TEK SOFTWARE DEVELOPMENT PRODUCTS

VOL. IV, ISSUE NO. 3
DECEMBER, 1985

USER GROUP NEWS

3

3 8-BIT MICROPROCESSOR SUPPORT SYSTEM FOR Z80, Z80C, 6809, 6809E, AND 8085

DESIGN AUTOMATION GROUP

Tektronix®
COMMITTED TO EXCELLENCE

FORWARD

We are reformatting the PRODUCT PERFORMANCE REPORT (PPR) SECTION to include easy to read headers. This new formatting will not only provide more detailed information requested from our customers, but it will also be easier reading for quicker referencing.

ABOUT THE "USER GROUP LIBRARY SECTION"

Programs reported in the User Group Library will be available through your Tektronix Applications Engineer. When updates are available, they will be reported in each issue and a separate total listing will be produced annually.

REGISTERED TRADEMARKS

The following trademarks are listed throughout this issue of USER GROUP NEWS. For your convience, we have selected not to list them as footnotes at the bottom of each issue. The trademarks included in this issue are as follows:

VAX, DEC, VMS & ULTRIX are registered trademarks of Digital Equipment Corporation.

UNIX is a registered trademark of AT&T Bell Laboratories.

TNIX is a registered trademark of Tektronix, Inc.

VRTX is a registered trademark of Hunter & Ready

IBM / PC AT XT is a registered trademark of IBM Corporation

Myrl Kwiatkowski

Editor

CONTENTS

GENERAL INFORMATION SECTION

INTRODUCING TEKMATE

TEKMATE

Microcontroller and Microprocessor
Support Systems From SDP

NOW AVAILABLE!

Tektronix announces SPD's TekMate 8-bit systems emulation support for Z80, Z80C, 6809, 6809E and 8085 microprocessors.

SDP is addressing the needs of both the individual designers and the large development groups by providing a powerful host-compatible analysis tool at a budgetable price.

From Concept to code TekMate's development systems answers your design strategy needs.

TekMate ADVANTAGES

TekMate offers many key features.

- Non-Stop Execution Analysis
- Real-Time Trace
- 64K No Wait-State Memory
- Excellent Price / Performance
- Friendly User Interface
- Micro Controller Support (68HC11 coming soon)
- Tektronix Support

PRODUCT DESCRIPTION

TekMate is a powerful emulation system for 8-bit microprocessor and microcontroller chips. Advanced hardware and software analysis tools make TekMate an extremely capable development support system. TekMate can support all emulation needs without adding hardware options. **TekMate is compatible with most popular host computers.** The TekMate system can be easily moved to other hosts and used as a shared resource on your system. (i.e., A TekMate system being used on a VAX can be moved to an IBM PC by adding the IBM PC software support). TekMate's automatic interface configuration eliminates setting baud-rate and parity switches each time the system is moved. **TekMate meets the needs of both individual designers and large development groups.**

The operation and high performance of TekMate make the user more productive. **A familiar interface, in the same format as the host resource, make TekMate easy to understand, even for new and infrequent users.** TekMate's features include **non-stop, real-time and transparent operation, no wait-state memory, expanded breakpoint capability, ease of use and low price. These features make TekMate an extremely cost effective and powerful 8-bit emulation support system.**

Host Compatibility

TekMate support tools are available for the IBM PC AT/XT, VAX VMS V4.X, μVAX VMS, and the Tektronix 8560 series. Auto baud rate and auto parity select (baud rates from 300 to 19.2K) allow TekMate to be easily connected to the host by **providing an automated setup.**

Expanded Prospective Customer Base

TekMate is useful and cost effective throughout the life cycle of product development. TekMate supports design engineering, as well as being suitable for engineering test, service support, and manufacturing applications.

User Interface

TekMate control software includes a host-specific user interface, as well as a command interface. The host command interface consists of a "PC Tool"-like interface for the IBM PC, or a COLORKEY+ interface for VAX VMS or 8560 series. A direct command interface is also offered in each environment and is adapted to the syntax of the specific host.

TekMate FEATURES

Nonstop Emulation

Many prototypes can not be conveniently stopped and restarted. Interruption of the emulation process could result in a time-consuming reestablishment of the conditions that existed prior to the emulation break. **TekMate solves this problem.** TekMate's Real-Time Trace capability allows the user to analyze and evaluate microprocessor activity **while the emulation activity continues uninterrupted. Nonstop emulation lets you observe activity in the prototype without execution interruption.**

Real-Time and Transparent Emulation

The operation of the prototype with the chip is predictable when the emulation support is both real-time and transparent. Observations and measurements obtained using TekMate accurately reflect the operation of the prototype when the emulator is replaced by the chip.

Register Trace

TekMate allows the user to find logic problems in embedded code and hardware using Register Trace. After problems are detected, the user can focus on the problem by establishing address ranges of interest. With up to 16 address ranges established, only relevant register data will be displayed.

Bus Trace

TekMate allows the user to find functional problems in the integrated code and hardware in real-time by displaying the Real-Time Trace Buffer while the emulator runs continuously or is halted as required. The user can then focus on the problem by selecting the address areas to be displayed by using the 16 register trace address ranges to select the data to be acquired.

Emulation Break

Certain types of information, such as memory and register content relative to a specific event are captured by stopping emulation. TekMate allows the user to halt emulation in a way that preserves the condition of the system.

Memory Support

64K of real-time no wait-state memory is standard with TekMate. The user can select the address ranges that will access prototype or emulation memory on 256K byte boundaries throughout the address range of the microprocessor.

Symbolic Debug

Symbolic Debug allows the user to debug code utilizing the symbolic values created while writing the code. A virtual symbol table provides an unlimited number of symbols, allowing the designer to integrate and debug at the same level of abstraction used throughout the code development process.

Three Emulation Modes

TekMate can provide resources when a prototype is not ready or fully functional. The designer is provided with a powerful tool for software analysis, integration, and system evaluation.

- Mode 0 - All TekMate support resources are provided. No prototype is required.
- Mode 1 - User clock, Mappable memory, User I/O, and SVCs - Full or Partial (selectable) use of prototype resources
- Mode 2 - User clock, User memory, User I/O, and SVCs - Full use of prototype resources

True Processor Reset

On reset, many chips leave some registers in a random state. The TekMate reset command performs a hardware reset on the CPU and preserves the state of the registers without artificial initialization, reflecting the true reset status of the chip.

8540A Compatible

- TekMate commands are based on the 8540A command set, making TekMate compatible with the 8540A.
- Full simulated I/O support.
- 8540A load file support

Trigger In and Out

- A maskable trigger input is provided to allow external equipment to halt the emulation process.
- Trigger out pulse occurs on an emulation break. The leading edge occurs when the break conditions are met, and trailing edge occurs when the emulator has halted.

User Support

- Extensive On-line Help
- Informative Error Messages
- Detailed Manuals
- Purchasable Applications Engineering Service
- User Group News Subscription

Software Support

TekMate has been designed to accept the Tektronix standard load formats of SAS and LAS binary, extended TekHex, and the industry vendor standard formats of Intel Hex and Motorola Hex. This range of supported formats allows the use of language support tools from Tek and third party vendors.

Micro		Host		
	IBM PC	**VAX-VMS**	**μVAX-VMS**	**8560**
Z80	ASM PASCAL C	ASM PASCAL C	ASM PASCAL C	ASM
6809	ASM	ASM	ASM	ASM
8085	ASM PASCAL C	ASM PASCAL C	ASM PASCAL C	ASM

Notes:

Tektronix distributed Microtec Research Inc., products provides PASCAL and C for the Z80 and 8085. They also provide all support listed for the IBM PC. Third Party Software is available for the above processors.

TekMate is fully compatible with the existing Z80, 8085, and 6809 assemblers from SDP (includes linker, loader, library generator).

Environmental Characteristics

	Temperature Range
Operating:	0°C to 40°C
Storage:	-55°C to 75°C
	Altitude Range
Operating:	Sea Level to 4,500m
Storage:	Sea Level to 15,000m
	Humidity
	0 to 90% non-condensing (0°C to 40°C)
	Power Requirements
Line Voltages:	90 to 132 VAC 180 to 250 VAC
Line Frequency:	48 to 66 Hz
	Dimensions
Width:	14.5"
Depth:	17.5"
Height:	5.25"
	Weight
Net:	18 lbs. 12 oz.
Shipping:	30 lbs. 10 oz.

ORDERING INFORMATION

PRODUCT	DESCRIPTION	AVAILABILITY
	EMULATION SYSTEMS (Emulation control S/W listed Below)	
EZ80	Z80 Emulator	8 Weeks
Opt. 17	CMOS Support	8 Weeks
E6809	6809 Emulator	10 Weeks
Opt. 17	6809E Support	10 Weeks
E8085	8085 Emulator	12 Weeks
Opt. N1	WARRANTY PLUS Maint. Cont.	
Opt. A1	220V Euro Plug	
Opt. A2	220V UK Plug	
Opt. A3	220V AUST. Plug	
Opt. A4	220V N. AMR Plug	
Opt. A5	220V SWISS Plug	

Note No RS232 cable is provided, see Configuration Notes below.

	EMULATION CONTROL S/W Option Required	
SZ80	Z80 Emu. Control S/W	8 Weeks
S6809	6809 Emu. Control S/W	10 Weeks
S8085	8085 Emu. Control S/W	12 Weeks
Opt. 1A	8560 Support	8 Weeks
Opt. 1E	Vax VMS TU-58 Cas.	22 Weeks
Opt. 1F	Vax VMS tape	22 Weeks
Opt. 1M	MVax VMS 5-1/4 Flop.	22 Weeks
Opt. 1N	MVax VMS TK-50 Cas.	22 Weeks
Opt. 1Y	IBM PC Support 5-1/4 Flop.	12 Weeks
Opt. 3S	Additional 12 Month Coverage	
Opt. 3A	SSS +Update Kit - 856X Flop.	
Opt. 3E	SSS +Update Kit - VAXVMS TU-58	
Opt. 3F	SSS +Update Kit - VAXVMS Tape	
Opt. 3M	SSS +Update Kit - MVAXVMS Flop.	
Opt. 3N	SSS +Update Kit - MVAXVMS TK-50	
Opt. 3Y	SSS +Update Kit - IBM PC Flop.	

OPTIONAL ACCESSORIES

PART NUMBER	DESCRIPTION	FUNCTION
012-0911-00	**RS232 Cable 12'**	**Connects TekMate to the IBM PC**
012-1162-00	**RS232 Cable 10'**	**Connects TekMate to the 8560**
014-0061-00	**Stand**	**Supports TekMate on its side.**
070-5728-00*	**User Manual**	**TekMate Z80 Emulator Specifics**
070-5730-00*	**User Manual**	**TekMate 6809 Emulator Specifics**
070-5829-00*	**User Manual**	**TekMate 8085 Emulator Specifics**
070-5729-00*	**User Manual**	**TekMate System User Manual**
070-5739-00	**Service Manual**	**TekMate Z80 Emulator Specifics**
070-5741-00	**Service Manual**	**TekMate 6809 Emulator Specifics**
070-5830-00	**Service Manual**	**TekMate 8085 Emulator Specifics**
070-5738-00	**Service Manual**	**TekMate System Service Manual**

*** Provided with system**

For price information contact your Tektronix sales representative.

Configuration Notes

The TekMate consists of two parts: hardware (the E-system) and emulation control software (the S-system). You can order as many E-systems as needed for the chip you wish to emulate and one S-system for each host needed.

For example, a design team might be using five EZ80 systems on a VAX. The order would be: five EZ80, and one SZ80 Option 1F. If one of the EZ80's is to be used with an IBM PC occasionally then one Z80 Option 1Y would be added to the order.

Note: No RS232 cables are provided with the system. The user can provide cables or order cables as listed under the Optional Accessories section above for the IBM PC or the 8560. No standard cable is provided for the VAX system. The RS232 port on the TekMate system is a standard female RS232 connector. If the host connection is configured for terminal equipment connection, a null terminal configured cable will be required. If the host connection is configured for data communications equipment, a straight-line cable will be required.

John Owens SDP Marketing

ICOM40A SPEED-UP

New releases of ICOM40A are available.

ICOM40A Opt 1C for UNIX 4.2	V02.01-00
ICOM40A Opt 1E & 1F for VMS V4	V02.02-00 1E
	V04.05-02 1F

These versions have significantly improved download times, specifically when code and symbols are downloaded. The following benchmarks were run showing the improved timing.

	Filesize	Symbol/Code	Old "ICOM lo"	New "ICOM lo"
UNIX	192K	90%/10%	4 min 43 sec	23 sec
	175K	48%/52%	5 min 19 sec	2 min 45 sec
	68K	0%/100%	1 min 57 sec	1 min 15 sec
VMS	192K	90%/10%	4 min 23 sec	14 sec
	175K	48%/52%	5 min	2 min 35 sec
	68K	0%/100%	1 min 47 sec	1 min 15 sec

Software Subscription customers should be receiving their updated copy. If you have not received your copy, please contact your sales representative.

Marilyn Hanson SDP Marketing

SOFTWARE VERSION LIST (Dec. 1985)

The following is the latest releases of SDP software versions. If subscribers have not received their updates through Software Subscription Service (SSS) updates, please contact your local sales representative. Further information regarding SSS is covered in the PRODUCT PERFORMANCE SECTION.

