THE CACHE r.c.l.

REGISTER

Published for the Chicago Area Computer Hobbyist's Exchange

P.O. Box 94, Bourbonnais, Illinois 60914 Hotline (312) 849-1132

CUTS Software!!!

by Tom Digate

Do you need software for your SOL or CUTS cassette interface? If you answered yes to this question, then read on.

I have, with the help of Larry Marek and his Tarbell cassette interface, converted the CACHE software library to SOLOS / CUTER compatibility. What does the library have in it? Basically, it contains about 53 BASIC programs, 40 CCOS compatible source programs, and 30 object programs. I am still in the process of modifying these programs to run using SOLOS routines; however, I have converted CCOS to use SOLOS routines.

SCOS (SOLOS Casette Operating System) is a modified version of Processor Technology Corp. Software Package No. 1. The modifications include 9 extra commands: SSAV, SLOA, OSAV, OLOA, OLOADX, SPEE, RENU, PRIN, and VERI. These commands allow the user to save and load source and object files without line numbers, and verify the integrity of source files. The program now runs at 10c. 0 and uses 4k for program area, 256 bytes for read / write area, and 1.75k for its symbol table. In the future, I hope to add a relocator to it to allow one to run it in any 4k block of memory.

I would like to share this software with all CUTS cassette interface users. I will send you a copy of SCOS, all worthwhile CACHE library programs, and some documentation on how to use SCOS for the price of the cassette and copying — $3.00.

If anyone is interested in starting a CUTS users group within CACHE, I would be willing to coordinate a software library.

Send your requests to Tom Digate, 1366 S. Finley Rd., Apt. 3s, Lombard, IL 60148, or call me at 312—620-4656.

SOLOS Cassette Operating System
SCOS)

SCOS is a self contained program development system for any 8080 system which uses Processor Technology Corp. SOLOS / CUTER monitor. Included in SCOS is an executive to handle memory files, an assembler, an editor, and cassette storage routines.

To use the system 6k of memory must be available for use by the system. This memory is allocated as follows:

0000 - FFF Operating Program
1000 - 10FF Special System RAM
1100 - 17FF Symbol Table ASSEMBLER only)

In addition, other memory must be available for source and object files necessary for the users programs.

Below is a list of the calls made to SOLOS. Note— At present, these calls are made to SOLOS residing at COOOH.

Loc.	Inst.
OE26	Call Sout
OE36	Call Sinp
OE3D	'80H - This defines key to abort listings (MODE on SOL20's)
OE41	Call Sinp
O5B5	Call WRBLK
OE96	Call RDBLK
OF24	Location of speed constant for SOLOS VDM driver
008E	Call Sinp
0032	'- key to abort current input line
0065	Backspace character - now set at 5FH

EXECUTIVE COMMANDS:

ENTR	Enter data to memory
DUMP	Display memory data
FILE	Create, assign or display file information
EXEC	Execute a program
ASSM	Assemble a source file to object code
LIST	List a file - with line numbers
PRIN	List a file - without line numbers
DELT	Delete lines of file
PAGE	Move a page (256 bytes) of data
SPEE	Set display speed
SSAV	Save the current source file
OSAV	Save an object file
SLOA	Load a source file
OLOA	Load an object file
OLOADX	Load an object file and execute it
RENU	Renumber the current source file
VERI	Verify the current source file for integrity

COMMAND FORMAT

RENUM

Renumbers the current file, as shown by the file command, in increments of 10 starting at 10.

VERIFY

Partially verifies the contents of the current file. It checks: 1) that every line has a valid 4 byte as cii line number; 2) that all the characters in the line are valid printable ascii characters (from 20H to 7FH); that each line is terminated by an ascii carriage return; 4) that the line length stored as the first byte of each line points to C / R at the end of the line; 5) that a valid end-of-file (01H) is found as a line length. The verify command does n-o-t check that the information in the file table is correct, only that the source file is valid.

SSAV / NAME /

Source save - Saves the current source file on cassette. Name can be any 1 to 5 character value. If less than 5 char., the remaining characters are filled in as blanks. Note- the name used in this command does not have to be the same name as used when the file command was given.

Continued on Page 2 . . .

FEEDBACK: September Meeting

by Tom Dunsheath

Well, this is FEEDBACK No. 2, intended for those of you who could not attend the September 18th CACHE meeting, or who were too busy gawking at the exhibits by Chicagoland computer stores, or who were otherwise socially or economically engaged.

SOME OF THE DETAILS

Bill Precht opened the meeting promptly, and outlined future meetings:

—October 16 — More Videographics
—November 20 — Heathkit, with a little if, i.e., if they will come.
—December 18 — Christmas Party
—January 16 — Repairs

Bill also reminded the members of the Triton Personal Computing Conference, and offered brochures.

Several group purchases are planned:

—Printer, Practical · Automation, 8½ x 11, 120 characters per second, 132 characters per line, upper case only, RS 232C, $600.00 in groups of ten, thirty days delivery. If interested, see Bill Precht.

—Hayes Communication Adapter Kit, S-100 bus modem, 100 to 300 baud, $150 less 20 per cent in groups of ten. If interested, see Dave Jaffee.

—Superscope Cassette Recorder, $73.47 in groups of ten. If interested, see Dick Gerlach.

Cliff Barber is still looking for writers, photographers and articles to contribute to the exapnded **CACHE coverage in the Electronics Journal.**

If you want to leave a message for anyone, remember the hot line, (312) 849-1132.

