx. 52 sec. @ 1 MHz). In the first part of each pass, test patterns are written to the entire 512K bytes. After pausing for 200 msec. to flush out potential dynamic memory refresh problems, the test patterns are read back. Any locations with errors are displayed on the terminal to indicate the address (page # and address within page), and which bits were in error. For example:

```
=J 1000**** 3E E790 06 **
```

In this example, MEMTEST was loaded manually starting at $1000, and ST-MON's "J hhhh" command was used to execute it. The first 4 passes (represented by the 4 asterisks) were OK, then on the fifth pass two bits (represented by the value $06) were in error. The error occurred in page $3E at address $E790. Each of the 8 bits in a byte in memory are stored in a different memory chip on the RAM-512 board, with bytes located on an even address in chips U27-U34, and those with an odd numbered address in chips U19-U26. Chips U19 and U27 store the high order bit of each byte, and U26 and U34 store the low order bits. Thus the bit error code of $06 at address $E790 indicates chips U32 and U33 are partially bad.

Since a full 65,536 passes would take 19 days (65,536 x 52 sec.), the test can be aborted at the end of any pass by keying any character on the keyboard.

The current version of MEMTEST occupies memory from $1000-$110F, and uses $2000-$2FFF to save the contents of the RAM-512's page 0 so it can restore the data in that page before exitting the program.

For more details study the listing in section 3.8.