PRODUCT NUMBER	VERSION	DATE
ACEDIT 1A	03.00-00	06-07-83
ACEDIT 1Z	03.00-00	06-07-83
ASM1750 1A	02.02-21	12-20-85
ASM1750 1E *3	02.00-15	09-18-84
ASM1750 1F	Y03.00-18	08-13-85
ASM1750 1M *2	Y03.00-18	10-16-85
ASM1750 1N	Y03.00-18	10-16-85
ASM1802 1A	01.01-56	11-02-82

ASM1802 1Z	4.0A	12-05-82
ASM3870	4.0B	12-05-82
ASM6500	4.0A	10-10-80
ASM6800 1A	01.03-18	02-15-82
ASM6800 1Z	4.0A	11-18-82
ASM68000	4.0	10-17-84
ASM6809 1A	02.01-21	12-13-85
ASM6809 1B *2	01.00-06	03-28-84
ASM6809 1C	02.00-16	03-05-85
ASM6809 1D *2	01.00-06	08-08-83
ASM6809 1E *2	02.01-09	06-28-84
ASM6809 1F	Y03.00-16	06-21-85
ASM6809 1G *2	Y01.00-00	11-28-83
ASM6809 1H *2	01.00-06	01-28-85
ASM6809 1J	01.00-06	01-28-85
ASM6809 1M *2	Y03.00-16	10-16-85
ASM6809 1N	Y03.00-16	10-16-85
ASM6809 1Z	01.09-65	01-04-83
ASM6809	4.0	01-28-81
ASM68K 1A	02.03-21	12-13-85
ASM68K 1B *2	02.04-16	01-10-85
ASM68K 1C	02.04-16	01-10-85
ASM68K 1D	01.00-06	08-11-83
ASM68K 1E *2	02.03-09	06-28-84
ASM68K 1F	Y03.04-16	06-21-85
ASM68K 1G *2	Y01.00-00	11-28-83
ASM68K 1H *2	02.04-16	01-28-85
ASM68K 1J	02.04-16	01-28-85
ASM68K 1M *2	Y03.04-16	10-16-85
ASM68K 1N	Y03.04-16	10-16-85
ASM68K 1Z	01.15-65	01-07-83
ASM7720A 1A	01.00-16	03-04-85
ASM7809 1A	01.00-16	07-19-84
ASM7811 1A	01.00-16	07-05-84
ASM78C06 1A	01.00-16	04-13-84
ASM8048 1A	02.04-08	05-04-83
ASM8048 1Z	4.1A	12-05-82
ASM8051 1A	01.08-56	12-05-82
ASM8051 1Z	01.09-61	12-05-82
ASM8085 1A	02.00-21	12-13-85
ASM8085 1B *2	02.00-16	03-04-85
ASM8085 1C	02.00-16	03-05-85
ASM8085 1D *2	01.00-06	07-21-83
ASM8085 1E *2	02.01-09	07-10-84
ASM8085 1F	Y03.00-16	06-21-85
ASM8085 1G *2	Y01.00-00	11-28-83
ASM8085 1H *2	01.00-06	01-28-85
ASM8085 1J	01.00-06	01-28-85
ASM8085 1M *2	Y03.00-16	10-16-85
ASM8085 1N	Y03.00-16	10-16-85
ASM8085 1Z	4.0A	12-05-82
ASM8086 1A	02.09-21	12-13-85
ASM8086 1B *2	02.07-16	01-10-85
ASM8086 1C	02.07-16	01-10-85

ASM8086 1D *2	01.01-08	12-22-83
ASM8086 1E *3	02.07-09	07-02-84
ASM8086 1F	Y03.07-16	06-21-85
ASM8086 1G *3	Y01.00-00	11-28-83
ASM8086 1H *2	02.07-16	01-28-85
ASM8086 1J	02.07-16	01-28-85
ASM8086 1M *2	Y03.07-16	10-16-85
ASM8086 1N	Y03.07-16	10-16-85
ASM8086 1Z	01.18-41	07-08-82
ASM8086/88	4.1B	01-30-81
ASM9900 1A	02.04-21	12-13-85
ASM9900 1Z	01.04-35	06-22-82
ASM9900	4.0A	12-05-82
ASMZ80 1A	03.00-21	12-13-85
ASMZ80 1B *2	02.00-16	02-14-85
ASMZ80 1C	02.00-16	02-20-85
ASMZ80 1D *2	01.00-06	07-21-83
ASMZ80 1E *2	02.01-09	06-28-84
ASMZ80 1F	Y03.00-16	06-21-85
ASMZ80 1G	Y01.00-00	11-28-83
ASMZ80 1H *2	01.00-06	01-28-85
ASMZ80 1J	01.00-06	01-28-85
ASMZ80 1M *2	Y03.00-16	10-16-85
ASMZ80 1N	Y03.00-16	10-16-85
ASMZ80 1Z	4.0A	12-05-82
ASMZ8000	4.0	09-17-80
ASMZ8K 1A	02.04-21	12-13-85
ASMZ8K 1Z	01.01-01	12-07-81
CCC68K 1A	02.09-00	02-25-85
CCC68K 1B *4	02.09-00	02-14-85
CCC68K 1B VOL 2 *2	01.00-00	04-18-84
CCC68K 1C	02.09-00	02-18-85
CCC68K 1D VOL 2 *2	01.00-01	07-19-83
CCC68K 1E *6	02.06-00	01-28-85
CCC68K 1F	01.01-00	09-04-84
CCC68K 1F	02.06-00	01-28-85
CCC68K 1F	Y03.06-00	07-22-85
CCC68K 1G VOL 2 *2	01.01-00	11-16-83
CCC68K 1H *4	02.05-00	01-28-85
CCC68K 1J	02.05-00	01-28-85
CCC68K 1M *4	Y03.06-00	10-16-85
CCC68K 1N	Y03.06-00	10-16-85
CCC8086 1C	01.03-00	09-27-85
CCC8086 1E *6	01.03-00	09-20-85
CCC8086 1F	01.03-00	09-13-85
CCC8086 1M *4	01.03-00	10-16-85
CCC8086 1N	01.03-00	10-16-85
CDB68K 1A	01.08-00	03-05-85
CDB68K 1B	01.09-00	02-25-85
CDB68K 1C	01.09-00	02-25-85
CDB68K 1E	01.10-00	03-14-85
CDB68K 1F	Y03.06-00	07-22-85
CDB68K 1H	01.04-00	02-25-85
CDB68K 1J	01.04-00	02-25-85

CDB68K 1M	Y03.06-00	10-16-85
CDB68K 1N	Y03.06-00	10-16-85
CLDEDIT 1A	01.08-00	12-18-85
CLDEDIT 1B *2	01.04-00	04-08-85
CLDEDIT 1C	01.04-00	04-08-85
CLDEDIT 1E *2	02.01-00	08-09-84
CLDEDIT 1F	Y03.01-00	07-22-85
CLDEDIT 1H	01.03-00	04-08-84
CLDEDIT 1J	01.03-00	04-08-84
CLDEDIT 1M	Y03.01-00	10-16-85
CLDEDIT 1N	Y03.01-00	10-16-85
COLORKY 1B *2	1.2	03-26-84
COLORKY 1C	03.00-01	07-22-85
COLORKY 1E *3	02.00-01	08-16-84
COLORKY 1F	Y03.00-01	07-08-85
COLORKY 1J	1.2	07-22-85
COLORKY 1M *3	Y03.00-01	10-16-85
COLORKY 1N	Y03.00-01	10-16-85
DDL 1A	2.0	08-09-84
DDL 1B *2	2.0	08-09-84
DDL 1C	2.0	08-09-84
DOS/50 VERSION 2 O/S	2.1A	11-16-82
ICOM40 1B *2	2.0	01-10-85
ICOM40 1C	02.00-00	01-10-85
ICOM40 1D *2	1.0	08-02-83
ICOM40 1E	02.02-00	02-11-85
ICOM40 1F	04.04-00	05-31-85
ICOM40 1G	1.0	11-16-84
ICOM40 1H *2	1.0	12-20-84
ICOM40 1J	1.0	01-09-85
ICOM40 1Y	01.03-00	05-03-85
ICOM40 3B *3	2.0	12-20-84
ICOM40 3C	02.00-00	01-09-85
ICOM40 3D *2	1.0	08-02-83
ICOM40 3E	1.0	11-04-83
ICOM40 3F	1.0	11-04-83
ICOM40 3G	1.0	11-16-83
ICOM40A 1B *2	02.01-00	08-05-85
ICOM40A 1C	02.01-00	08-05-85
ICOM40A 1E	02.02-00	08-01-85
ICOM40A 1F	04.05-02	07-26-85
ICOM40A 1J	1.0	08-01-85
ICOM40A 1M	04.05-02	10-16-85
ICOM40A 1N	04.05-02	10-16-85
ICOM40A 1Y	01.03-00	08-01-85
ICOMSRC 1C	02.01-00	08-05-85
LAS LINKER/LIBGEN	02.08-00	09-07-82
MDL6800 1Z	2.1A	12-05-82
MDL8085 1A	3.0	02-06-84
MDL8085 1Z	2.1A	12-05-82
MDLZ80 1A	3.0	02-17-84
MDP-FTE FIXED DISC DIAGS	1.00	04-23-83
MICROLAB 1 3870/3872	1.9	01-30-81
MICROLAB 1 MCS48	1.1	01-30-81

MICROLAB MONITOR 6801/120	1.0	10-05-82
PAS68K 1A	03.06-00	12-20-85
PAS68K 1B *4	01.02-00	02-16-84
PAS68K 1C	03.02-00	06-19-85
PAS68K 1D *5	01.02-00	02-16-84
PAS68K 1E *5	02.03-00	08-30-84
PAS68K 1F	Y03.02-00	07-18-85
PAS68K 1H *4	01.02-00	01-28-85
PAS68K 1J	01.02-00	01-28-85
PAS68K 1M *4	Y03.02-00	10-16-85
PAS68K 1N	Y03.02-00	10-16-85
PAS8080 1Z	4.03	05-30-84
PAS8086 1A *2	02.10-01	08-02-85
PAS8086 1C	02.13-00	02-11-85
PAS8086 1F	Y03.11-00	08-07-85
PAS8086 1J	02.13-00	02-11-85
PAS8086 1M *6	Y03.11-00	10-16-85
PAS8086 1N	Y03.11-00	10-16-85
PAS8086 1Z	01.01-07	10-12-82
PDB68K 1A	03.03-00	12-20-85
PDB68K 1B	02.01-00	03-14-84
PDB68K 1C	02.07-00	07-18-85
PDB68K 1D	02.01-00	01-12-84
PDB68K 1E	02.02-00	04-22-85
PDB68K 1F	Y03.05-00	07-16-85
PDB68K 1H	02.01-00	01-28-85
PDB68K 1J	02.01-00	01-28-85
PDB68K 1M *2	Y03.05-00	10-16-85
PDB68K 1N	Y03.05-00	10-16-85
PDB8086 1A	02.00-05	08-02-85
PDB8086 1C	02.01-02	08-16-85
PDB8086 1F	Y01.01-00	10-25-85
PDB8086 1M *2	Y01.01-00	10-25-85
PDB8086 1N	Y01.01-00	10-25-85
PDBZ8K 1A	01.05-00	12-17-82
PLDEDIT 1A	02.04-00	05-18-84
PLDEDIT 1B	01.05-00	02-11-85
PLDEDIT 1C	01.05-00	02-11-85
PLDEDIT 1D	01.02-00	09-13-83
PLDEDIT 1E *2	02.00-00	08-09-84
PLDEDIT 1F	Y03.00-00	07-16-85
PLDEDIT 1G	01.03-00	11-16-83
PLDEDIT 1H	01.05-00	01-28-85
PLDEDIT 1J	01.05-00	01-28-85
PLDEDIT 1M	Y03.00-00	10-16-85
PLDEDIT 1N	Y03.00-00	10-16-85
PVRX68K 1A	03.00-00	05-03-85
PVRX68K 1C	03.00-00	08-09-85
PVRX68K 1F	03.00-00	08-13-85
STRUCTA 1A	01.04	08-16-84
STRUCTA 1B *2	01.00	08-09-84
STRUCTA 1C	01.00	08-09-84
STRUCTA 1F	01.02-00	12-13-85
STRUCTA 1M *3	01.02-00	12-13-85

STRUCTA 1N	01.02-00	12-13-85
TNIX 2.0 TO 2.1 UPDATE *3	2.10	04-26-84
TNIX OPERATING SYSTEM *5	2.1	05-01-84
TNIX REVISION	2.1B	03-29-85
TTA/HLP 1A	1.0	01-06-84
8086 SBC	1.0	07-22-81
8086/8088 DIAGS	1.4	04-20-82
8300E01 8080A EMULATOR	2.1	08-13-82
8300E02 6800/6802 EMULATOR	2.1	08-13-82
8300E04 Z80A EMULATOR	2.1	07-18-83
8300E05 TMS9900 EMULATOR	1.0	12-10-80
8300E06 8085A EMULATOR	2.2	02-06-84
8300E07 3870/72/F8 EMULATOR	2.1	12-15-82
8300E09 1802 EMULATOR	2.1	10-12-82
8300E10 8048/21/41/22 EMU	2.0	06-15-82
8300E14 6500/01 EMULATOR	1.1	10-09-81
8300E33 9900/89 EMULATOR	2.0	08-05-82
8300E38/40 80186/88 EMU	2.1	10-22-84
8300P15/18 8086/88/87 EMU	1.15	11-01-82
8300P16 8088 EMULATOR	1.10	05-09-82
8300P20/22 Z8001/2 EMU	1.9	03-30-82
8300P26 68000 EMULATOR	1.20	07-11-83
8300P28 6809 EMULATOR	2.0	03-30-82
8300P29/30 6801/120 EMU	1.1	11-01-82
8300P37 NSC-800 EMULATOR	1.1	07-31-85
8300P38/39/40 68000/08/10 EMU	2.0	08-02-83
8300S01 RT11/50 O.S.	1.00	09-07-82
8540/50/60/U04 INTEL COMM.	1.00	09-17-82
8550 GUIDE	1.0	06-15-82
8550 SYSTEM DIAGS	3.0	08-16-84
8550F20 EXTENEDED HEX	1.0	12-10-82
8550F30 PROM PROGRAMER	2.1	03-30-82
8550S01 RTDOS UTILITY	1.00	08-20-82
8550U04 EHEX COMMAND	1.0	09-17-82
8560 LINKER/LSTR/LIBGEN	02.02-00	06-22-82
8560/61 SYSTEM DIAGS	01.03-01	07-11-84
8560U01 TEXT PROCESSING	2.0	06-16-83
8560U02 NATIVE PROGRAMMING	2.0	06-16-83
8560U03 AUX UTILITIES *2	2.0	06-16-83
8560U04 EHEX COMMAND	3.4	09-17-82
8560U05 UNICOM	1.01	11-21-83
8086 SBC	1.0	07-22-81
8086/8088 DIAGS	1.4	04-20-82
8300E01 8080A EMULATOR	2.1	08-13-82
8300E02 6800/6802 EMULATOR	2.1	08-13-82
8300E04 Z80A EMULATOR	2.1	07-18-83
8300E05 TMS9900 EMULATOR	1.0	12-10-80
8300E06 8085A EMULATOR	2.2	02-06-84
8300E07 3870/72/F8 EMULATOR	2.1	12-15-82
8300E09 1802 EMULATOR	2.1	10-12-82
8300E10 8048/21/41/22 EMU	2.0	06-15-82
8300E14 6500/01 EMULATOR	1.1	10-09-81
8300E33 9900/89 EMULATOR	2.0	08-05-82
8300E38/40 80186/88 EMU	2.1	10-22-84