With business out of the way, Bill Precht introduced Sheldon Epstein, who chaired the panel discussion.

Sheldon recounted the growth in rocketry over the last 50 years, Von Braun to the Space Shuttle, and the parallel growth in computers over the past 30 years, ENIAC to the Home Computer. He offered the estimate that there are now about 50,000 of the latter in place. And he pointed out the analogy between a Home and a Computer; a home is a hole in the ground that you fill with money; a computer is a box in which you do the same.

Sheldon introduced the panel members: Ed Cooper of American Microprocessors, Keith Cook of Computerland Arlington Hts., John Clark of Data Domain, Chuck Faso of Computerland Niles, Ed Curtiss of Lillipute Computer Mart, Nabih Mangouvi of Nabih's Inc., Ron Hummell of Heathkit Lincolnwood, Gerry Koppel of AAA Chicago Computer Center, Larry Sipovic of Semiconductor Specialists, Bill Rotenberry of Midwest Microcomputers, Fred Prehn of Heathkit Downers Grove, Roy Emerson of Bits and Bytes and David Vominah of Quantum Computer Works.

As indicated previously, the interaction between the audience and the panel ranged over matters far and wide: delivery time (with much laughter), maintenance policies, mathematical floating point packages, floating point boards, graphics, the need for a solid power supply, a BASIC compiler, providing FORTRAN to an existing ASR terminal, and so forth.

As also noted previously, the panel discussion ranged so broadly as to defy this writer's ability in reportage. Unlike FEEDBACK No. 1, where the presentation of the Bally Arcade opened up a few idea loops, and then neatly brought them to a stable closed loop condition, the panel discussion precludes such analysis. One is tempted to name this column "Difusion" or "Osmosis". It's not that the subject was complex, with multi-variate loops and interrelationships; it's simply that no overall structure was apparent.

But the panel discussion did truly reflect the state of the art in Personal Computing; varigated, differentiated, vigorous, growing, lusty, innovative, exciting, novel and even revolutionary. And as a true reflection of the state of the art, the panel discussion was very worthwhile.

But did the panel discussion deliver what it promised? It was, after all, entitled "Selecting Systems" and sub-titled "How do you get Started?" If the discussion was intended as an outreach to the novice, it probably was a failure. Very likely, those who came in the door with these questions went out the same door with still the same questions.

Which recalls the words of wisdom of my Grandmother, "Work begun is half done." And the words of Cassius Longinus when his plans against the Romans did not succeed, "In great attempts, it is glorious even to fail."

Building The SOROQ IQ 120

by Charles Douds

The SOROQ IQ 120 video terminal looked good to me, so I ordered one. I wanted the 80-character wide line with upper and lower coase for a manuscript preparation system. Of course, I would need at least 24 lines for that. I would be nicer to have enough lines to show a full typed page and to do it all with 7x9 characters with descenders, but I couldn't afford a display as fancy as that. After all, 24 lines is nearly half a typed page and the funny looking lower case g,j,p,q,y (all the characters that descend below the line) will appear properly when they are actually typed out!

I was delighted to find that the SOROC "kit" is only partly a kit. Everything, and I mean everything, comes completely assembled except for the main circuit board. All the nasty power supply, wiring harness, and tedious keyboard assembly is done. absolutely all the mechanical assembly - which I usually hate most of all because they never seem to get the holes so that they line up exactly right - is done.

Just take off the bottom plate and in the inch deep space there is the circuit board and all the parts. That 13½ x 14 board certainly looked like a monster compared to the S-100 boards we are used to.

After reading the instruction manual, which I found to be generally well written, I counted parts . . . opened up the packing list as I should have done in the first place . . . and found that the 24 line option and lower case option parts were backordered. Almost everything else was there. I had two extra 14-pin IC sockets and was missing one 16-pin socket. Fair enough trade with a company whose business is suddenly booming. I suspect under such circumstances it is not too surprising if there is an error once in a while.

Continued on Page 4 . . .

. . . from Page 1

SLOA / NAME /

Source load - Scans the tape for the file name given above. When the named file is found, the source file is read into memory. At the end, the file table is updated so the source program may be edited. If a checksum error occurs, the message 'TAPE ERROR' will be printed. If the MODE SELECT key is depressed while reading the file, a tape error will also be invoked.

OSAV / NAME / SSSS EEEE

Object save - Saves an image of memory from SSSS thru EEEE on tape. The auto-execute address is set to SSSS.

OLOAD / NAME / SSSS

Object load - Loads an object file into memory at address SSSS. If a tape error occurs, a 'TAPE ERROR' message will print out.

OLOADX / NAME / SSSS

Same as OLOAD except that after loading the file, the program is executed at the address which was entered as the load address ssss.

SPEED XX

Sets the display speed of the VDM Driver. XX can be any value between 0 and FF (HEX). 0 is the fastest and FF is slowest.

PRINT

Does the same as list except that line numbers are not displayed. Useful for writing letters.

Miscellaneous Comments

The SCOS assembler can reference A, B, C, D, E, H, L, and M. However, PSW and SP are not built in. Both of them represent 6, so you may PUSH 6 or DAD 6 instead of PUSH PSW or DAD SP. If you wish, you can specify: SP EQU 6 and PSW EQU 6. This will allow you to use SP and PSW later in the program.