3.8 Appendix A - MEMTEST Listing

```
                    ************************************************
                    *        "MEMTEST.BIN"
                    * TEST EXTENDED MEMORY ON ST-2900 RAM-512 BOARD
                    *
                    * NOTE - it destroys all of the data previously stored
                    *        on the RAM-512 board, except for page 0 data.
                    *        Takes approx. 52 seconds per pass for 512K
                    *        @ 1 MHz, or 19 days for 65536 passes!!
                    *
                    * To run:
                    *    +++GET MEMTEST.BIN
                    *    +++MON
                    *    =J 0100
                    *    =F W
                    *  or use the "D F" command after powerup/reset,
                    *   as MEMTEST.BIN is "linked".
                    *
                    * (c) 1985 by Sardis Technologies,
                    *           all rights reserved
                    * Last modified November 11, 1985 1:30 pm
                    ************************************************
                    *
                    * EQUATES
                    *
              007F  HIBANK  EQU    $7F       $1F/$7F FOR 128K/512K <<<<<<<<
              FFB8  RAMBD1  EQU    $FFB8     CONTROL PORT FOR RAM-512 BOARD #1
              FFBC  RAMBD2  EQU    $FFBC     CONTROL PORT FOR RAM-512 BOARD #2
              2000  SAVBUF  EQU    $2000     a 4096 byte save buffer <<<<<<<
              0004  EOT     EQU    $04

              FED7  OUTCH   EQU    $FED7     indirect
              FED9  OUT1SP  EQU    $FED9     .  "
              FEDD  OUT2HX  EQU    $FEDD     .  "
              FEDF  OUT4HX  EQU    $FEDF     .  "
              FEC5  INCHEK  EQU    $FEC5     .  "
              FEC7  INCH8   EQU    $FEC7     .  "
              FEE5  PSTRNG  EQU    $FEE5     .  "
              FEC3  SIGNON  EQU    $FEC3     .  "


1000                        ORG    $1000
                    *
                    * INITIALIZE
                    *
1000 34    01       MEMTEST PSHS   CC        DISABLE INTERRUPTS
1002 1A    50               ORCC   #$50
1004 32    7C               LEAS   -4,S      RESERVE SPACE ON STACK
1006 4F                     CLRA             INIT PASS # TO 0000
1007 5F                     CLRB
1008 ED    E4               STD    0,S
                    *
                    * SAVE PAGE E (BANK 0)
                    *
100A C6    80               LDB    #$80
100C F7    FFB8             STB    RAMBD1
100F 8E    E000             LDX    #$E000
```

```
1012 108E 2000              LDY    #SAVBUF
1016 EC   81        MT05    LDD    ,X++
1018 ED   A1                STD    ,Y++
101A 8C   F000              CMPX   #$F000
101D 25   F7                BLO    MT05
                    *
                    * CALC PSEUDO-RANDOM NUMBER BY ACCUM 8 BIT CHECKSUM
                    *  OF MEMORY FROM $0000-$DFFF
                    *
101F 8E   0000              LDX    #$0000    START AT $0000
1022 86   A5                LDA    #$A5      INIT CHECKSUM
1024 A8   80        MT10    EORA   ,X+       ACCUM CHECKSUM
1026 8C   E000              CMPX   #$E000    DONE?
1029 25   F9                BLO    MT10      .N
102B A7   63                STA    3,S       .Y, STORE

                    *
                    * STORE PATTERN INTO ALL BANKS OF RAM-512 BOARD
                    *
102D 86   7F        MT20    LDA    #HIBANK   INIT BANK #
102F A7   62                STA    2,S
1031 108E E000      MT30    LDY    #$E000    BANK BEGINS AT $E000
1035 A6   62                LDA    2,S       SELECT BANK
1037 8A   80                ORA    #$80
1039 B7   FFB8              STA    RAMBD1
103C 1F   20        MT35    TFR    Y,D       USE ADDRESS
103E A3   E4                SUBD   0,S       SUBTRACT PASS #
1040 34   02                PSHS   A         ADD MSB + LSB
1042 EB   E0                ADDB   ,S+
1044 E0   62                SUBB   2,S       SUBTRACT BANK #
1046 EB   63                ADDB   3,S       ADD RANDOM #
1048 E7   A0                STB    ,Y+       STORE INTO MEMORY
104A 108C F000              CMPY   #$F000    BANK DONE?
104E 25   EC                BLO    MT35      .N
1050 6A   62                DEC    2,S       ALL BANKS DONE?
1052 2A   DD                BPL    MT30      .N
                    *
                    * DELAY FOR 200 MSEC. TO TEST FOR DYNAMIC MEMORY REFRESH
                    *  PROBLEMS (DO NOT COMBINE THE TWO LOOPS BELOW INTO ONE,
                    *  NOR INTO 2 CALLS TO ONE SUBROUTINE)
                    *
1054 86   9C                LDA    #156      (2)