8300P15/18 8086/88/87 EMU	1.15	11-01-82
8300P16 8088 EMULATOR	1.10	05-09-82
8300P20/22 Z8001/2 EMU	1.9	03-30-82
8300P26 68000 EMULATOR	1.20	07-11-83
8300P28 6809 EMULATOR	2.0	03-30-82
8300P29/30 6801/120 EMU	1.1	11-01-82
8300P37 NSC-800 EMULATOR	1.1	07-31-85
8300P38/39/40 68000/08/10 EMU	2.0	08-02-83
8300S01 RT11/50 O.S.	1.00	09-07-82
8540/50/60/U04 INTEL COMM.	1.00	09-17-82
8550 GUIDE	1.0	06-15-82
8550 SYSTEM DIAGS	3.0	08-16-84
8550F20 EXTENEDED HEX	1.0	12-10-82
8550F30 PROM PROGRAMER	2.1	03-30-82
8550S01 RTDOS UTILITY	1.00	08-20-82
8550U04 EHEX COMMAND	1.0	09-17-82
8560 LINKER/LSTR/LIBGEN	02.02-00	06-22-82
8560/61 SYSTEM DIAGS	01.03-01	07-11-84
8560U01 TEXT PROCESSING	2.0	06-16-83
8560U02 NATIVE PROGRAMMING	2.0	06-16-83
8560U03 AUX UTILITIES *2	2.0	06-16-83
8560U04 EHEX COMMAND	3.4	09-17-82
8560U05 UNICOM	1.01	11-21-83

Myrl Kwiatkowski SDP Marketing

USER NOTES SECTION

DIFF.LST TEXT FILE INTRODUCED ON NEW VERSION RELEASES

Effective immediately, all SDP modified software products (Version 2, 3, etc.) will contain a text file called DIFF.LST on the installation media.

The purpose of this file is to list the differences between this current release and the previous version. This file will provide an answer to the question: "Why should I install/use this new version? What did you change?" THE EXISTENCE OF THIS FILE WILL NOT BE DOCUMENTED, but the file is accessible for AEs and customers alike who elect to copy it and read it.

The SDP PLANDS Design Team is releasing the first update to include this file; the DIFF.LST file from Version 3 PLANDS/TNIX is reprinted in the next article.

Byron Lunz Customer Marketing Manager

PRODUCT CHANGES FOR VERSION 3 TNIX 68000/68010 PASCAL

The TNIX 68000/68010 PASCAL Version 3 update has been produced to improve the compiler's capacity and performance, fix any problems reported to SDP Engineering before October 2, 1985 relating to this product, and to bring an enhanced ICS package to TNIX. This update requires that you have an 856X system with a PDP 11/73 processor and a minimum memory configuration of 512K.

I. Compiler Capacity and Performance

The Version 3 compiler's capacity has been increased and its performance has been improved. There are significant improvements in the number of symbols permitted per module and in compiler speed (lines per minute). Specific benchmarks depend on the combinations of code which you use.

II. Compiler Changes

1. File Format for Internal Temporary Files

The file name format for internal temporary files created by the rewrite procedure is now zXXX.tmp, where XXX is a three-digit number.

2. Program Termination

The compiler no longer calls EXITQQ to exit the program. Instead, control is returned to ICS generated code which determines the proper exit method. See Section IV.5 of this document for further information.

3. Dispose Fix for Separate Address Spaces

The dispose procedure now functions properly when using separate address spaces.

4. Port Variable Addresses Now Use Full 68000/68010 Address Space

Port variable addresses are now able to use the full 68000/68010 address space. They were previously limited to 0-7FFF hex.

5. Compiler Now Exits Gracefully on Long Lines

The compiler produces a more useful error message when a line exceeds the maximum of 132 characters.

6. Fix for Pack and Unpack

The pack and unpack procedures in the library rts.lib now function properly.

7. Packed Data Structure Code Corrected

All known problems regarding packed data structures have been fixed.

8. Sets Code Corrected

A problem in the procedure which assigns values to sets has been corrected. Another problem which prevented the set difference and set union operations from being re-entrant was fixed.

9. Module Names Now 14 Characters

The maximum size for module names has been increased from 8 to 14 characters. This allows the section type to be prepended to the name and still fit into the 16-character LAS format.

10. Three-Byte Variables Now Handled Properly

The Version 3 compiler was fixed to generate the proper size moves for three-byte variables.

11. Correct Section Length Now Entered in Object Code

The compiler now enters the actual size of the section into the object code instead of a value one greater than the section size. This will correct linker errors which occurred in some instances.

12. Nested External Declarations Now Flagged by Parser

The Version 3 compiler is more strict in enforcing some of the rules of PASCAL. For example, external routines which are not declared globally will be flagged as a syntax error by the parser.

13. Multiple Origin Variables May Now be Accessed

The Version 2 compiler was not able to access more than one variable with the origin attribute. This restriction has been removed in the Version 3 compiler.

14. Correct Code Now Generated for MOD Function

A bug was fixed which caused the MOD function in the library rts.lib to work improperly in many cases.

15. Tag Checking List Now Disposed Correctly for all Structures

A problem (PPR 5019) regarding the disposal of the tag checking list was fixed.

16. Compiler Aborts Properly on Nested Externs

The Version 2 compiler aborted abruptly when trying to compile a module with nested external procedure having more than two parameters. The Version 3 compiler generates the appropriate error messages.

III. ICS Enhancements for Version 3

1. New Command Line Option: -d[#]

This option generates an ICS configuration for PDB. If the optional number is specified, the debug trap vector corresponding to that number is set up for PDB. The default trap vector is 15.

This option replaces the PASCAL_DEBUG[TRAP#n] option on the PRIVILEGE_STATE directive.

2. New Command Line Option: -m[#]

This option provides full support for emulation modes 1 and 2. The system command file is generated for the desired mode based on your use of this option. If you do not specify -m, specify -m without a number, or specify -m0, a mode 0 configuration is created. In this case, all memory is mapped to your 8540. If you specify -m1, a mode 1 configuration is created. In this case, memory is mapped based on your use of the PROTOTYPE option described below. If you specify -m2, a mode 2 configuration is created. In this case, all memory is mapped to your prototype.

3. New Command Line Option: -p

This option parses your ICS source file without generating any output files.

4. New Command Line Option: [objfiles ...]

You can now specify object modules or object libraries on the ICS invocation line. This is useful if you want to test a module or library before adding it to your ICS source file. If you are specifying both modules and libraries, it is best to list the libraries after the modules.

5. New Directive: PORT_MEMORY

This directive allows you to allocate memory that is known to your 8540 or to your prototype, but not to the linker.

6. New Directive: USER_DEFINED_SYMBOL

This directive allows you to define global linker symbols in your ICS source file. Each symbol that is defined is added as a -D option in the linker command file.

7. System Command File Invocation Line

The system command file (.ie file) now supports the three new invocation line options. The -e option will assemble the .ia file, link using the .ic file, set up the emulator, and download the load module if you are not configured for PDB. The -k option performs the same as -e except that the .ia file is not assembled. The -d option sets up a configuration for low level debugging. It will reassemble the .ia file and re-link if necessary. You may also specify object modules and object libraries on the invocation line.

8. PROTOTYPE Option on Memory Directives

This option has been added to the INSTRUCTIONS_ROM, CONSTANTS_ROM, GLOBAL_VAR_RAM, HEAP_STACK_RAM, PORT_MEMORY, RESET_MEMORY, and INTERRUPT_CONFIGURATION directives. If you specify the PROTOTYPE option, the associated memory ranges are mapped to your prototype in emulation mode 1.

9. Section Names Permitted on Selected Memory Directives

You may now specify section names on the INSTRUCTIONS_ROM, CONSTANTS_ROM, and GLOBAL_VAR_RAM directives. This enables you to locate specific sections of code at specific memory locations from within ICS.

10. FAST_RTS Option on FLOATING_POINT_SUPPORT Directive

This option enables you to access the 68000 PASCAL fast floating point library (fpfastr4) without specifying it as a user floating point library.

11. ICSP Provides Option for Invoking ICS

You can now invoke ICS with the options of your choice from within ICSP.

12. ICSP Provides an Escape to Shell Option

You can now escape to the TNIX shell from within ICSP if you need to execute TNIX commands.

IV. ICS Differences Between Version 3 and Version 2

1. Different ICS Invocation Line

The Version 2 invocation was

```
ics [-eklsv] file.is
```

while the Version 3 invocation is

```
ics [-d[#]eklm[#]pv] file.is [objfiles ...]
```

2. ICS -s Option No Longer Supported

The -s option on the ICS invocation line is no longer accepted. The assembly language source file (.ia file) is always generated unless you specify the -p option.

3. Different System Command File Invocation Line

The Version 2 invocation was

```
file.ie
```

while the Version 3 invocation is

```
file.ie [-dek] [objfiles ...]
```

4. Different Syntax for EMULATOR_ADDRESS_SPACES Directive

The syntax for the EMULATOR_ADDRESS_SPACES directive has been updated to match that of the 68000 C ICS. The old format was

```
                  { CONTIGUOUS                    }
EMULATOR_ADDRESS_SPACES { SEPARATE [, P&D_CONTIGUOUS ] }
```

while the new format is

```
                                  { S&U_SEPARATE                    }
EMULATOR_ADDRESS_SPACES           { S&U_CONTIGUOUS   [ , P&D_SEPARATE ] }
```

The new form allows the S&U_CONTIGUOUS,P&D_SEPARATE memory configuration to be supported. The Version 2 form prohibited this combination.

5. Different Program Termination Process

The Version 3 ICS is responsible for the termination of your application. After calling MAINQQ (the main PASCAL program entry point), ICS terminates in one of three ways:

- If service calls are enabled and FILE_SUPPORT is not NONE, ICS calls the version of EXITQQ in the POSI library. This version of EXITQQ halts your application.
- If service calls are enabled and FILE_SUPPORT is NONE, ICS performs an SVC abort.
- If service calls are disabled and you are using your own version of POSI, ICS calls the version of EXITQQ in your POSI library.
- If service calls are disabled and FILE_SUPPORT is NONE, ICS calls the version of EXITQQ in the NOIO library. This version of EXITQQ terminates your application by entering a planned infinite loop. This is because there is no means of establishing communication with your terminal.

These exit methods also imply that you are responsible for terminating the application if you do not use the standard initialization code. Under Version 2, the compiler was responsible for program termination and an SVC abort was not performed if service calls were enabled and FILE_SUPPORT was NONE. Additionally, the Version 2 ICS performed a jump to MAINQQ instead of a call.

6. System Command File Generated for PDB Does Not Invoke PDB

The system command file no longer invokes PDB automatically if the -d option is selected on the ICS invocation line. This prevents a fixed invocation of PDB from being enforced and it prevents a host system error from occurring if PDB is not installed. PDB should be invoked after the emulator has been set up using one of the following schemes:

```
$ ics -ed file.is
$ pdb file.lo
```

7. Configuration for PDB Handled on ICS Invocation Line

The PASCAL_DEBUG[TRAP#n] option of the PRIVILEGE_STATE directive is no longer supported. You should generate your ICS files for PDB by using the -d option on the ICS invocation line.

8. Dash ("-") is the Only Valid Range Separator

The string ".." is no longer accepted as a valid range separator for the memory directives and the interrupt specifications on the INTERRUPT_PROCEDURE directive. The dash ("-") is accepted as before and is the only valid range separator.

9. Changes to ICSP

The secondary menus and questions for ICSP have been updated to properly reflect the Version 3 ICS.

10. Error Messages

The ICS and ICSP error messages have been restructured to reflect the Version 3 product.

Byron Lunz Customer Marketing Manager

VMS SA TOOLS

If you have a laser printer that understands Tek graphics, the following procedure will help you get hard copy data from your SA Data Flow Diagrams (DFD).

```
SA/SHOW/TERM=4014/OUTPUT=DFD.4014  0.DFD
CONVERT/PAD/FDL=DFD.FDL  DFD.4014  OUT.4014
```

The saved DFD output file must be converted from variable length record to fixed length record format and the terminal settings must be changed to eliminate VMS altering the output to the terminal.

The DFD.FDL file must first be created using the FDL editor (or another text editor). The minimum contents of the FDL file are shown here:

```
RECORD
        BLOCK_SPAN
        CARRIAGE_CONTROL
        FORMAT
        SIZE
```

The terminal settings are changed with the commands:

```
SET TERM/NOWRAP
SET TERM/FORM
```

This method could also be used to convert the output from the SA/SHOW command for input to a laser printer.

The SA/SHOW command option to redirect output to a file was intended to save copies of the DD for printing since the DD is normally in a non-text file format. The output file from the SA/SHOW command is a variable length record file with an assumed CR/LF at the end of each record.

The DFD graphics image is a continuous byte stream intended for the terminal. There is no notion of records. Interspersing CR/LF will cause the display to be garbled when sent to the terminal from the redirected output file.

Rainer Wieland Senior Software Engineer

SEPARATE I&D ASSEMBLERS AVAILABLE FOR THE 8560

SDP is currently providing several separate I&D assemblers which are built specifically for the 11/73. The installation disc for these assemblers contains two versions: Common I&D for LSI 11/23, and Separate I&D for LSI 11/73. Both versions can exist on the system, the ASM invocation will check the CPU and select the corresponding program.