If you should want to restart your machine and re-enter SCOS without wiping out the file table, you can re-enter at location OOOCH instead of location 0.

When executing an object file, you can return to SCOS by merely executing a RET instruction as long as the stack is not messed up. If it is, JMP 001FH instead.

Some useful routines that you might want to use are listed below:

Loc. at	Routine
OOOCH	Return to SCOS
OO8EH	Wait, read SINP into A and B
009CH	Send Char. in A to SOUT
01D3H	Print A in Hex (clobbers BC, and HL)
01E3H	As above, but print space after hex.

The following is an explanation of the commands which are supplied with Software Package No. 1, which are carried thru into SCOS. It is recommended that users obtain a source listing of Package No. 1 if they desire to modify it. Included on the tape that supplies SCOS is the source code for the modifications made to Package No. 1. Package No. 1 was originated by Processor Technology Corporation, and originally modified by Ward Christensian.

The executive has one error message . . . WHAT? . . . indicating an improper command or an error on parameters following the command

COMMAND FORMAT

ENTR AAAA – Enter data to memory

This command is used to enter data to memory starting at address AAAA and continuing until a return command (/) is given. Data is entered in hexidecimal format.

Example:

ENTR 500
0 0A 30 44 FF FE /

Continued on Page 3 . . .

SUBSCRIPTION INFORMATION

If you're a subscriber to the CACHE Register, then check the upper right hand corner of your mailing label; the code listed there indicates your subscription status.

The possible codes, as well as descriptions are listed below:

YYMM	The first two numbers of the four digit code on the right of the first line indicate the year that your subscription was entered most recently. The second two digits indicate the month of that year. Your subscription expires one year from that date.
FREE	Free subscription to another group which sends us their newsletter.

NEW MEMBERSHIPS

CACHE is a not-for-profit organization dedicated to investigating the roles and uses of microcomputers, and related small-size computing devices, in the hobbyist field.

CACHE is always interested in acquiring new members who share our interest in microcomputers, who need assistance with their own devices, or who are simply interested in what these new-fangled devices are.

If you would like additional information, write to CACHE in care of the P.O. box number shown on the front of this Register.

Or better yet, send in $10, check or money order, and you'll receive one issue of the Register per month for the next year.

Be sure to include your name, address, phone number and zip code.

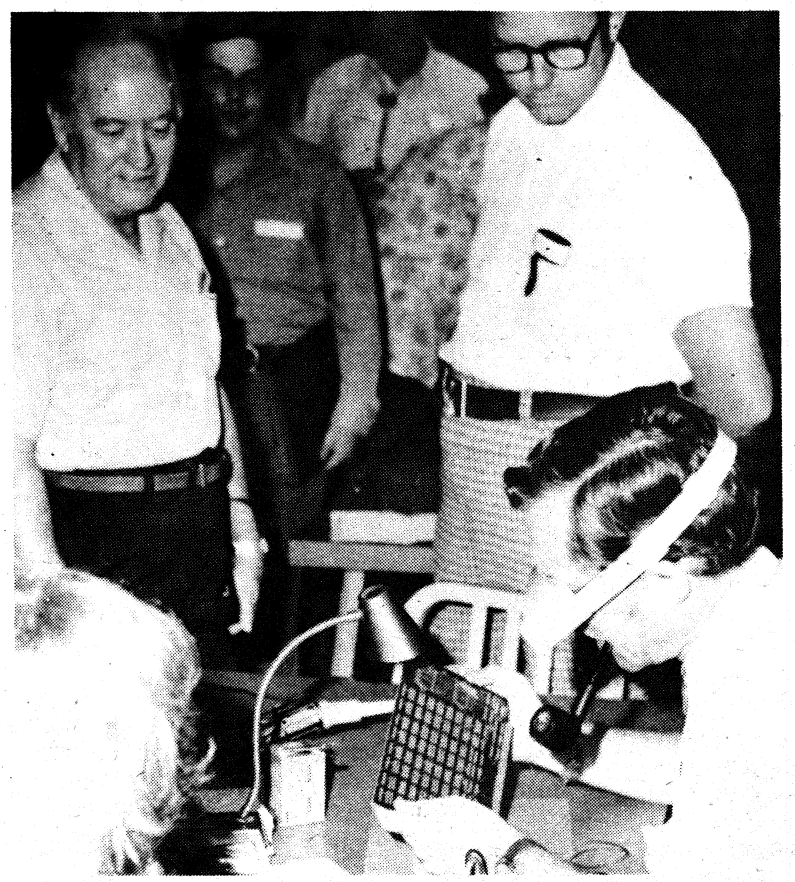
You might even include a note telling us how you heard of CACHE; whether you own a system (and what the system is, if you do); ideas or comments on the Register, meetings, organization, etc.; whether you need or can offer assistance in any area, e.g. software, interfacing with the real world, et al.; and so on.

FUTURE MEETINGS

Nov. 20th —
We are hoping to have the people from *Heath* demonstrating the new Heathkit computers.

Dec. 18th
Our annual *Family Computer Christmas Party.*

Jan. 15th
The theme here will be *"Get 'Em Going"* with the emphasis on repair of ailing bombs. Bring your sick computer to the clinic.