1056 5F             MT40    CLRB             (2) \
1057 5A             MT42    DECB             (2)  > 1282 CYCLES
1058 26   FD                BNE    MT42      (3) /
105A 4A                     DECA             (2)
105B 26   F9                BNE    MT40      (3)

105D 86   9C                LDA    #156      (2)
105F 5F             MT46    CLRB             (2)
1060 5A             MT48    DECB             (2)
1061 26   FD                BNE    MT48      (3)
1063 4A                     DECA             (2)
1064 26   F9                BNE    MT46      (3)
```

```
                        *
                        * READ PATTERN BACK TO SEE IF STILL OK
                        *
1066 86   7F                    LDA    #HIBANK   INIT BANK #
1068 A7   62                    STA    2,S
106A 108E E000          MT50    LDY    #$E000    BANK BEGINS AT $E000
106E A6   62                    LDA    2,S       SELECT BANK
1070 8A   80                    ORA    #$80
1072 B7   FFB8                  STA    RAMBD1
1075 1F   20            MT55    TFR    Y,D       USE ADDRESS
1077 A3   E4                    SUBD   0,S       SUBTRACT PASS #
1079 34   02                    PSHS   A         ADD MSB + LSB
107B EB   E0                    ADDB   ,S+
107D E0   62                    SUBB   2,S       SUBTRACT BANK #
107F EB   63                    ADDB   3,S       ADD RANDOM #
1081 E8   A0                    EORB   ,Y+       DATA MATCHES?
1083 27   20                    BEQ    MT60      .Y
1085 8D   69                    BSR    OUTSPC    .N, OUTPUT BANK/ADDRESS/BITS
1087 A6   62                    LDA    2,S       OUTPUT BANK #
1089 AD   9F FEDD               JSR    [OUT2HX]
108D 8D   61                    BSR    OUTSPC
108F 30   3F                    LEAX   -1,Y      OUTPUT ADDRESS
1091 AD   9F FEDF               JSR    [OUT4HX]
1095 8D   59                    BSR    OUTSPC
1097 1F   98                    TFR    B,A       OUTPUT BIT ERROR CODE
1099 AD   9F FEDD               JSR    [OUT2HX]
109D 8D   51                    BSR    OUTSPC
109F AD   9F FEC5               JSR    [INCHEK]  ABORT THIS ROUTINE?
10A3 26   29                    BNE    MT89      .Y

10A5 108C F000          MT60    CMPY   #$F000    BANK DONE?
10A9 25   CA                    BLO    MT55      .N
10AB 6A   62                    DEC    2,S       ALL BANKS DONE?
10AD 2A   BB                    BPL    MT50      .N
10AF 86   2A                    LDA    #'*       .Y, OUTPUT "*"
10B1 AD   9F FED7               JSR    [OUTCH]
10B5 AD   9F FEC5               JSR    [INCHEK]  ABORT THIS ROUTINE?
10B9 26   13                    BNE    MT89      .Y
10BB AE   E4                    LDX    0,S       INCR. PASS COUNT
10BD 30   01                    LEAX   1,X
10BF AF   E4                    STX    0,S
10C1 1026 FF68                  LBNE   MT20      DO NEXT PASS
10C5 30   8C 2C                 LEAX   <MSG1,PCR "all passes done"
10C8 AD   9F FEE5               JSR    [PSTRNG]
10CC 20   04                    BRA    MT90
                        *
                        * RESTORE PAGE $EXXX AND END
                        *
10CE AD   9F FEC7       MT89    JSR    [INCH8]   DISCARD ABORT CHARACTER

10D2 C6   80            MT90    LDB    #$80
10D4 F7   FFB8                  STB    RAMBD1
10D7 8E   2000                  LDX    #SAVBUF
10DA 108E E000                  LDY    #$E000
10DE EC   81            MT95    LDD    ,X++
10E0 ED   A1                    STD    ,Y++
10E2 108C F000              CMPY   #$F000
10E6 25   F6                BLO    MT95

10E8 32   64                LEAS   4,S
10EA 35   01                PULS   CC
10EC 6E   9F FEC3           JMP    [SIGNON]

                    *
                    * OUTPUT 1 SPACE CHARAC
                    *
10F0 6E   9F FED9   OUTSPC  JMP    [OUT1SP]

                    *
                    * DATA AREA
                    *
10F4 3E 3E 20 36    MSG1    FCC    '>> 65,536 passes done !! <<',EOT
10F8 35 2C 35 33
10FC 36 20 70 61
1100 73 73 65 73
1104 20 64 6F 6E
1108 65 20 21 21
110C 20 3C 3C 04

               110F ENDTST  EQU    *-1

                            END    MEMTEST

0 ERROR(S) DETECTED

SYMBOL TABLE:

ENDTST 110F   EOT    0004   HIBANK 007F   INCH8  FEC7   INCHEK FEC5
MEMTES 1000   MSG1   10F4   MT05   1016   MT10   1024   MT20   102D
MT30   1031   MT35   103C   MT40   1056   MT42   1057   MT46   105F
MT48   1060   MT50   106A   MT55   1075   MT60   10A5   MT89   10CE
MT90   10D2   MT95   10DE   OUT1SP FED9   OUT2HX FEDD   OUT4HX FEDF
OUTCH  FED7   OUTSPC 10F0   PSTRNG FEE5   RAMBD1 FFB8   RAMBD2 FFBC
SAVBUF 2000   SIGNON FEC3
```