- Separate I&D ASM can handle about twice as many symbols as the Common I&D non-virtual assembler, which means more modules can be assembled without using the virtual.
- The virtual 11/73 ASM can keep more symbols in memory at one time, so disc accesses are reduced when virtual is required.
- Both separate I&D ASM (virtual and non-virtual) are non-overlayed so they execute faster than the common I&D versions (which require overlays).
- The separate I&D assemblers handle the same source and generate the same object code as the Common I&D assemblers.

The current separate I&D assemblers available are:

ASM68K	V02.03-21
ASM8085	V02.00-21
ASMZ80	V03.00-21
ASM9900	V02.04-21
ASM6809	V02.01-21
ASMZ8K	V02.04-21
ASM8086	V02.09-21
ASM1750	V02.02-21

Software Subscription customers will be receiving their updates. If you have not received your copy by the end of January, contact your sales representative.

Stephen Wood Software Applications Manager

VMS COMMAND TO PUT LINE NUMBERS IN A FILE

The following VMS command can be used to generate a file listing with line numbers:

```
sear/number "" file/output=file.lst
```

Stephen Wood SDP Applications Manager

MIL-STD 1750A SECTION

MIL-STD-1750A/LAS Linker Incompatibility

Some MIL-STD-1750A implementations and SDP's 1750A-1 emulator support what the MIL-STD calls the "Expanded Memory Option". (This option provides the capability to address more than 64K Words of memory). Linking software modules to take advantage of this option is a unique and challenging problem, and different from the problems which the LAS linker was designed to solve.

SDP's linker is not practical for linking into expanded systems. It is, however, capable of linking non-expanded systems. This limitation is stated in the 1750A Assembler Manual (070-5111-00) on pages 9A-1 and 9A-13. These pages state that the limitation is inherent in the assembler; it would be more accurate to say that this is a limitation of the assembler/linker package.

SDP is currently studying the requirements of a linker for supporting expanded memory systems. Users requiring expanded systems in the short term must avail themselves of software products from other vendors for that purpose. Since we know of no expanded linker currently available which produces Extended Tekhex or LAS object code, the user will have to provide a conversion to those formats to download to the emulator.

Al Marshall SDP Engineering

8540 ROMPATCH 72

In User Group News Volume 4 Issue 2 in the Product Performance Section Rompatches 73 and 74 was published omitting 72.. Without Rompatch 72 a user is unable to install Rompatches 73 and 74. Up to this time Rompatch 72 had not been published. The intent of this article is to provide Rompatch 72 for those users that wish to install Rompatches 73 and 74.

Rompatches 73 and 74 correct a breakpoint display problem with the 1750A-1 Emulator. After a breakpoint the Processor Status (PS) will always show a value of 0, no matter what the true value of the PS. Only the displayed value of the PS is incorrect. The actual value that would be restored during a break and continue is correct.

Rompatches 72 thru 74 are as follows:

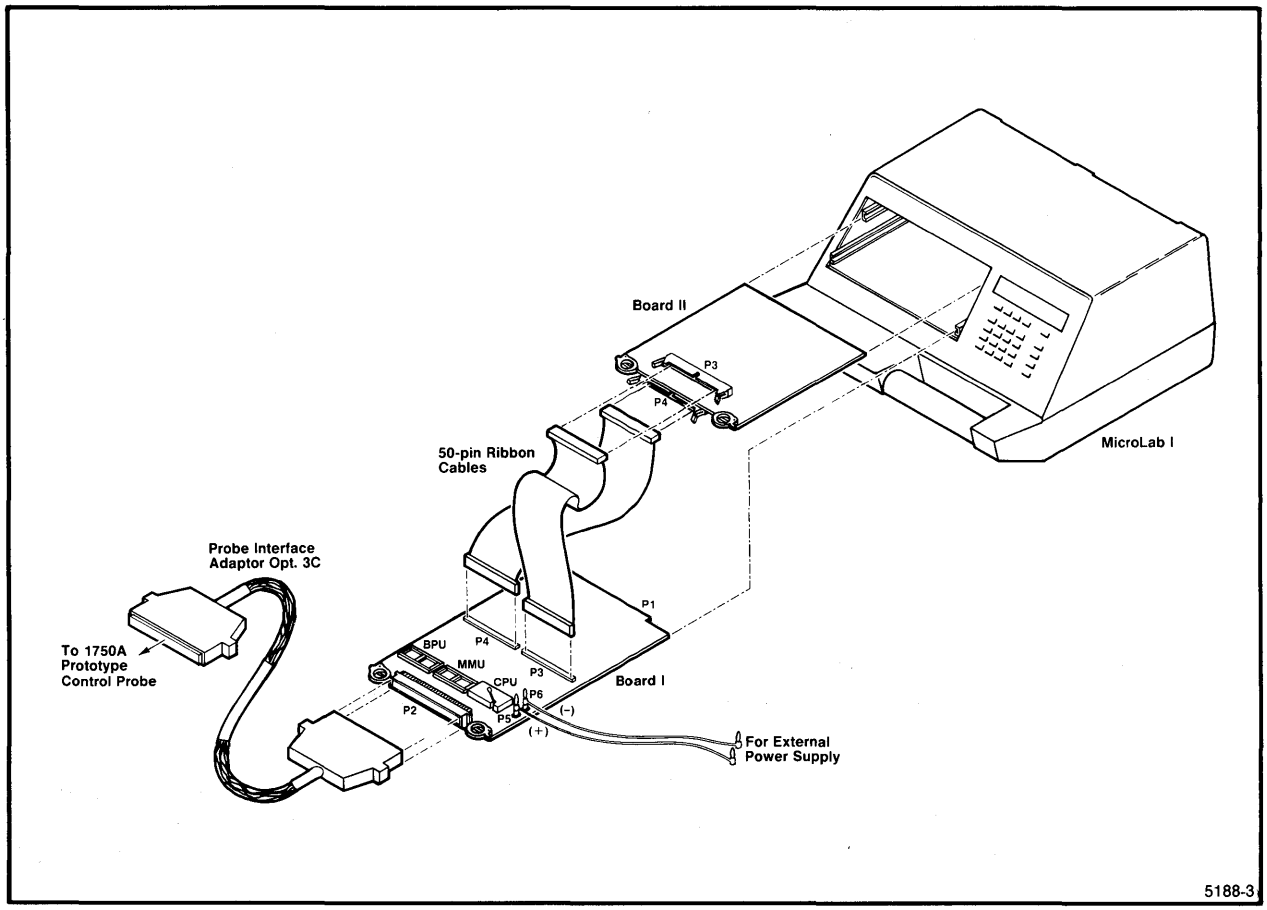
rompatch 08b50 72 0d70 /334400/1 3f6b89
rompatch 02d97 73 0ec /256900/6 3f5ff8
rompatch 0c29 74 6f8 /256900/6 0401cc1ada1f5a23

Kevin King Application Engineer

F9450 MICRO-LAB AVAILABLE

The F9450 Personality Cards are now available for the Micro-Lab system. The Micro-Lab is a portable package that is generally used for field service and troubleshooting applications related to the V1750A. The Micro-Lab can also be used as a software execution environment, although it has limited on-board memory and requires an external power supply. By using the V1750A "mapping" function, the larger emulator memory* can be used in place of the Micro-Lab memory.

* Up to 128 KW of emulator memory is available with the 8540 mainframe and up to 393 KW with the 8540A.



The F9540 Personality Cards can be used with either the F9450 In-Circuit Probe (010-6604-00), or the Euro-Card PIA (010-0438-00). The system can be configured to support expanded or non-expanded memory. An F9450 CPU and F9451 MMU are included with the card set. The F9452 BPU is not included with the card set, but a socket is available and it can be added. The on-board memory system incorporates 8 KW of EPROM that contains a built-in diagnostic operating system, 8 KW of RAM and 1 KB of fast RAM. The fast RAM is used to test full speed, no-wait state operation at a 20 MHz clock frequency.

Because of the high current requirements of the F9450 CPU, an external +5 VDC supply is required. A Tektronix recommended supply, or equivalent, can be used.

Here are the part numbers for the F9450 Micro-Lab system and it's manuals.

Ordering Nomenclature	Description
067-0892-00	Micro-Lab Mainframe
-01	Power Option A1
-02	Power Option A2
-03	Power Option A3
-04	Power Option A4
070-2827-01	Manual, Micro-Lab Mainframe
018-0216-00	F9450 Personality Cards (includes 12 MHz F9450 CPU, F9451 MMU & Euro-Card PIA)
070-5188-00	Manual, F9450 Personality Card
067-1235-00	+5 VDC Supply

Bill Bevan Military Program Manager

V1750A NOW INCLUDES 8540A MAINFRAME

The V1750A Software Integration Unit now includes the NEW 8540A mainframe. All V1750A units shipped after October 1, 1985 will include the 8540A in place of the 8540. That's good news for new and existing customers.

New Customers

For new customers the good news is more value at no charge. The new V1750A offers the following:

- Increased Program Memory Capacity.

The V1750A can now support up to 393 KW (786 KB) of program memory. That's a 3X increase

over the previous 128 KW (256 KB) maximum. The standard V1750A will still include 64 KW (128 KB) of program memory.

- Lower Price for Bundled Program Memory.

- Extended Hex Functions, Standard.

  The V1750A now includes the Extended TekHex, Motorola and Intel Hex download functions in the base system.

- Lighter Weight and Higher Reliability

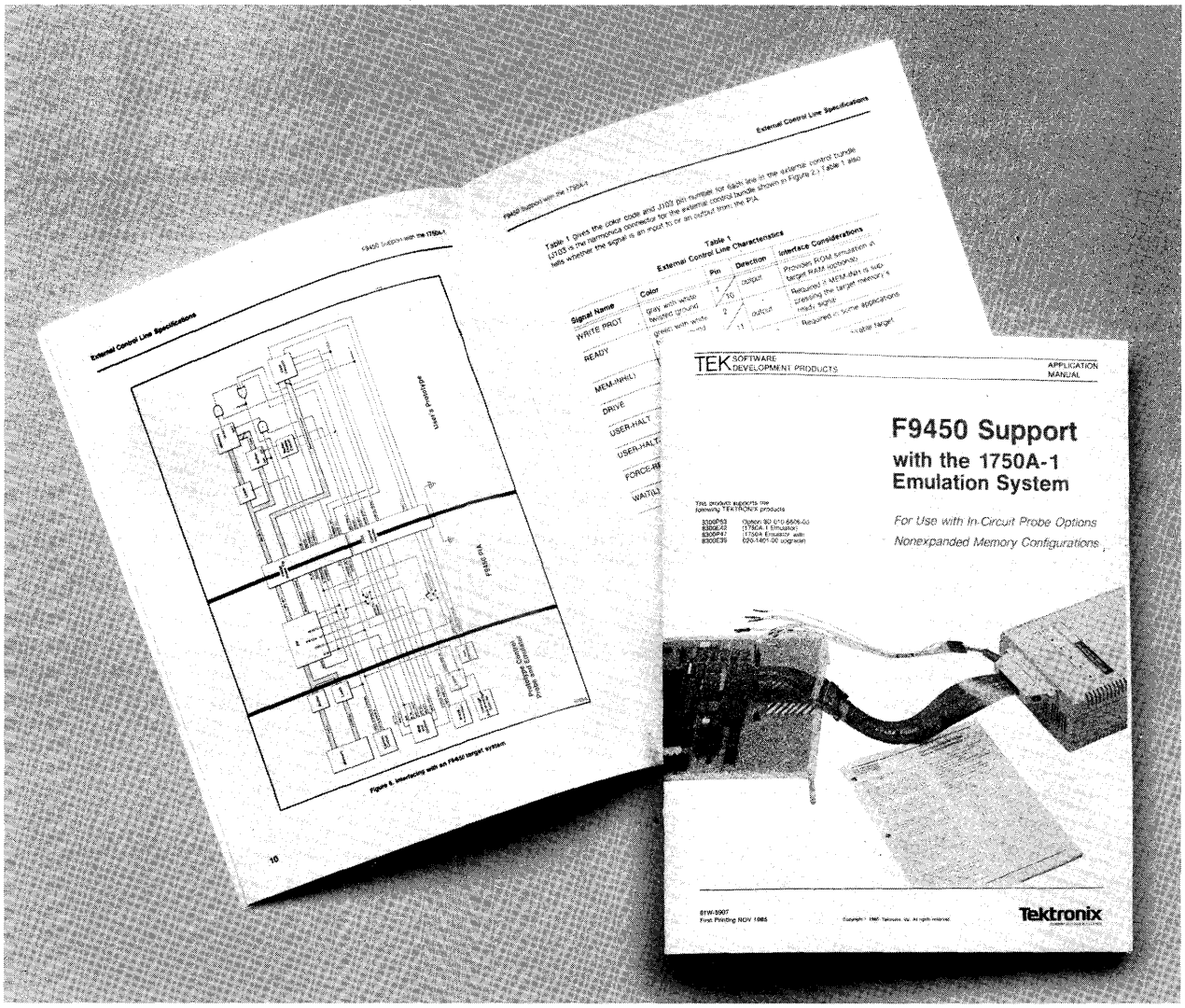
  The 8540A incorporates a new high-efficiency switching power supply that replaces the older transformer version. The supply eliminates about a 100 watts of "dead power" that was dissipated as heat. This will significantly improve the system's reliability and resulting MTBF. The new supply also reduces the weight of the V1750A by about 20 pounds.

Existing Customers

The good news for existing customers is that their V1750A can be upgraded to include the 8540A. A package is available that upgrades an existing 8540 mainframe, to an 8540A mainframe; All for about 15% of the original price of the base V1750A system. This package must be installed at a Tektronix field office so the V1750A will need to be moved from the customer site.

Bill Bevan Military Program Manager

F9450 APPLICATION MANUAL



A new Application Manual is available explaining the "how to" of connecting the Tektronix F9450 Probe Interface Adapter (PIA) to a Fairchild F9450-based target system and verifying that connection.

This Application Manual is intended for use with the 1750A-1 Emulator installed in a Tektronix 8540 Integration Unit. Diagrams show how to connect memory control circuitry and detail how the PIA provides the interface between the emulator and the user's prototype. The Application Manual is free upon request from your local Tektronix sales representative. Ask for literature #61W-5907.