Demonstrations at the monthly meetings conducted by CACHE tend toward the practical and informative. People interested in computers can usually find others in attendance of similar skill levels. Novices find those who can explain the elementary items, while advanced electronics and programming experts exchange ideas over the more esoteric concepts of the field. CACHE has more than 380 members and the monthly meetings are well attended. Shown in the photo is Chuck Douds who demonstrated the proper techniques for soldering a memory board at a recent meeting. Douds also authored the article on page 4 of this CACHE Register on his successful attempt to build a SOROQ IQ 120 Video Terminal. If you have attempted a similar project we'd like to hear from you. Put it in writing and mail it to us or bring it with you to the next monthly meeting.

Editorial

Personal computers are "an idea whose time has come". It has only been three years since the advent of the Micro 8 and the Altair. This month's issues of **popular Mechanics, Popular Science,** and **Mechanix Illustrated** all feature personal computers on their covers. These publications also signal another evolutionary step in the growth of the micros; the **Pet** and **Radio Shack**'s package are "ready to go" and will spread the number of homes with computers quickly. As the author states, "Major inventions have always changed our lives and society in ways that could not be predicted."

We especially want to thank Geoff Lowe, Ward Christensen and Chuck Douds for their help in getting out the interim issues. Let us hear from you, we welcome your comments and suggestions.

New Books

Your Home Computer
James White 211pp
Published by Dymax, Menlo Park, Calif. 1977

A good starting point for the newcomer to home computers, this overview by James White includes a basic introduction, a guide to selection of micros, a listing of computer stores, computer clubs, computer vendors, and a large section on applications.

This last section involves computers in fine arts, games, home office record keeping, and hobbies.

The author thoughtfully keys each area of interest with the name and address of people in that particular field.

CACHE has two members mentioned; Chuck Douds for model railroading and Ward Christensen on word processing.

If you've a friend who's just seen his first micro, knows nothing about them and now wants to know more, this is the book!

It's in our CACHE library now, courtesy of the author.

New Stores

As you read last month, NEW stores are springing up everywhere. Two new ones on the northwest side are Computerland on Rand Road, just north of Palatine Road and Data Domain in the Plaza de Las Flores in Schaumburg.

Computerland is part of the national group featuring Imsai and other computers, while Data Domain is allied with the Bloomington, Indiana store of the same name.

We visited Computerland shortly after they opened and chatted with John Gibbs and Keith Cook. They intend to offer complete systems including software and service. Lines offered in addition to Imsai are Cromenco, Apple, Polymorphic, TDL, and Mini-Mag. They plan to have a full line of tools, TTL chips, etc. as their inventory expands.

At Data Domain, John Clark and his wife share the duty; carrying a full line of magazines, books, kits, and of course, computers such as Apple, Vector, Alpha-Micro, etc.

Drop in at either one of these new stores, you'll find a warm welcome and knowledgeable folks manning both.

Cliff Barber

OCTOBER 16 MEETING

Date: Oct. 16

Videographic Techniques
Ron and Joe Lachman of the Univ. of Illinois, Circle Campus

Place: Northern Illinois Gas Building
1 block north of the corner of
Golf and Shermer Roads,
Glenview, IL.

Time: 12:00 Doors open
1:30 General meeting
2:00 Featured topic

. . . from Page 2

DUMP AAAA BBBB — Dump Contents of Memory

This command is used to examine the contents of memory. The values contained in memory from locations AAAA to BBBB are displayed in hexidecimal. Each line of display consists of an address followed by the contents of the next 16 memory locations. If BBBB is not specified only location AAAA will be displayed.

FILE / NAME / AAAA

This command is used to enter, examine, or modify parameters of files created in the system. Up to six files can exist simultaneously with any one of the files called as "current". Depending on the form of the command and following parameters the following functions are performed.

Create a file with the name, NAME starting at address ADDR and make it current. If a file with the same name already exists output error message NO NO.

FILE / NAME / 0

Delete file with name NAME and make no file current. Note no file can start at ADDR 0.

FILE / NAME /

Get file NAME and make it current. Save all parameters of existing current file.

FILE

Display parameters of the "current" file in the following format with AAAA and BBBB being the beginning of file and end of file addresses:

NAME AAAA BBBB

FILES

Display the parameters of all files currently saved by the system.

EXEC AAAA Execute a program

This command is used to execute a program at address AAAA.

LIST N List file

..This command is used to display the lines entered by the user into the file. The output consists of the lines in the file starting at line number N. If N is not specified the display starts at the beginning of the file. The user can terminate the display by using sense switch seven on the front panel.

DELT L1 L2 Delete line (s) from file

This command is used to delete lines entered by the user from the file. All lines starting at line L1 and continuing up to and including L2 are deleted from the file. If L2 is not specified only L1 is deleted.

PAGE AAAA BBBB Move page of data

This command is used to move one page (256 bytes) of data from address AAAA to BBBB.

ASSM (E) AAAA BBBB Assemble a source file to object code.

This command is used to assemble a source program written by the user and located in the file area. The assembler performs the assembly, assigning addresses to the object code starting at AAAA. On the second Pass the object code is placed in memory starting at location BBBB. If BBBB is not specified it assumes the same value as AAAA. During pass one certain errors are displayed and dyring pass 2 a complete listing is produced. If the optional E is specified in the command only those lines which contain errors are listed.

EDITOR

The editor is a line oriented editor which enables the user to easily create program files in the system. Each line is prefaced by a fixed line number which provides for stable line referencing. Since line numbers can range from 0000 to 9999 (Decimal) there are 10,000 lines that can exist in each file. (If enough storage exists.) As the user types lines on the input device they are entered into the file area. The editor places all line numbers in sequence and automatically overwrites an existing line in the file if a new line with the same line number is entered by the user. A feature of the editor is that the file area never contains any wasted space.