4.0 OS-9 SOFTWARE SECTION

4.1 Contents of the "ST-2900 OS-9 RAM-DISK" disk

DRIVERS directory:

Ramdsk09	RAM-Disk device driver
RO	RAM-Disk device descriptor

CMDS directory:

Vfyramd	utility to check the entire RAM-Disk for checksum errors

SOURCE directory:

Ramdsk09	source code of above	optional at extra cost
RO	" " " "	
Vfyramd	" " " "	

4.2 Installing the OS-9 RAM-Disk software

1) Make a copy of the "ST-2900 OS-9 RAM-DISK" diskette, then put the original into a safe place and use only the copy from then on.

2) Use OS-9's "LOAD" command to load "RAMDSK09" and "RO" from the "DRIVERS" directory into memory.

3) Use SFORMAT to format the /RO virtual drive -- only the device name and volume name parameters should be specified.

4) /RO can now be accessed just like any other disk drive by OS-9 utilities and application programs.

5) ** VERY IMPORTANT ** The information stored in the RAM-Disk is very volatile, meaning that such things as a momentary "blink" in the power being supplied to the system can result in the data being instantly "erased". To prevent the loss of important data, you should frequently back up from RAM-Disk to floppy disk those files that are new or have been modified.

6) There are two ways to automatically install the RAM-Disk every time you boot OS-9:
 a) create a new "OS9Boot" file that includes the RAMDSK09 and RO modules (take note of the warning in section 4.5)
 b) put the RAMDSK09 and RO files onto the system disk and add LOAD commands to the "startup" file
In either case you will likely want to add the following command to the "startup" file to format the RAM-Disk:

```
   VFYRAMD +F
```

After the RAM-Disk is re-formatted, you may want to copy your entire system disk to the RAM-Disk. Follow the hints on pp.34-35 of the ST-2900 OS-9 Conversion Package manual re using DSAVE/makecopy. Lastly, execute the CHX /RO/CMDS and CHD /RO commands, and OS-9 will begin to fly for you!!

4.3 RAM-DISK STRUCTURE

Each "track" on the RAM-Disk is defined to contain 16 256-byte "sectors", and fits exactly into one 4K page on the RAM-512 board. The RAM-512 board contains 128 such pages, but since page 0 is used to replace the $E000-$EFFF memory on the CPU board, and page 1 is reserved for storing checksums, only 126 "tracks" are allowed, for a total of 2016 "sectors".

LSN 0 is stored at $E000-$E0FF on page 2, all the way through to LSN 2015 stored at $EF00-$EFFF on page 127.

Each sector has a 2 byte checksum associated with it, and since there are 2016 sectors (126 pages @ 16 sectors), all the checksums for the entire "disk" fit nicely into one 4K page (2016 x 2 < 4096 bytes). By storing the checksums on a different page from their associated sectors, all sizes are convenient powers of 2, and fit without waste -- 256 bytes per sector, 16 sectors per track, one 4K page per track, etc.

The overhead to calculate, store, and compare checksums reduces throughput so slightly that the confidence and peace of mind regarding data integrity afforded by the checksums makes their use very worthwhile.

The RAM-Disk is fast enough that if you need to write a program that uses a very large array, it is feasible to implement the array as a random access file and use OS-9's I$SEEK, I$READ, I$WRITE system calls to point to, read, and write array elements.

4.4 VFYRAMD

Syntax: VFYRAMD [/devname] [+F]

The VFYRAMD command is used to test all sectors in the RAM-Disk for correct checksums and to report all sectors with errors. The "/devname" parameter is the name of the RAM-Disk drive to be checked (usually "/R0") -- if this parameter is omitted, device /R0 is assumed. When the "+F" parameter is specified, if any bad checksum is discovered, further testing is aborted and the SFORMAT program (supplied with the ST-2900 OS-9 Conversion Package) is automatically called to re-format the RAM-Disk.

If you need ~~For~~ more information, ~~study the supplied~~ source code is available on disk.

4.5 RAMDSK09

The RAMDSK09 module is a "device driver", not a command. In conjunction with the /RO device descriptor, it "fools" OS-9 into thinking it has an additional floppy disk drive (but an extremely fast one) connected. The data in this "drive" is not stored on a real floppy diskette, but on the RAM-512 memory board.

VERY IMPORTANT -- If the RAMDSK09 module is put into the "OS9Boot" file, it should be added to the beginning of the file, not the end, to ensure that the "MOVE" subroutine in RAMDSK09 does not occupy any part of the $E000-$EFFF address range.

If you need ~~For~~ more information, ~~study the supplied~~ source code is available on disk.

4.6 RO Device Descriptor

The "RO" device descriptor is used to tell the RAMDSK09 device driver and the rest of the operating system the attributes of the virtual disk, such as the number of physical cylinders.

If you need ~~For~~ more information, ~~study the supplied~~ source code is available on disk.

4.7 Appendix A - PRINTERR and RAM-Disk

Once you have pointed the current execution and data directories to RAM-Disk, you would think that OS-9's PRINTERR command would look for the ERRMSG file on RAM-Disk. Not so!! Once you have executed the PRINTERR command you will first have to run DEBUG and perform the following three steps:

a) find the "/DO/SYS/ERRMSG" string within module "PRINTERR" in memory
b) change the first part of the string from "/DO" to "/RO"
c) run MODFIX to update the CRC of the PRINTERR module