Bill Bevan Military Program Manager

THIRD PARTY VENDOR SECTION

TEK/DEC AGREEMENT

We are pleased to announce that SDP and Digital Equipment Corporation have expanded their cooperative marketing agreement contract. In addition to supporting ASSEMBLERS 68000, 8086, 6809, Z80, 8085, PASCAL COMPILER 68000, PASCAL DEBUG 68000, LANGUAGE EDITOR PASCAL, C COMPILER 68000, COLORKEY+ AND ICOM40 COMMUNICATIONS, our new contract will include: CLANDS68K, PLANDS8086, CLANDS8086, ASM1750A, and SA TOOLS to 8540 expanding the entirety of our existing products.

Myrl Kwiatkowski SDP Marketing

PRODUCT PERFORMANCE SECTION

SDP PRODUCT PERFORMANCE REPORTS

The Product Performance Reports (PPR's) which appear on the following pages have been submitted or edited since the last issue of User Group News.

A PPR number appears in parentheses after each PPR title. SDP Software Subscription Service (SSS) customers submitting PPR's will receive preferential treatment. Please reference this number in any correspondence with us or with your local Applications Engineer. We will keep you informed on the progress toward solutions via future reports in this section. We will also try to provide "work-arounds" whenever possible.

Users who are subscribers to the SDP Software Subscription Service will receive all official releases of their products automatically. It is the user's responsibility to keep their end-user address up-to-date to insure proper delivery. Non-SSS subscribers in need of product updates should contact their local Tektronix' sales representative.

You will find a list of currently-shipping versions of all SDP software products published in the PRODUCT INFORMATION SECTION of this issue.

Byron Lunz Customer Marketing Manager

PAS68K PROCESSING OPERATION (4052)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00, VAX VMS (11/750) V3.6

DESCRIPTION:
If the type of a function is not given this aborts the compiler.

ENGINEERING ANALYSIS/RESPONSE:
This has been fixed and compiler now flags this as an error.

COMPILER LIMITATION ON PAS68K (4053)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00 VAX VMS (11/750) V3.6

DESCRIPTION:
If a large number of long include files are specified in a PASCAL source file this aborts the compiler.

ENGINEERING ANALYSIS/RESPONSE:
This is a limitation in the compiler which should be documented. The REAL problem here is that the total input file becomes too large for the compiler to handle--the problem doesn't pertain to the number of included files per se but to the length of the file caused by the includes.

INTEGER ARITHMETIC ON PAS68K (4054)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00 VAX VMS (11/750) V3.6

DESCRIPTION:
A long program containing only integer arithmetic had a) integer overflow and b) an integer multiply followed by integer division which always gave 1 as the answer (no matter what numbers were used). When a statement C := A mod B; was replaced with C := A - ((A div B) * B);, both of the above situations went away.

ENGINEERING ANALYSIS/RESPONSE:
This has been fixed.

ICS COMMAND ON PAS68K (4055)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00 VAX VMS (11/750) V3.6

DESCRIPTION:
If the following VMS commands are typed in:

$ICS "-K PROG.IS"
$@PROG.IE
the error "Unable to redirect STDIN. Run terminating." is given. By adding
the verbose option:
$ICS "-VK PROG.IS"
$@PROG.IE
the error disappears and the program runs.

ENGINEERING ANALYSIS/RESPONSE:
The new ICS base totally removes this condition.

MATHCK RANGE CHECKING ON PAS68K (4056)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00 VAX VMS (11/750) V3.6

DESCRIPTION:
When PASCAL programs are run with range checking enabled, it is still possible to zero the element beyond the last element of an array. This seems to be due to MATHCK. Apparently "(*$mathck-*)" suppresses range checking and "(*$mathck+*)" does not re-enable it.

ENGINEERING ANALYSIS/RESPONSE:
This is operator error. The compiler *should* behave this way. We will review manual to ensure that this is clearly noted.

ERROR IN PASCAL MANUAL (4058)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00 VAX VMS (11/750) V3.6

DESCRIPTION:
Under PASCAL debug it seems that only the first 16 characters of variable names can be specified. There is no mention of this in the PASCAL manual which states that non-global variable names can be of any length.

ENGINEERING ANALYSIS/RESPONSE:
This has been fixed--the debugger and compiler now both truncate to 16 characters.

UPLEVQQ LIBRARY ROUTINE (4064)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX V2.1A, PASCAL 8086 V02.10-00, ICS V02.11-01

DESCRIPTION:
The library routine UPLEVQQ has been omitted from the run-time library provided with the product. This routine is also not documented.

The routine is called when a variable outside the current scope is referenced.

The workaround is to assemble and link one of the following programs with PASCAL programs which are producing unresolved global reference errors to UPLEVQQ at link time.

For programs in which SMALL CODE is specified, use the following:

```
        SECTION  UPLVSC,ALIGN(1),CLASS=INSTRQQ
        GLOBAL   UPLEVQQ
; CONSTANTS IN CX
; RETURN POINTER IN BX

UPLEVQQ MOV BX,BP
```

```
LOOP1    MOV BX,SS:4[BX]
     LOOP     LOOP1
     RET
```

For programs in which LARGE CODE is specified, use the following:

```
     SECTION  UPLVLC,ALIGN(16),CLASS=INSTRQQ
     GLOBAL   UPLEVQQ
; CONSTANTS IN CX
; RETURN POINTER IN BX

UPLEVQQ MOV BX,BP
LOOP1    MOV BX,SS:8[BX]
     LOOP     LOOP1
     RETS
```

ENGINEERING ANALYSIS/RESPONSE:
This does not exist in the library because calls to internal routines are used instead.

BYTE SIZE ERROR IN PAS8086 COMPILER (5014)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS8086 V02.10-00

DESCRIPTION:
When you declare a variable public or absolute by an ORIGIN statement (and maybe otherwise) the data section or absolute section has one more byte than it should. If you look at the pseudo-assembly language the block statement generates the correct number of bytes. If you do a lstr on the object module the size of the data or absolute module is one more than seems correct. We came across this problem because the linker could not link in a section due to no space. Here are three examples:

1) test86.ps - is a subrange type of address which should generate 2 bytes.
2) reg86.ps - is an integer type complied with -i option which should
   generate 2 bytes
3) array86.ps - is an array of integer 1..10 compiled with -i and should
   generate 20 bytes.

Compile any of these modules pas -ldvsia file.ps >file.pl then lstr -nosv file.po and look at D.MODE_ONE (or in case 1 and 2 A.MODE_ONE).

ENGINEERING ANALYSIS/RESPONSE:
The resolver was placing section length into object code when length minus one was required.

PASCAL SOURCE CREATES VMS ERROR (5019)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00, VAX/VMS 3.X

DESCRIPTION:
A PASCAL source which when compiled causes a VMS error 'Improperly handled condition, image exit forced.', and a stack/register dump. The listing gets generated, but the module name in the listing heading

is 'x'. LDE does not complain about the source. PAS8086 (VMS) has the same problem. PAS8086 (UNIX) aborts with a memory fault during phase 1. The source follows:

```
/*--------------------*/
module testmod;
type byte = 0..255;
type event_art = (info);

type event_info = record
        case event : event_art of
        info : (data : byte);
end;

var v [EXTERN] : event_info;

procedure task; EXTERN;
procedure comp;
begin
  if (v.data < 1) or (v.data > 8) then
  begin
    {v.data := 0;}
  end;
end;

procedure active; PUBLIC;
begin
  case v.event of
  info : task;
  end;
end;

end.
/*--------------------*/
```

If the line '{v.data := 0;}' is changed from a comment to a source line, everything works fine.

ENGINEERING ANALYSIS/RESPONSE:
We have carefully studied for quite a while. It had deep-rooted causes and has now been fixed.

PDBZ8K SC BREAK STATUS (5027)

SYSTEM CONFIGURATION AND VERSION NUMBERS: Z8KPDB Version 1.05-00, Z8002 Emulator, 8540, 8562 TNIX 2.1a

DESCRIPTION:
If the user places a hardware breakpoint after a SC #0 instruction the emulator fails to execute the SC instruction.

Notes: 1) This is an unmodified Z8002 running at 6 MHz, 2) Happens on several (4) emulators, 3) Situation does not exist if trace is on or if no breakpoints exist around the SC #0 instruction, 4) This is occuring in mode 0, no prototype connected, 5) Field Service has checked at least one of the emulators exhibiting the situation and all is well according to them.

ENGINEERING ANALYSIS/RESPONSE:
No fix is planned at this time.

PDB68K PACKED RECORD PROBLEM STATUS (5029)

SYSTEM CONFIGURATION AND VERSION NUMBERS: 68000 PASCAL Compiler V2.02-01, 68000 PDB V2.0

DESCRIPTION:
In m.ps there is a structure LINE which is a packed record. It seems the compiler is doing something incorrectly (like wrong code). If you break on IDU_DEMO#6 and look at the variable CKI the value given is 00 which is incorrect. If you redeclare LINE as an unpacked record and recompile and rerun it, this situation goes away. So, there is some kind of problem with the code produced with PACKED RECORD in this file.

ENGINEERING ANALYSIS/RESPONSE:
This will be fixed in the next release of PDB68K on TNIX.

DISPLAY INDEX ROUTINE (5030)

SYSTEM CONFIGURATION AND VERSION NUMBERS: 68000 PASCAL Compiler V2.02-01, 68000 PDB V2.0

DESCRIPTION:
In this program there is a constant called DISPLAYS which is initialized. It is of type MENU_TEXT which is an array[1..116] of LINE. LINE is a RECORD. In PDB you set a breakpoint and want to look at values of DISPLAY. If you type DISPLAYS then you see all values of DISPLAYS which is fine. But there is no way to see a specific element of DISPLAYS. This is extremely frustrating if you want to check the 100th element of DISPLAYS. You also can not reference a part of the record which is probably due to the fact that you can not break it down into one element of the array. In the manual there are similar examples regarding variables, and I wonder if the problem here is that DISPLAYS is a constant.

ENGINEERING ANALYSIS/RESPONSE:
Constant arrays should be allowed in index routine. This was fixed in VMS V02.02-00 & V03.05-00; UNIX V02.07-00. Working on TNIX.

ICS INVOKES IMPROPERLY (5031)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS8086

DESCRIPTION:
The ICS macros that initialize the 80186 peripheral control block (PCB) do not function properly. The test case hung when it attempted to access the PCB. The bug is located in the PCB_IO_Init$ and

PCB_MEM_Init$ ICS macros. I have provided the small code, small data versions of these macros for reference.

```
;       ########################################
        MACRO   PCB_IO_Init$                    ; PCB_IO_Init$   pcb_base
;       ########################################
                                ; Sets bit 12 of the relocation
                                ; register to 0 (i/o state)
                                ; { Assume that pcb_base is
                                ;   256 byte aligned and less
                                ;   than or equal to 0ff00H }
        IF "#" = 1
pcb_base SET    0"1"
         MOVW   dx, #bits(pcb_base, 0, 16)      ; DX := Control block base addr.
         ADDW   dx, #0feH               ; DX := DX + offset to rel. reg.
         MOVW   ax, #bits(pcb_base, 8, 12)      ; AX :=Upper 12 bits of pcb_base
         ORW    ax, #02000H                     ; Bit 13 = 1, Bits 15,14,12 = 0
         OUT    dx, ax                  ; Output AX to reloc. reg.
        ELSE
         WARNING; ICS internal error - pcb_io_init$ invoked improperly
        ENDIF

        ENDM

;       ########################################
        MACRO   PCB_MEM_Init$                   ; PCB_MEM_Init$      pcb_base
;       ########################################
                                ; Sets bit 12 of the relocation
                                ; register to 1 (mem state)
        IF "#" = 1
pcb_base SET    0"1"
         PUSH   es                      ; Save old value of ES
         MOVW   ax, #bits(pcb_base, 4, 16)      ; ES := Control block base addr.
         MOVW   es, ax
         MOVW   bx, #0feH               ; BX := Reloc. reg. offset
         MOVW   ax, #bits(pcb_base, 8, 12)      ; AX :=Upper 12 bits of pcb_base
         ORW    ax, #03000H                     ; Bits 12,13 = 1, Bits 15,14 = 0
         MOVW   es:[bx], ax             ; Set reloc. reg. up for i/o
         POP es                         ; Restore old value of ES
        ELSE
         WARNING; ICS internal error - pcb_mem_init$ invoked improperly
        ENDIF

        ENDM
```

ENGINEERING ANALYSIS/RESPONSE:
This is fixed on the interim release for VMS and all subsequent releases. It will also be fixed in the upcoming TNIX release. For UNIX users, here's a work around:

Since I am the assigned engineer, I am providing the correct versions of the PCB_IO_Init$ and PCB_MEM_Init$ macros. They were given to me and they have been verified on our 8560. The customer can edit each file of macros (ics.scsd.mc, ics.scld.mc, ics.lcsd.mc, and ics.lcld.mc) and make the necessary corrections. The corrected macros should look the same for all four memory models. This correction will be made to the 4.X VMS version when it is released.

```
;       ########################################
        MACRO   PCB_IO_Init$                    ; PCB_IO_Init$   pcb_base
;       ########################################
                                ; Sets bit 12 of the relocation
                                ; register to 0 (i/o state)
                                ; { Assume that pcb_base is
                                ;   256 byte aligned and less
                                ;   than or equal to 0ff00H }
        IF "#" = 1
pcb_base SET    0"1"
         MOVW   dx, #0ff00h             ; DX := Orig. block base addr.
         ADDW   dx, #0feH               ; DX := DX + offset to rel. reg.
         MOVW   ax, #bits(pcb_base, 8, 12)      ; AX :=Upper 12 bits of pcb_base
         ORW    ax, #02000H                     ; Bit 13 = 1, Bits 15,14,12 = 0
         OUT    dx, ax                  ; Output AX to reloc. reg.
        ELSE
         WARNING; ICS internal error - pcb_io_init$ invoked improperly
        ENDIF

        ENDM

;       ########################################
        MACRO   PCB_MEM_Init$               ; PCB_MEM_Init$      pcb_base
;       ########################################
                                ; Sets bit 12 of the relocation
                                ; register to 1 (mem state)
        IF "#" = 1
pcb_base SET    0"1"
         MOVW   dx, #0ff00h             ; DX := Orig. block base addr.
         ADDW   dx, #0feH               ; DX := DX + offset to rel. reg.
         MOVW   ax, #bits(pcb_base, 8, 12)      ; AX :=Upper 12 bits of pcb_base
         ORW    ax, #03000H                     ; Bits 12,13 = 1, Bits 15,14 = 0
         OUT    dx, ax                  ; Output AX to reloc. reg.
        ELSE
         WARNING; ICS internal error - pcb_mem_init$ invoked improperly
        ENDIF

        ENDM
```

PASCAL PACKED DATA STRUCTURE (5041)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00, VMS 3.7

DESCRIPTION:
The compiler produces incorrect code when dealing with a packed data structure comprised of two other packed structures. If the size of the first "sub-structure" is an odd number of bytes, and you refer to a byte-aligned element of 8 or 16 bits size in the second "sub-structure", the compiler produces code which only affects the 1 or 2 lower bits of the element instead of the whole element.