The Editor **ALWAYS** operates on the current file.

The editor does not automatically assign line numbers. The uer must first, when entering a line of data, enter a decimal number which will be interpreted as being the line number. Valid line numbers must contain four digits . . . preceding zeros must be included. An entry to the editor is terminated by the carriage return key. No more than 80 characters may be input for one line.

All lines are ordered by the ascending numeric sequence of their line numbers. If the user wishes to insert lines after the initial entry is made it is suggested that he input the original lines with line numbers at least five units difference.

ASSEMBLER

When the Assembler is given contrll by the executive it proceeds to translate the Symbolic 8080 Assembly Language (Source) program into 8080 machine (object) code. The Assembler is a two pass assembler which operates on the "current" file. Features of the Assembler include:

free format source input
symbolic addressing, including forward
references and relative symbolic references.
complex expressions may be used as arguments
self defining constants
multiple constant forms
up to 256 five character symbolss
reserved names for 8080 registers
ASCII character code generation
6 Pseudo Operations (assembler directives)

The assembler translates those lines contained in the current file into object code. The second character following the line number is considered to be the first source code character position. Hence the character immediately following the line number should normally be blank. Line numbers are not processed by the assembler they are merely reproduced on the listing.

The assembler will assemble a source program file composed of STATEMENTS, COMMENTS, and PSEUDO OPERATIONS.

During Pass 1, the assembler allocates all storage necessary for the translated program and defines the values of all symbols used, by creating a symbol table. The storage allocated for the object code will begin at the first byte dictated by the 1st parameter in the original Executive ASSM command.

During pass 2, all expressions, symbols and ASCII constants are evaluated to absolute values and are placed in allocated memory in the appropriate locations. The listing, also produced during pass 2, indicates exactly what data is in each location of memory.

STATEMENTS may contain either symbolic 8080 machine instructions or pseudo-ops. The structure of such a statement is:

NAME OPERATION OPERAND COMMENT

The **name field,** if present, must begin in assembler character position one. The symbol in the name field can contain as many characters as the user wants, however only the first 5 characters are used in the symbol table to uniquely define a symbol. All symbols in this field must begin with an alphabetic character and may contain no special characters.

The **operation field,** contains either a 8080 operation mnemonic or a system pseudo-operation code.

The **operand field** contains parameters pertaining to the operation in the operation field. If two arguments are present they must be separated by a comma. Example:

0015 FLOP MOV M,B COMMENT
0020 COMMENT
0025 JMP BEG
0030 CAKK FKIO
0035 BEG ADI 8+6-4
0040 MOV A,B

All fields are separated and distinguished from one another by the presence of one or more blank characters. (Spaces)

The **comment field** is for explanatory remarks. It is reproduced on the listing without processing. See example 0015. Comment lines must start with an asterisk ( ) in character position 1. See example 0020.

SYMBOLIC NAMES

To assign a symbolic name to a statement one merely places the symbol in the **name field.** To leave off the name field the user skips two or more spaces after the line number and begins the operation field. If a name is attached to a statement, the assembler assigns it the value of the current Location Counter. The Location Counter always holds the address of the next byte to be assembled. The only exception to this is the EQU pseudo-op. In this case a symbol in the **name field** is assigned a value which is contained in the **operand field** of the EQU psuedo-op statement. Example:

0057 POTTS EQU 128

assigns the value 128 to the name POTTS. This data can then be used elsewhere in the program as: eg ADI POTTS.

Names are defined when they appear in the name field. All defined names may be used as symbolic arguments in the argument field. See examples 0015, 0025, 0030, 0035.

In addition to user defined names, the assembler has reserved several symbols, the value of which is predetermined. These names may not be used by the user except in the operand field. They are (with their value in parenthesis):

A-the accumulator (7)
B-Register B (0)
C-Register C (1)
D-Register D (2)
E-Register E (3)
H-Register H (4)
L-Register L (5)
M-Memory (through H,L) (6)

In addition to the above reserved symbols, there is the single special character symbol ($). This symbol changes in value as the assembly progresses. It is always equated with the value of the program counter after the current instruction is assembled. It may only be used in the operand field. Examples:

JMP $ means jump to the next location
MOV A,B, after this instruction; i.e., the
MOV instruction.
LDA -+5
DB 0
DB 1 means load the data at the fifth location
DB 2 after this location. In this case the data has
DB 3 the value 5.
DB 4
DB 5

RELATIVE SYMBOLIC ADDRESSING

If the name of a particular location is known, a nearby location may be specified using the known name and a numeric offset. Example:

JMP BEG
JPE BEG+4
CC SUB
Call $+4B
BEG MOV A,B
HALT
MVI C, 'B'
INR B

In this example the instruction JMP BEG refers to the MOV A,B instruction. The instruction JPE BEG+4 refers to the INR B instruction. BEG+4 means the address BEG plus four bytes. This form of addressing can be used to locate several bytes before or after a named location.

CONSTANTS

The Assembler allows the user to write positive or negative numbers directly in a statement. They will be regarded as decimal constants and their binary equivalents will be used appropriately. All unsigned numbers are considered positive. Decimal constants can be defined using the descriptor "D" after the numeric value. (Not required, default is decimal).