The compiler produces correct code if the length of the "sub-structures" are an even number of bytes, or if the "main structure" is not packed.

Apparently the routine STUL2QQ (which is called to do the store) is passed the number of bytes to move, and should be passed the number of bits.

Pad the "sub-structure" to an even number of bytes.

ENGINEERING ANALYSIS/RESPONSE:
STUL2QQ routine parameters not matched, this has been fixed.

PASCAL TPAS.COM FILE (5045)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00, VMS V3.7

DESCRIPTION:
The tpas.com file turns off the "FACILITY", "SEVERITY" and "IDENTIFY" fields of the system error messages, and never turns them back on.

ENGINEERING ANALYSIS/RESPONSE:
CORRECTION: this is not an ENHANCEMENT--it was a bug and is now fixed. WORKAROUND: Type "set message /delete/facility/ident/severity/text" after compilation is completed.

PASCAL LISTING ERROR OUTPUT (5046)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00, VMS V3.7

DESCRIPTION:
If listing is disabled in a part of the source code by {$list-} directives, no errors in that part of the code are reported in the list file. All errors should be part of the listing.

ENGINEERING ANALYSIS/RESPONSE:
This will not be changed. This is as is it should be. The information is not lost if the user captures the error output.

CHAR STRING OMITTED IN MANUAL (5047)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00, VMS V3.7

DESCRIPTION:
A character string must be declared when starting index 1 in order to be assigned a "string-value". This requirement is not stated in the manual. For example:

```
var str1 : packed array [1..5] of char;
  str2 : packed array [11..15] of char;

str1 := 'Hello'; { no problem }
str2 := 'Adios'; { compiler gives error }
```

ENGINEERING ANALYSIS/RESPONSE:
This is cited in the Language Reference Manual (See Wirth).

COMPILER GENERATES INCORRECT CODE (5048)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.03-00, VMS 3.7

DESCRIPTION:
The compiler produces incorrect code when dealing with a subrange data type of three bytes in a packed record, if the subrange data type is located at an even address. The compiler generates a MOVE.L instruction, thus affecting 4 bytes.

Pad the record with a dummy byte in order to force an odd address for the subrange data type.

ENGINEERING ANALYSIS/RESPONSE:
The front-end and the code generator were both mishandling 3-byte ordinals.

CCC68K INTERNAL ERROR 943 EXPLANATION (5050)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CCC68K V02.09-00, TNIX 2.1b

DESCRIPTION:
For some C source files the compiler will die in the code generation phase with the error

'cg(943) Internal error.'

ENGINEERING ANALYSIS/RESPONSE:
The error 943 means the compiler has run out of memory and can not finish the compilation. This only occurs on the 856X version of the compiler.

PDB68K POINTER REFERENCE PROBLEM FIXED (5052)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PDB68K V02.01-00, VMS V3.7

DESCRIPTION:
PDB does not deal with pointers correctly in many situations. If you ask for the type of a pointer, PDB often answers "Untyped pointer".

ENGINEERING ANALYSIS/RESPONSE:
This has been fixed in base on 5/85 and will be fixed in all future releases of PDB68K..

SETKSH CAN'T FIND 8086 ASSEMBLER (5054)

SYSTEM CONFIGURATION AND VERSION NUMBERS: ColorKey+ V02.00-01 VMS, VMS V3.X

DESCRIPTION:
Setksh is looking in the wrong place for the 8086 Assembler. The assembler is in 80186.DIR but setksh is looking in 8087.DIR, so the assembler is never found. Recommended solution: change KSHOPTS.COM to look for the assembler in 80186.DIR when 'microname' = 8086. See KSHOPTS.COM for details.

ENGINEERING ANALYSIS/RESPONSE:
The interim VMS 4.0 8086 Assembler installs in two directories --- 8086 and 80186. This should take care of the ColorKey+ situation. All future releases of ASM8086 for all operating systems will also install in two directories.

PAS8086 EXTERN PROCEDURE (5060)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS8086 V2.10-00

DESCRIPTION:
When an EXTERN PROCEDURE with parameters is declared within a procedure, (rather than at the MODULE or PROGRAM level), there is an extra PUSH BP instruction generated when the external procedure is called.

ENGINEERING ANALYSIS/RESPONSE:
The PASCAL Language Reference Manual (pp 6-10, 8-2, 3) states that any procedure or function which is declared public or extern must be global. This was not strictly enforced in our compiler but will be in future versions.

PASCAL VARIABLES (5065)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS8086 V02.10-00, TNIX 2.1b

DESCRIPTION:
The PASCAL Language Reference Manual states on page 3-15 that a variable declared with ORIGIN is automatically given the PUBLIC attribute. This is not true. In addition, declaring a variable both ORIGIN and PUBLIC causes an error. There is no "clean" way to do this.

ENGINEERING ANALYSIS/RESPONSE:
This is now fixed.

PASCAL GENERATES BAD CODE (5071)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K 1A V02.02-01, TNIX 2.1 on 11/73

DESCRIPTION:
If the following PASCAL program is compiled without optimization, all is well. However, if optimization is turned on, bad code is generated.

```
program bug;

{$nilck-} {required since we are pointing outside of our heap}

type word = -32768..32767;

var addr1 {origin 16#F00000]:^word;
   addr2 [origin 16#F00004]:^word;
   i: integer;

   begin
    i:=addr1^ =addr2^;
    end.
```

Apparently the optimizer thinks both addr1 and addr2 point to the same place.

ENGINEERING ANALYSIS/RESPONSE:
This has been fixed.

ICS COMMENT LINES (5072)

SYSTEM CONFIGURATION AND VERSION NUMBERS: ICS68K V2.05-00, ICSP68K V2.02-00, VAX VMS V3.7

DESCRIPTION:
When running ICSP with the following lines in the .IS file, ICSP reports a file format error and tells the user to run ICS to determine what the error is. ICS reports no errors.

```
;
INSTRUCTIONS_ROM00420H-0D7FFH ; approximately 52K for CODE
;
CONSTANTS_ROM    0D800H-0DFFFH;.5K for CONST
;
GLOBAL_VAR_RAM  0E000H-0EFFFH,010000H-01FFFFH ; 64K for DATA
HEAP_STACK_RAM  0F000H-0FFCFH ; approx. 4K for H/S
;
```

By removing the empty comment lines between memory declarations the error is not reported. i.e., this works -

```
;
```

```
INSTRUCTIONS_ROM00420H-0D7FFH ; approximately 52K for CODE
CONSTANTS_ROM   0D800H-0DFFFH;.5K for CONST
GLOBAL_VAR_RAM  0E000H-0EFFFH,010000H-01FFFFH ; 64K for DATA
HEAP_STACK_RAM  0F000H-0FFCFH ; approx. 4K for H/S
;
```

The manual does not specify that there can not be comment lines between memory declarations.

ENGINEERING ANALYSIS/RESPONSE:
This entire concept has been reworked in the latest generation of ICS for all processors and languages.

PLDE GENERATES ERROR MESSAGE (5073)

SYSTEM CONFIGURATION AND VERSION NUMBERS: VAX 11/750 Ultrix 1.0 (4.2BSD)

DESCRIPTION:
Invocation of the ldeconfig command, supplied on the PLDE distribution tape, results in the message "Bus error - core dumped."

A workaround is to use the ldeconfig command supplied with CLDE.

ENGINEERING ANALYSIS/RESPONSE:
This has been fixed in UNIX and VMS.

68000 ADDRESSING (5079)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CCC68K V02.06-00, VMS 4.X

DESCRIPTION:
Because the 68000 is sensitive to certain accesses of odd addresses, a switch to cause all variables to start on even addresses would be convenient. This is not a factor for customers using strictly LANDS, but can cause problems to customers trying to interface to other systems.

ENGINEERING ANALYSIS/RESPONSE:
There is an easy workaround: force alignments by not packing records. We are not planning for this enhancement.

MACRO PARAMETER REFERENCING (5096)

SYSTEM CONFIGURATION AND VERSION NUMBERS: VAX/VMS Version 3.X, ASM68K Opt 1F V2.09-00

DESCRIPTION:
If a user references the 0th parameter in a macro the assembler blows up with a VMS stack trace. The customer has about 300 pages in listings to wade through to determine the situation. While referencing the 0th parameter in a macro is illegal, a error message in the listing would be of much greater use than a VMS blowup.

ENGINEERING ANALYSIS/RESPONSE:
All future assembler releases will include the fix.

CDB GENERATES CHECKSUM ERROR (5099)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CDB68K V01.08-00, TNIX 2.1b

DESCRIPTION:
When the following source is compiled and loaded via CDB, an error message 'cdb: checksum error' is given. If the '-af' switch is used to specify the fast floating point library, no error is given by CDB when loading, and the program executes.

```
-----------------------
#include <math.h>
main() {
  double a;
  a = cos((double) 23.45);
  printf("%e0, a);
}
-----------------------
```

Note: The same .lo file which gives the error message under CDB can be
loaded via 'lo' and will execute.

ENGINEERING ANALYSIS/RESPONSE:
This program works on $TEKPATH/tek/c/68000/cdb. The user should make a backup copy of the original and then run this program (requires native programming option).

PASCAL COMPILER EXITS INCONSISTENTLY (5103)

SYSTEM CONFIGURATION AND VERSION NUMBERS:
8086 PASCAL Compiler for the 8560 V2.10-00
68000 PASCAL Compiler for the 8560 V2.02-01
68000 PASCAL Compiler for the VAX/VMS V2.03-00

DESCRIPTION:
The following file causes an extremely ungraceful exit from the compiler. The way it exits is different depending on the compiler.

The 8086 Compiler on the 8560 bombs with 2
sh: # bus error - core dumped errors
one after phase 1 and one after phase 3 and then gets an

RES: 924(c) Compiler Internal Error

A listing is produced but no object code.

The 68000 Compiler on the 8560 bombs after phase 1 with an

Internal Execution Error # 19

No object code is produced.

The Z8000 Compiler compiles this code and produces an object file.

The 68000 Compiler on the VAX/VMS after phase 1 gets an error from the VAX which says:

> Improperly Handled Condition image exit forced then gives
> signal arguments, a stack display and a register dump.

After this display the compiler continues and says it completed with no errors but does NOT produce an object file.

****

```
module ICPUTILS;

VAR MESSAGE:                    integer;
  XMIT_BUFFER:                  integer;

  procedure IICPPUTBUF;
  procedure IICPSTXMTF(
                  PRT:          integer;
                  SLOT:         integer;
                  TEMP:               integer); extern;
  begin

    XMIT_BUFFER := MESSAGE;       { put msg into buffer }
  end;

  PROCEDURE IICPBRDSEL(TRP:INTEGER);

   BEGIN

     TRP := TRP - 8;     {changing port 8-15 to relate to 0-7}
   END;

   procedure iicpstxmtf (prt : integer;
   slot_no : integer;
   temper: integer);public;

   VAR pbit : integer;

   BEGIN

      slot_no :=  pbit;
   END;
end.
```

****

There is a procedure that is declared external which is nested in another procedure and then later defined. If you rearrange this the problem goes away. If you take out the middle procedure or limit the parameters passed to two the problem also goes away. But, depending on the order of things in the file, which compiler you are using etc. this problem is handled differently.

ENGINEERING ANALYSIS/RESPONSE:
We have fixed in the 68K products on TNIX and VMS version 4 both of which will be released in the near future.

PDB68K BUFFERS OUTPUT (5105)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PDB68K V02.03-01, TNIX 2.1b

DESCRIPTION:
When executing under PDB, the output of the PASCAL program keeps a buffer of the last 512 (256?) characters. These characters do not come out until either a breakpoint is encountered, or a control-C is typed. This fact is not stated in the manual.

ENGINEERING ANALYSIS/RESPONSE:
This will be fixed (to not buffer) in the next release of PDB68K 1A, which is scheduled for 1/86.

FLOATING POINT ROUTINES CRASH (5106)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K V02.08-01, TNIX 2.1b

DESCRIPTION:
Under certain conditions the floating point routines supplied with PAS68K 1A are not interruptible. Small modules work fine, but some large modules crash. Apparently the A6 register gets corrupted on a return and the program goes haywire.

ENGINEERING ANALYSIS/RESPONSE:
This is now fixed.

CCC68K SIN/COS (5107)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX version 2.1

DESCRIPTION:
The Floating Point libraries distributed with the 68K C compiler fail to return correct results. The sin() function return -0.00 from PI(180 degrees) to 2PI(360 degrees) and the cos() function returns -0.00 from PI/2(90 degrees) to 3PI/2(270 degrees). These routines do function correctly in the Default Floating Point library.

ENGINEERING ANALYSIS/RESPONSE:
A solution has been tested (but not evaluated), and solves the reported problem, with no apparent ill effects.

PDB DOESN'T TRACE REAL FUNCTION WELL (5108)

SYSTEM CONFIGURATION AND VERSION NUMBERS: VMS 4.0 Y03.05-00, UNIX 4.2 V02.02-00
DESCRIPTION:
When you trace a function that returns a real value, PDB fails when it tries to print the return value when the function exists. On UNIX, PDB will print an incorrect value. On VMS, PDB will exit with an Access Violation and a stack and register dump.