Hexadecimal constants may be defined using the descriptor "H" after a numeric value. IE. +10H, 10H, 3AH, OF4H.

Note that a hexadecimal constant **cannot** start with the digits A-F. in this case a leading 0 must be included. This enables the assembler to differentiate between a numeric value and a symbol.

ASCII constants may be defined by enclosing the ASCII character within single quote marks, i.e., 'C'. For double word constants two characters may be defined within one quote string.

EXPRESSIONS

An expression is a sequence of one or more symbols, constants or other expressions separated by the arithmetic operators plus or minus.

PAM +3
ISAB-'A'+52
LOOP+32H-5

Expressions are calculated using 16 bit arithmetic. All arithmetic is done modulo 65536. Single byte data cannot contain a value greater than 255 or less than -256. Any value outside this range will result in an assembler error.

PSEUDO-OPERATIONS

The pseudo-operations are written as ordinary statements, but they direct the assembler to perform certain functions which do not always develop 8080 machine code. The following pages describe the psuedo-ops.

ORG — Set Program Origin: label ORG expression
Where the label is optional, but if present will be equated to the given expression.

END — End of Assembly: This pseudo-op informs the assembler that the last source statement has been read. The assembler will then start on pass 2 or terminate the assembly and pass control back to the executive. This pseudo-op is not needed when assembling from a memory file since the assembler will stop when an end of file indicator has been reached.

EQU — Equate Symbolic Value; The EQU is used to make two symbols equivalent in value; Label EQU expression

Where: Label — is a symbol the value of which will be determined from the expression. Expression - is an expression which when evaluated will be assigned to the symbol given in the name field.

DS—Define Storage— The DS causes the awsembler to advance the Assembly Program Counter, effectively skipping past a given number of memory bytes.

label DS expression

DB—Define Byte; This pseudo-op is used to reserve one byte of storage. The content of the byte is specified in the argument field.

label DB expression

DW—Define Word; This pseudo-op is used to define two bytes of storage. The evaluated argument will be placed in the two bytes; high order 8 bits in the low order byte, and the low order 8 bits in the high order byte. This conforms to the Intel format for two byte addresses.

ASSEMBLER ERRORS

The following error flags are output on the seembler listing when the error occurs. Some of the errors are only output during pass 1.

O Opcode Error
L Label Error
D Duplicate Label Error
M Missing Label Error
V Value Error
U Undefined Symbol
S Syntax Error
R Register Error
A Argument Error.

LIST OF SOLOS / CUTER COMPATIBLE SOFTWARE

All software is recorded at 1200 Baud using the standard CUTS format as described in the SOLOS / CUTER Users manual.

Side No. 1
File Name Remarks
SCOS
..A modified version of Proc. Tech's Software Pkg. No. 1
/ MODS1 /
..Source for mods made to Software Pkg. No. 1.
/ 12KBM /
..Source for mods made to 12k BASIC 3.2 Nop's STA's and SHLD's that modify 1 / 0 constnatly. Also has mods to use SOLOS I / O and cassette routines.
/ KALEI /
..Genrates kaliedoscope patterns using alphabet.
/ TRAIN /
..Source for program printed in PTC's ACCESS newsletter. It puts a picture of a train on screen.
/ SPLAT /
You control a spaceship and try to avoid hitting surrounding stars. Also has demo mode.
/ IDUMP /
ASCII Dump routine. Uses SCOS.
/ CONDA /
Conditional Assembler (allows IF ENDIF statements)
/ LIFE /
Based on a cell growth game in **Scientific American.**
/ PTSAV /
Paper tape save in INTEL format. No CKSUM

Continued on Page 4 . . .

. . . from Page 3

/ SIMUL /
Used to debug a program. You can simulate a running program. It will trace program execution, display all registers.
/ TIMSQ /
Turns VDM into a Times Square like display.
/ TSLIC /
Routine to allow time slicing between CCOS and BASIC. This routine needs interupts.
/ DLIFE /
Dazzler life program.
/ SYMPER /
Sorts and prints symbol table for assembler.
/ EXTND /
This program converts 8k 3.1 MITS BASIC programs to 12k 3.2 format.
/ EDIT /
This routine allows test editing to be done to SCOS files. Allows cursor movement.
/ CCOST /
Source for Tarbell CCOS. These are mods used in RC-COS.
/ CHANG /
String search and change for source files.
/ DISAS /
Disassembler
/ TBLEX /
Extends symbol table for assembler
/ DRAW /
Draws pictures using an analog output port.

LIST OF SOLOS / CUTER COMPATIBLE SOFTWARE

All software is recorded at 1200 Baud using the standard CUTS format as described in the SOLOS / CUTER Users Manual.
Side No. 2

File Name	Remarks
12KSB	12k Basic. Has 12KBM installed for patches at 5F00.
HELLO	
LUNAR	Lunar lander program
BOMBR	
CHMST	Mad chemist program
CHIEF	
GUNNR	
FLIP	
CHASE	
SNARK	
STREK	Startrek
HURKL	
TRAP	
DEPTH	
ZAP	
DRAG	

. . . from Page 1

It wasn't hard to complete the board - it just that it takes some time to get 134 IC sockets soldered in without errors. It is a good idea, I find, to do all the 16-pin sockets before the 14-pin. After all, a 14-pin socket will fit into a 16-pin socket's place, but not vice bersa. Because of feed-through holes and other such things it's not always obvious that you are making a mistake if you start with the smaller sockets.