ENGINEERING ANALYSIS/RESPONSE:
We are in the process of fixing this now. It will be in the VMS release for V4.

PASCAL MODULE BOMBS COMPILERS (5110)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K, PASZ8K, PAS8086 856X running TNIX 2.1b on an 11/73

DESCRIPTION:
The PASCAL module below bombs all of our PASCAL compilers on an 8560 running an 11/73. Each compiler gives slightly different results. If the module is compiled with PAS68K on an 11/23 based system, it goes through without any problems. The output the stderr for each compiler is noted below. the invocation was identical for each : pas -ldsv test.ps.

PAS68K:
MDP PASCAL 68000/68010 Compiler V02.02-01 (8560)
Copyright (C) 1983 Tektronix, Inc.
Phase 1...
Phase 2...
Phase 3...
Phase 4...

*** Internal Execution error # 19

PASZ8K:
MDP PASCAL Z8001/2 Compiler V01.10-06 (8560)
Copyright (C) 1982 Tektronix, Inc.
Phase 1
Phase 2
Phase 3
Phase 4
Phase 5

*** Execution error # 19

PAS8086:
MDP PASCAL 8086/8088 Compiler V02.10-00 (8560)
Copyright (C) 1984 Tektronix, Inc.
Phase 1
Phase 2
Phase 3
Phase 4

```
sh: 21533 Memory fault - core dumped
Phase 5
0 Errors
0 Warnings
```

Only the 8086 generates a listing file:

```
MDP PASCAL (V02.10-00  (8560))                    TESTING1
22-Aug-85 16:38:25

  1             module testing1;
  2             const   numrows          = 24;
  3               numcols          = 80;
  4               screensize   = 1920;
  5
  6             type    strng          = packed array [ 1..screensize ] of char;
  7               s_ptr          = ^strng;
  8               rcd            = RECORD
  9           0:B         VAL1         : s_ptr;
 10           2:B               val2   : integer;
 11           6:B               val3   : boolean;
 12           7:B               val4   : strng;
 13                       end;
 14             end.
          0 Errors
        0 Warnings

     Debug :  ON
   Optimize : OFF
  SmallCode :  ON
  SmallData :  ON
SmallInteger : OFF
```

The module in question:

```
module testing1;
const numrows          = 24;
      numcols          = 80;
      screensize   = 1920;

type  strng          = packed array [ 1..screensize ] of char;
      s_ptr          = ^strng;
      rcd            = RECORD
            VAL1         : s_ptr;
            val2   : integer;
            val3   : boolean;
            val4   : strng;
            end;
end.
```

ENGINEERING ANALYSIS/RESPONSE:
This is now fixed in 68000 Compilers version V03.03-00 and higher. (That's the interim VMS release.) It will also appear in the next 8086 release as well.

CCC68K AUTO INCREMENT PROBLEM (5113)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX 2.1b, VAX UNIX 4.2bsd, VAX VAX 4.X

DESCRIPTION:
If auto-increment is after a 'while' statement on one of the variables that is in the while expression, the auto-increment will occur at the time of the compare instead of after the while. See example below:

```
int i, v;
main() {
 register char *c;
 while (*c) {
  i = 0;
  while (v >= *c)
   i++;
  c++;
 }
}
```

ENGINEERING ANALYSIS/RESPONSE:
Put a label before the auto-increment statement.

CCC68K REGISTER VARIABLES (5114)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX 2.1b, VAX UNIX 4.2bsd, VAX VMS 4.X

DESCRIPTION:
If the user defines 4 data register values and the 4th is in an Address register, if '|=' is used on the variable, the operation will be performed in a data register and the assignment will never take place. See example below.

```
fun(ptr, nchuncks, width, mask) {
 register unsigned long a, b, c;
 int d, width, mask;

 for (a = 0; a < 10; a++) {
  c = get_row(ptr, a, nchuncks);
  c |= (mask | (~0 << width) | d);
  if (~c) {
   printf();
  }
 }
}
```

ENGINEERING ANALYSIS/RESPONSE:
Remove the register preferred or expand x |= y to x = x | y.

CCC68K PUSHING 1-BYTE STRUCTURES (5115)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX 2.1b, VAX UNIX 4.2bsd, VAX VMS 4.X

DESCRIPTION:
Pushing a structure by value, with size 1 byte and assignment is done to another element of the same type, produces bad code under the -d option. Extra code is produced, causing the assembler error "illegal effective address form". See example below:

```
struct complex {
 char c;
};
extern struct complex p;
abc(s)
struct complex s; {
 s = p;
}
```

ENGINEERING ANALYSIS/RESPONSE:
Do not have structures of size 1 byte. Make them just char type.

CCC68K PUSHING FLOAT VALUES (5116)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX 2.1b, VAX UNIX 4.2bsd, VAX VMS 4.X

DESCRIPTION:
Pushing an element of a structure of type float produces bad code. 4 bytes are pushed, and the float is not converted to double. See example below:

```
struct ftype {
  float f;
} s;
main() {
  printf("%g0, s.f);
}
```

ENGINEERING ANALYSIS/RESPONSE:
Cast the float to double.

856X FLOPPY DISC MOD (5118)

SYSTEM CONFIGURATION AND VERSION NUMBERS: 8560, 8561, 8562 with the LSI 11-73 installed.

DESCRIPTION:
Extensive flexible disc access, as in fbr, occasionally will hang up the 8562. If more than one process was active at the time, the other processes will continue active until a disc access is required at which time they will hang as well. Finally only re-booting the 8562 will clear the system.

ENGINEERING ANALYSIS/RESPONSE:

The following changes are made to the MSC board

1) lift U1080 pin 4
2) wire from U2015 pin 9 to U1080 lifted pin
3) wire from U2015 pin 10 to U2060 pin 4
4) wire from U2015 pin 11 to U1100 pin 3
5) wire from U2015 pin 12 to U1040 pin 4
6) wire from U2015 pin 13 to U4020 pin 11

ASSEMBLER FORWARD REFERENCE (5119)

SYSTEM CONFIGURATION AND VERSION NUMBERS: LAS Assemblers

DESCRIPTION:
If a global is equated to an expression containing a forward reference, the information in the object module will not be correct. See example below.

```
            global    glob1
glob1       equ       forward
forward     byte      00h
            end
```

ENGINEERING ANALYSIS/RESPONSE:
Move the EQU statement to the end of the program so that there is no forward reference.

ACE UNDER KSH CAN ABORT (5120)

SYSTEM CONFIGURATION AND VERSION NUMBERS: 856X TNIX V2.1b, ACE V3.00-00

DESCRIPTION:
ACE will abort with the message "insufficient memory, aborting" if TERMCAP is defined as the terminal entry instead of /tek/lib/termcap. KSH sets TERMCAP as the terminal entry, therefore, ACE does not run when directly entered.

ENGINEERING ANALYSIS/RESPONSE:
Invoke ACE through ace.assist or to redefine TERMCAP before editing.

ACCESS VIOLATIONS ON C (5122)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CCC68K Y03.06-00, VMS 4.1

DESCRIPTION:
The following program, when compiled, gives an access violation and core dump:

```
typedef struct {
 unsigned char state;
 unsigned char fille;
} FSM_DATA;

typedef FSM_DATA FSM_TAB[];
typedef FSM_TAB *IND_FSM_DATA[];
typedef IND_FSM_DATA *FSM_PTR[];

extern FSM_TAB ccsfsmdata[];
extern FSM_TAB eeafsmdata[];

IND_FSM_DATA cirpind0[] = {
 ccsfsmdata[1],
 eeafsmdata[2]
};
```

Note: The native C Compilers on azure and copper detect syntax errors on lines 14 and 16.

ENGINEERING ANALYSIS/RESPONSE:
The Parser should have flagged these declarations as incorrect (the compiler should disallow declarations of pointers to null dimensioned arrays). The typedefs used to show is not the problem. Since this is an incorrect program, this situation is not serious.

ICS SHARABLE IMAGE PROBLEM (5123)

SYSTEM CONFIGURATION AND VERSION NUMBERS: VAX VMS V4.1, CCC68K Y03.06-00

DESCRIPTION:

When ICS is installed as a sharable image, VMS will not allow it to execute properly. If the sharable attribute is removed, ICS runs fine.

This situation occurs when both [tek8540.bin]ics.exe (the driver) and [tek8540.c.68000]ics.exe (the executable) are made sharable. Apparently, VMS cannot handle 2 sharable images in memory at the same time.

ENGINEERING ANALYSIS/RESPONSE:
1) Make only one of the ics.exe's sharable, preferably the second.
2) Rename the driver to icsdriver.exe and redefine the symbol to
ics :== "$tek8540_device:[tek8540.bin]icsdriver.exe"
Note: You need to make new name (icsdriver.exc) sharable.

PAS68K FORWARD PROCEDURE (5124)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX 2.1B, PAS68K V02.08-01

DESCRIPTION:
The following program compiles without error. But at link time it gives a link error message saying 'name in I.FORW already defined.' By running lasdmp on the .po file one can see that the error is correct.

```
-----------------------------------
program forw(output);
procedure one; public; forward;
procedure two; public; forward;
procedure one;
begin;
  writeln('procedure one');
end;
procedure two;
begin;
  writeln('procedure two');
end;
begin;
   writeln('main line forw');
   one;
   two;
end.
-----------------------------------
```

ENGINEERING ANALYSIS/RESPONSE:
This has been tested on the version under development and will behave properly on version V03.03-00 and higher. There is no semantic errors in this example.

COMPILER GENERATES INCORRECT ADDRESS (5128)

SYSTEM CONFIGURATION AND VERSION NUMBERS: VAX/VMS V4.2, CCC8086 V1

DESCRIPTION:
When the address of a member of a two dimensional array is taken in the initialization of a static or global variable, the array subscript calculation is done incorrectly. The compiler does not issue a message; it simply generates an incorrect address.

One dimensional arrays work correctly, and taking the address of element 0 works correctly.

```
char a1[][2];
char *p1 = a1[3];          /* Generates incorrect address in p1 */
char *p2 = &a1[5][7];      /* Generates incorrect address in p2 */
char *p3 = a1[0];          /* Works correctly */
char *p4 = &a1[0][0];      /* Works correctly */

char a2[];
char *p5 = &a2[3];         /* Works correctly */
```

ENGINEERING ANALYSIS/RESPONSE:
This will be fixed in 8086 V2 compiler.

UNIONQQ NON-INTERRUPTIBLE (5129)

SYSTEM CONFIGURATION AND VERSION NUMBERS: TNIX V2.1, PAS68K V02.02-01

DESCRIPTION:
It appears that the runtime library routine UNIONQQ (set union) is non-interruptible, thus rendering PASCAL statements of the form

```
set := set_1 + set_2;
```

unusable unless protected from interrupts.

ENGINEERING ANALYSIS/RESPONSE:
An interrupt in the midst of an add routine could have left the stack pointer with an odd value if interrupted at the proper (improper) time. This has been fixed.

MANUAL SHOWS INCORRECT ICS SOURCE FILE (5130)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CCC68K, 856X Host

DESCRIPTION:
Pg 6-15 shows an example of an ICS source file which is incorrect. The USER_DEFINED_SYMBOLS must be prefixed with "c." for example "c.pedport=3000". This example needs to be corrected.

ENGINEERING ANALYSIS/RESPONSE:

The example will be corrected in all 68K manuals (TNIX, UNIX 4.2, VMS 3.4).

C COMPILATION ERROR (5131)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CCC68K 1C V02.09-00, VAX/UNIX 4.2BSD

DESCRIPTION:
The following program gives an internal error 942 on compilation. Changing the line 'typedef char TEXT;' to '#define TEXT char' eliminates the problem.

```
typedef char TEXT;
typedef struct {
  TEXT *a;
  TEXT *b;
  TEXT array1[8];
  TEXT array2[9];
} COMM;
static COMM comm = {
  0, 0, {0}, {0}
};
```

ENGINEERING ANALYSIS/RESPONSE:
Will be fixed in the next releases of CCC68K on UNIX and VMS.

ON-LINE MANUAL INCORRECT (5133)

SYSTEM CONFIGURATION AND VERSION NUMBERS: All 856X, All Versions

DESCRIPTION:
The on-line manual page for the tail command describes the use of the -s flag to see the start of a file instead of the tail of a file. This flag does not work as documented, but seems instead to default to displaying the entire file.

ENGINEERING ANALYSIS/RESPONSE:
Description of -s option will be deleted in TNIX 2.1c man page.

CLDEDIT/TNIX OVERWRITES COLORKEY+ FILES (5135)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CLDEDIT,OPT.1A, Ver. 01.07-00; TNIX Ver.2.1b

DESCRIPTION:
The installation command file on the CLDEDIT/TNIX disc overwrites various ColorKey+ script files indiscriminately, causing numerous errors as the user attempts to build and/or execute ColorKey+ setups.

ENGINEERING ANALYSIS/RESPONSE:

Four ColorKey+ files (/tek/ksh/lib/kshoptions, /tek/ksh/scripts/include/edit, /tek/ksh/scripts/main and /tek/lib/termcap) are installed without regard to Keyshell Version Number. The script, edit, unfortunately, is the UNIX version. (This is why vi comes up on the key labels, but ace does not.) The Keyshell Version Number of the script, main, is 1, whereas the current main is Version 4. The solution is to re-build the CLDEDIT 1A product with a new install procedure, and the correct version of the four ColorKey+ files. This will allow it to be installed on TNIX 2.1b or 2.1c. The CLDEDIT 1A ECO will be ready by December (1985).

ICOM40 MANUAL INCOMPLETE (5136)

SYSTEM CONFIGURATION AND VERSION NUMBERS: ICOM40A, VAX/VMS V4

DESCRIPTION:
The Manual does not discuss downloading symbols, or give the new syntax for doing so. Somewhere around page 3-9 it should give an example ie., "ICOM symlo /input=VAXFILE.EXT".

The manual does not give the limitations of executing 8540 strings and permstrings. Although the 8540 User's manual shows examples with multiple commands within one string ie "SETUP = sel z80; ds; d 0 100" when invoked with ICOM40 this type of string will not execute properly because of the config term interface.

ENGINEERING ANALYSIS/RESPONSE:
Change pages will be developed for all ICOM40 manuals to reflect these.

CCC68K SOURCE INSTRUCTION (5138)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CCC68K V02.09-00 (UNIX)

DESCRIPTION:
When the following source is compiled, a 'byte 0' instruction is generated at the end of the assembly output. Apparently this is to pad the data section out to an even number of bytes, but it causes problems in em 1.

TC has seen this problem before.

The source:

```
char a;
char b;
char c;
```

The assembly code:

```
        gen.l
        trn$
        name cn.extvar
        section     ic.extvar, align(2), class=CODE.A
        section     dc.extvar, align(2), class=DATAC.A
        section     di.extvar, align(2), class=DATAI.A
```

```
        section     dv.extvar, align(2), class=DATAV.A
        section     du.extvar, align(2), class=DATAU.A
        resume      ic.extvar
        global      c.a
        resume      du.extvar
c.a     block 1
        global      c.b
        org   /2
c.b     block 1
        global      c.c
        org   /2
c.c     block 1
        resume      du.extvar
        byte  0                 <--- should be 'block 1'
        end
-------------------------------
```

The workarounds are either 1) pad the declarations to be an even number of bytes, or 2) change the 'byte 0' to a 'block 1'

CLDE'S SYMBOL TABLE CAPACITY (5139)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CLDEDIT 1A, TNIX V2.1a

DESCRIPTION:
There is a condition with #IFDEF in conjunction with stdio.h. The user uses conditional includes for stdio.h according to the variable DEBUG. This variable DEBUG is true when wanting to include stdio.h for debug purposes. If it is false, user does not wish to include stdio.h. This should be fine since the only time stdio functions are called is when DEBUG is true. Regardless of file size, the system is having problems parsing the file because it generates the following error: out of symbol table space

ENGINEERING ANALYSIS/RESPONSE:
CLDE has a symbol table that keeps all of the #defines and typedefs definitions. The include files must have many typedefs, defines, and/or macro definitions that are kept in CLDE's symbol table. The symbol table is not as big as the compiler's since CLDE has both the pre-processor and the parser symbol table.

The error message is correct, CLDE's symbol table is full.

LDE SCROLLING (5141)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PLDEDIT 1A, V02.04, 8562, TNIX V2.1a, Terminal Type: VT102

DESCRIPTION:
LDE paging through files. For example if user scrolls down 3 full pages of text, and starts scrolling back up the screen, the text becomes jumbled and will not be correct until a <control>L is performed. This condition is consistent, and the length is not a factor: the same situations are exhibited on a 100 line file or a 1000 line file.

ENGINEERING ANALYSIS/RESPONSE:
There is no way to duplicate this problem, since lde does not have a termcap entry for VT102, and the ldeconfig file supplied for the VT100 contains a bug (DEL key does not get recognized). Lde reports a error message 'termcap entry too big' for the VT100 termcap entry in /etc/termcap.

This means the user has his own termcap entry and his own ldeconfig file. My best guess is that the termcap entry is incorrect, more than likely its the number of lines specified for the screen or the scroll entries, but then again it could be any entry.

PLDE DELETES FILE (5145)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PLDEDIT 1A V02.04-00 TNIX 2.1 with 11/73

DESCRIPTION:
If you are running plde, and then invoke plde again thru the exec sys command, and then return from the second invocation, the temporary termcap file gets deleted.

ENGINEERING ANALYSIS/RESPONSE:
The only time the above would happen is if the variable TERMCAP contains the actual termcap entry and not a file (ColorKey+ does this).

The lde driver takes the TERMCAP entry and places it in a temporary file under /tmp, the driver forks lde by execv, which never returns to the driver. Lde has to recognize that the TERMCAP entry is a dummy file created by the lde driver in order for lde to delete the file before exiting.

Solution (1)
Make sure TERMCAP entry is a file, that does not reside in /tmp.

Solution (2)
When invoking the second lde, reset the TERMCAP variable to be a termcap file.

COMPILER OPTIMIZATION (5151)

SYSTEM CONFIGURATION AND VERSION NUMBERS: 8560 TNIX V2.1b, Z8000 PASCAL Compiler V1.09-08

DESCRIPTION:
The compiler causes bad code to be generated for the following program. The switches used were valid

```
----------------------------------------
MODULE UTMEMTST;

{$mathck-} {$nilck-} {$rangeck-} {$tagck-}
```

```
.
100 lines of user's PASCAL source, documented in actual PPR,
omitted here for brevity
.
.

END.
```

The optimizer causes bad code to be produced for the statement:

IF EEPROM_TST_PTR.EEPROM_PTR < 0 THEN

(If optimization is turned off the condition does not occur.)

It appears to be caused by the optimizer remembering that R12 contains EEPROM_TST_PTR. Once it generates the code to load R13 with EEPROM_TST.PTR.EEPROM_PTR, it generates an LD that wipes out this value.

I found two workarounds for this problem:

1 If the statements starting with IF EEPROM_TST_PTR.EEPROM_BANK AND 16#0080) = 1 THEN are moved after the IF statement for EEPROM_TST.PTR.EEPROM_PTR correct code is generated.

2 If EEPROM_TST_PTR is assigned to a temporary pointer and the IF statement is changed to IF TMP_PTR.EEPROM_PTR < 0 THEN correct code is also generated

Both these workarounds help the immediate problem in the particular file, but do not explain how to avoid it in a new file.

COMPILER OMITS ERROR MESSAGES (5154)

SYSTEM CONFIGURATION AND VERSION NUMBERS: VAX/UNIX 4.2, CCC68K V02.09-00, VAX/VMS 4.1, CCC68K Y03.06-00, VAX/UNIX 4.2, CCC8086 V01.03-00, VAX/VMS 4.1, CCC8086 V01.03-00

DESCRIPTION:
The following program is incorrect, because it is trying to initialize a static variable with a variable expression, but the compilers do not generate an error message.

The VMS 68000 Compiler bombs with an access violation. The UNIX 68000 Compiler produces garbage assembly code that causes the assembler to issue an error. The 8086 compilers on both VMS and UNIX generate assembly code as if the user had used "&ddb[1]" as the initializer.

```
struct b { short a,b,c; };

struct k {
        struct b *i;
   };
```

```
struct b ddb[5];

struct k d = {
  ddb[1]
};
---------------------------
```

ENGINEERING ANALYSIS/RESPONSE:
This will be fixed in CCC68K V3 and CCC8086 V2 to generate an error message.

PLDE CREATES BAD FILE (5158)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PLDEDIT 1A, 856X Hosts, TNIX V2.1b

DESCRIPTION:
When installing PLDE on a system, that has not previously had LDE installed, sometimes during the installation a file is created called "/usr/lib/lde" Since this is supposed to be a directory several errors are created as the installation tries to put files under /usr/lib/lde. The fix appears to be to remove the file "/usr/lib/lde" and create a directory "/usr/lib/lde" and invoke the installation again.

CDB TRACE COMMAND (5162)

SYSTEM CONFIGURATION AND VERSION NUMBERS: 8560, TNIX V2.1b, 8540, rompatch 68, 68XXX emulator and 68000A probe, TTA

DESCRIPTION:
The trace command of CDB shows a "return" line for each pass through a loop, if the function ends on the loop. Inserting any executable statement after the loop (except "return()") fixes the condition.

POSI ROUTINE (5163)

SYSTEM CONFIGURATION AND VERSION NUMBERS: 8560, TNIX V2.1b, 8540, Rompatch 68, 68XXX Emulator and 68000A Probe

DESCRIPTION:
The POSI routine which assigns 8540 channels contains a channel table which is not explicitly initialized. The first time a program is loaded and run, the channels are assigned starting with 0 and working upwards. If an error occurs or the program is halted, and then the program is run from the entry point, the channels are correctly closed, but the table is left as is. The result is that new channels are assigned starting from the last channel not used previously, even though all channels are available. Eventually, the program will try to assign channels reserved for other use and errors will be reported. Depending on the number of channels used, the program will fail every third time, or fifth time, etc. The condition does not occur if the program is loaded each time.

C SOURCE GENERATES BAD CODE (5167)

SYSTEM CONFIGURATION AND VERSION NUMBERS: CCC68K V02.09-00, TNIX V2.1b

DESCRIPTION:
The following C source generates bad object code. In particular, the assignment statement puts both the source and destination addresses into a1, which doesn't work correctly.

```
------------------------------
struct operation {
  int i;
  int a;
};

struct PT_operation {
  struct operation *oper_ptr;
} *oper;

struct BufHeader {
  int h;
  int u;
  char *buffer;
} *ptr_bh;

main() {

  (*oper).oper_ptr = (*ptr_bh).buffer;
}
---------------------------------------
        gen.l
        trn$
        name cn.src1
        section     ic.src1, align(2), class=CODE.A
        section     dc.src1, align(2), class=DATAC.A
        section     di.src1, align(2), class=DATAI.A
        section     dv.src1, align(2), class=DATAV.A
        section     du.src1, align(2), class=DATAU.A
        resume      ic.src1
        global      c.main
c.main      link    a6,#0
        move.l      a1,-(sp)
;(*oper).oper_ptr = (*ptr_bh).buffer;
        move.l      c.ptr_bh,a1
        move.l      c.oper,a1           ; <-- a1 from last move is nuked
        move.l      8(a1),(a1)
Lp_1
        move.l      -4(a6),a1
        unlk  a6
        rts
;struct operation {  int i;  int a;};struct PT_operation {  struct operation *oper_ptr;} *oper;struct BufHe(
        global      c.oper
        resume      du.src1
c.operblock 4
        global      c.ptr_bh
```

```
c.ptr_bh     block 4
       end
-----------------------------
```

EXECUTION ERROR #19 (5168)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PAS68K 1A V02.08-01

DESCRIPTION:
When the following module is compiled an 'illegal instruction - core dumped' message is given (some have reported execution error 19). If the 'public' attribute is removed the error of the undeclared type is detected correctly.

```
-------------------------
module comp_test;
  procedure proc_test(param : undecl_type); public;
    begin
      { do nothing }
    end;
end.
-------------------------
```

This is similar to PPR 3123.

MODIFICATION OF FILES (5169)

SYSTEM CONFIGURATION AND VERSION NUMBERS: PLDE V02.04-00, TNIX V2.1b

DESCRIPTION:
If you attempt to modify the files 'lde.vt100.cfg' or 'lde.vt100k.cfg' using PASCAL ldeconfig a message 'not a valid configuration file' will be given. The C ldeconfig has no problem.

Apparently the file header is from a VAX and should say 8560. LDE can be used to edit the file and change this.

LDE CONFIG DELETES FILES (5170)

SYSTEM CONFIGURATION AND VERSION NUMBERS: LDE, TNIX V2.1b

DESCRIPTION:
When using ldeconfig it is possible to inadvertently delete the contents of the cfg file. The following sequence will do it:

```
$ ldeconfig lde.$TERM.cfg
delete contents of one of the fields
try to save file using ^W (gives msg about blank field)
exit via ^D
```

Apparently ldeconfig deletes the file before checking for blank fields.

LDE TEMPLATES (5171)

SYSTEM CONFIGURATION AND VERSION NUMBERS: LDE, TNIX, VAX/UNIX, VAX/VMS

DESCRIPTION:
The VT220 (and maybe VT240) do not have linefeed keys where our template for lde thinks it is.

The suggestion is to include a lde.vt200k,cfg file which configs the ENTER key to be a linefeed.

ASSEMBLER CORE DUMPS (5173)

SYSTEM CONFIGURATION AND VERSION NUMBERS: All LAS assemblers with -16 base

DESCRIPTION:
The following program causes the assembler to core dump. If the repeat/endr construct, the if/else/endif construct or the ';pop' line is removed, the assembly goes fine:

```
            macro       unsave
                  if ( '"1"' = 'all' ) ! ( '"1"' = 'ALL' )
            unsave      ax,bx,cx,dx,bp,si,di,ds,es
                  else
xx_unsave_count set     "#"
                        repeat xx_unsave_count > 0
             ;pop "xx_unsave_count"
xx_unsave_count set     xx_unsave_count - 1
                        endr
                  endif
      WARNING
            endm

AIRQE       UNSAVE  ALL

      end
```

PRODUCT PERFORMANCE REPORT

PRODUCT NOMENCLATURE AND SERIAL NUMBER: Enter the product description and/or order name and serial number, i.e., "ACEDIT OPT. 1A, B010101"

SYSTEM CONFIGURATION AND VERSION NUMBERS: Include version numbers for all involved products and operating system.

IS THIS SOFTWARE COVERED BY SOFTWARE SUBSCRIPTION SERVICE?

DESCRIPTION: Include source, results obtained, and results expected, **on disk or tape.** Please submit the minimum source code required to demonstrate the problem. Complete documentation will enable us to duplicate the problem.

REPORTED BY:

Customer Name _____ Date _____

Company Name _____ Title _____

Company Address _____

Internal Address/Dept. _____

City _____ State _____ Zip Code _____

Area Code _____ Tel. No. _____ Ext. _____

Send this form along with the tape or disc as described above, to your local Tektronix SDP representative. Your local representative will verify that this report is not already contained in our database and will then forward it to SDP Engineering for resolution.

INDEX OF VOLUME 3

COMPUTER AIDED
SOFTWARE ENGINEERING

USER GROUP NEWS

3

For further information, contact:

U.S.A., Asia, Australia, Central & South America, Japan
Tektronix, Inc.
P.O. Box 14752
Portland, Oregon 97214
For additional literature, or the address and phone number of the Tektronix Sales office nearest you, contact:
Phone: 1-(800) 342-5548
Oregon only: 1-(503) 629-1573
TWX: (910) 467-8708
TLX: 151754
Cable: TEKWSGT

Europe, Africa, Middle East
Tektronix Europe B.V.
European Headquarters
Postbox 827
1180 AV Amstelveen
The Netherlands
Phone: (20) 471146
TLX: 18312 – 18328

Canada
Tektronix Canada Inc.
P.O. Box 6500
Barrie, Ontario L4M 4V3
Phone: (705) 737-2700
TLX: 06-875672
Cable: TEKANADA

Tektronix®
COMMITTED TO EXCELLENCE