I especially liked the white IC sockets. Later on they made it much easier to check that no IC legs had gotton folded under or were hanging out over the edge of the socket.

I usually do about four sockets at a time, then check with my magnifying glass to make sure that I haven't missed soldering a pin or cold soldered one.

The assembly instructions are well-illustrated, showing you where each resistor, capacitor, and all other components go.

However, the assembly instructions do not give good detail about the power on the board. An additional drawing showing the on-board regulators would help as well as one giving all signals on all the jacks. This would be a great help in checking out the board.

efore putting in the ICs, I like to check that the proper voltages are present. The assembly instructions go for the "smoke test." Put everything together and plug it in. Much better, in my opinion, to check the voltages before installing the ICs. With the help of IC manuals I was able to do this myself and everything seemed okey.

I spent the rest of an evening installing ICs. Since it had those nice white sockets, I took the time to get out my quill india ink pen and write the IC type numbers on all the sockets. (Three 28-pin sockets that hold custom programmed "PLAs" were black.)

Incidentally, I always use an IC insertion tool to put in 14 and 16 pin ICs. This is a great help in avoiding trouble with bent-under legs. But still, I always check for bent IC legs. I fill the board from the center a row at a time and work out to the edge. That way after I do one row, I can put a white card behind it and using my Tensor light I can sight under the ICs and along the edge of the socket to make sure that no legs are bent under or sticking out over the edge. Again, I like the sockets SOROC selected.

They have a small rib along the outside that helps to ensure that the legs go in properly. All the ICs went in properly and with the help of my daughter we "proff-read" all the IC numbers. This provided an opportunity to make sure that all the ICs were inserted right way up. (All ICs do face the same direction on this board - none of those nasty situations where a few ICs have pin 1 at the opposite corner to all the others.)

Then it was time to insert the 40-pin UART and three 28-pin PLAs by hand. Iget the legs bent just enough so that the IC will go straight into the socket without too much messing around by holding the IC onedge against the table and pressing down on it with a small steel block I happen to have. I suppose a piece of hardwood would also do the job.

While the SOROC IQ-120 doesn't have a microprocessor, it is smart. It gets its intelligence from programmed logic arrays - the "PLAs". These chips are programmed in a manner similar to the 1702 PROMs, but instead of programming memory cells, connections between sets of AND and OR gates are programmed. The end result is that a board can operate very much like a fixed program, process control computer. The PLAs give the terminal a great degree of versatility.

While putting the chips in I found that a major chip manufactaurer had fouled up. The chip marked 74166 had 14 pins. Its socket had 16 pins.

My IC manual said a 84166 had 16 pins. Apparently someone in the IC factory had put the wrong label plate in the automatic production line. Strange things can sometimes happen even in the best of families.

With the 24-line and lower case options backordered, I put in the two jumpers needed for 12-line and uppercase only operation. I suppose I could have omitted the last one for there is a key on the keyboard marked "alpha" that gives uppercase only on the alphabet keys. Depressing that key makes the terminal function in a teletpye-like manner.

Up to this point I had spent about 12 hours. No speedy, but I had double or triple-checked everything as I went along. Troubleshooting can consume far more time.

Time to install the board. It snaps onto five plastic clips on the bottom plate. It dawned on me that the reason that plate was so heavy and the whole terminal has such a solid feel - it weighs 40 pounds - was that the chassis is made from heavily plated steel.

I switched it into half duplex so it would write on the screen as I keyed it, turned on the power, and heard a beep from the speaker just as the assembly manual siad I should. Looking at the schematics later I discovered that this meant the crystal frequency divider chain was working properly since the "bell" tone is derived from one of the vertical scan outputs. Not only that, but a cursor appeared on the screen of the Ball monitor (one of the companies IBM buys their monitors from) as soon as it warmed up.

And I could type "The quick brown fox jumped over the lasy dog's back." Hitting carriage return, the cursor - a solid block - jumped back to the beginning of the line as it should and there was the T showing through the cursor in inverse video. But how do I get the cursor down to the next line? There was no Line Feed key. I tried the down-arrow cursor control key and the cursor jumped down a line. Actually two lines. In the 12-line display mode it writes on every other line. It does what it is supposed to, but it is irritating and slows down typing to not have a Line Feed key next to the carriage return. The CR and cursor-down keys are uncomfortably far apart.

I tried the number keys and the numeric keypad keys and they all worked as they were supposed to. The numeric pad is a standard feature, not an extra price option.

But what about those special keys? "Clear" sent the cursor home and cleared the screen as it was supposed to. But neither the other three cursor control keys nor the Home or Tab keys did anything. And waht was that I had been smelling? Hmmm, maybe it would be a good idea to look at the circuit board. There it was, the second PLA had a strange looking color. The reason was immediately obvious. I had put it in upside down - and after all that checking! I had bombed one of the four custom programmed chips in the unit. All I can say is that when you double-check everything after midnight, you had better check it again the next morning. Another call to California and the PLA was received within a week.

In the meantime it was still possible to use the unit. I had noticed that the video display sometimes seemed to have a little "shimmy" in it. Everything remained quite readable, but things werent' crisp and sharp over the whole display. There would be a couple of lines that were fuzzy, and the starting point of the raster scan on the left would form a slowly wavering S-curve. Then about halfway down the screen there would be another couple of lines doing the same thing. This pattern would slowly move up the screen affecting one set of lines, then another.

Randy Suess tracked this one down. First we tried to adjust the horizontal hold, but that control doesn't exist. As the manual points out, its not needed when the viedo signal, horizontal sync, and vertical sync are not combined into a composite video signal. We're not dealing with a modified TV circuit as is the case in most monitors.

Turning up the brightness control until the raster was visible, it was easy to see the irregular left margin and the "fuzz." Then using a scope we saw a fuzz on the +5 volt line. It was about a 0.2 volt oscillation that lasted for about a millisecond and occurred every 8.3 msec.

Now 8.3 msec is one of those suspicious numbers when trouble shooting. It is the period of 1 / 120 second and that is the time betwee nthe peaks of full wave rectified 60 cycle power line. In other words, this number indicates something to do with the poer supply. It may seem surprising that a power supply can be an oscillator, but it happens. One reason I gave up working with transistor circuits was that every time I tried to design anamplifier I got an oscillator, and the best my oscillators would do was to amplify.

There isn't very much to the SOROC power supply. It is the vertical board to the right of the picture tube. The +5 and +15 volt IC regulators are mounted on a heavy aluminum heat sink. After looking it over, Randy reached for a 0.1 uf disk capacitor and touched its leads across the 0.1 uf capacitor that is in the upper front corner of the board. Immediately the raster straightened up and the oscillation disappeared from the scope trace. Since there were nice broad copper runs on the power supply board, it was only a moment's work to solder the capacitor in place.

I have seen another SOROC that has a similar problem. It's such an easy cure, there is no need to live with it even if you ordinarily stay away from anything other than digital circuits as I try to do.

After tweaking the focus adjustment we had a clean, crisp display. It has remained that way and been a real pleasure to work with.

My remaining problem was to interface the SOROC with my Digital Group system. The SOROC has a little switch on the back that can select any one of 15 standard baud rates from 75 to 19200 for its serial output. (Parallel output is not available.) If I had a I / O board with a UART — such as the Processor Technology 3P+S — it would be only a few minutes work to complete the interface. I intend to build a I / O board for the Digital Group system with three serial ports (terminal, printer and modem), but since it didn't exist at the moment, I decided to build a software UART.

The Digital Group software provides such a "UART". But maybe I should say "UAT" since it is a transmitter only. I found I had a 1488 chip to convert from the TTL output of the I / O board to the RS-232C signal the terminal needs as input. I plugged it into my Continental Speialty Co. breadboarding socket and brought out +12, -12, and ground lines from the power supply.

I set the terminal for 110 baud and half duplex mode; no need for it to "echo back" the character it receives when the transmission line is only a few feet long. After loading MAXI-BASIC and selecting "TTY" output, the video was appearing on the terminal as well as the monitor. Since 110 baud is rather slow, I decided to try for a faster rate. The timing constant is located at address 066FH. Its value is 1D for 110 baud. By trial and error I found that it was 15H for 150 baud and 05H for 300 baud. Another constant in the program would have to be changed to get to higher baud rates.

Before doing that, I decided to learn how to write a program that would receive data from the terminal. That would be only the first step, of course, to get the terminal fully functional. I would also have to patch the programs so that where they had a "Call TV", they would "Call terminal" instead. I couldn't directly patch the TV subroutine since it was on a EROM. After I am satisfied that everything is doing what I want it to, I can get a modified EROM burned and go back to using the original programs.

I have described the details of my experience in getting my SOROC up. Along the way I have described a number of its features. It also has a others that make it very cost effective compared to many other terminals.

The cursor is completely controlled either from the keyboard or by the computer. The keyboard has the usual four keys to move the cursor up or down a line and to forespace or backspace it. Forespace moves the cursor ahead without erasing the character at the new position. By pressing the Escape key, the equal key, and then two other alphanumeric keys — according to a code given in the manual — the cursor can be positioned anywhere on the screen. These same -code sequences can be transmitted to the terminal from the computer (Escape is 1BH). There is also a key to send the cursor "home" into the upper left-hand corner as well as another key to clear the screen and home the cursor. During entry, after the screen is filled subsequent lines cause the lines on the screen to scroll up. The line that scrolls off the top of the screen is lost, but of course the data has already been transmitted to the computer so it is available there.

In addition to having half and full deplex transmission modes, it also has "block" mode where the whole screen is transmitted at once. But this is more than just a transmission mode. It allows you to put a "preprinted form" on the screen. A form for a business application might ask for name address, catalog number of an item, description, quantity, etc. These words can be put on the screen from the computer and "protected". When they are protected they appear at reduced intensity and cannot be written over or changed during data entry. As the cursor moves along, it automaticallly skips over them. The Tab key will send the cursor to the next item to be entered. By using the cursor control keys (up, down, backspace, and forespace) corrections can easily be made before the entire page is sent to the computer. As you might guess, there is no scrolling in block mode.

Any key will automatically repeat simply by holding it down for more than a half second. The time is long enough that it doesn't cause trouble with ordinary typing. This is especially convenient in moving the cursor around. My daughter has great fun drawing pictures on the screen — and the computer doesn't have to be on for this.

Other functions provided through the use of escape codes (press the Escape key and then press another key) are erase to end of line, erase to end of page, erase all unprotected character positions, and erase everything including protected characters. The last is needed to put a new form on the screen.