Dissecting C. W. Moser's ASSM/TED 1.0

Francis Turco

Carl Moser's excellent assembler/text editor for the 6502 Microprocessor has been reviewed superficially in several publications.[1,2] So far, no one has done an in-depth write-up for a PET owner who wants to understand or modify his copy. The manual provided by Moser is adequate, but sketchy in some areas. I, for one, would like to see some articles by users who have figured out solutions to problem areas.

For example, PET owners find out (on page 35 of the manual) that "At present, the ASSM/TED does not contain a printer subroutine...". In another area, the ASSM/TED is designed for a "standard" PET and utilizes the audio cassette drives for off-line storage. The manual (Sections 6 & 7) discusses configuring the ASSM/TED for disk operation and using it with disk. This discussion is too brief to be understandable by a novice assembly language programmer.

In still another area, the editor has many powerful capabilities and will accept a full line of characters (65 typed characters) but the sense of the shift key is reversed. That is, shift gives lower case letters. Unshifted gives upper case letters. This proves to be cumbersome when typing a letter or manuscript from the PET keyboard.

In an effort to shed some light for others, who like myself, are trying to understand and modify their copy of ASSM/TED and perhaps stimulate some of you to share your findings, I am submitting some areas that I have uncovered in Moser's Assembler.

Figure 1 shows a memory map of the assembler/text editor. The assembler is written for a 16K PET and fills almost all useable memory space. As the figure shows, the assembler and text editor are co-resident and occupy the space from $2000 thru $3FFF. Commodore's monitor occupies the area from $0400 thru $076C. This leaves enough memory for a relocatable file ($1F00 thru $1FFF), a lable file ($1800 thru $1EFF), and approximately 4K for user programs ($0770 thru $17FF).

Table I is a list of addresses of major routines. This is a fun table -- try some experimenting with it. For example, RUN 8390 will assemble your program. RUN 8390 LIST will assemble and list. RUN 8470 will print your program. Table II provides a list of addresses of the pseudo opcode routines, while Table III contains some interesting areas that will be helpful to someone modifying his assembler.

Carl Moser's ASSM/TED is a very good program and will allow the PET owner to convert his PET into a 6502 development station with a little effort on his part. If the PET is equipped with a line printer off the IEEE port, the owner can easily get around the first problem area and get a listing of his source code and/or his assembly. This subject will be treated in PART II of this article.

1. **Compute**, Fall 1979, p. 100, "6502 Macro Assembler and Text Editor SYM Version" by Harvey Herman
2. **The PAPER**, Vol. II, Issue 6, August 1979, "Relocating Macro Assembler/Text Editor 1.0 by R. Busdieker

Figure 1. ASSM/TED 1.0 Memory Map

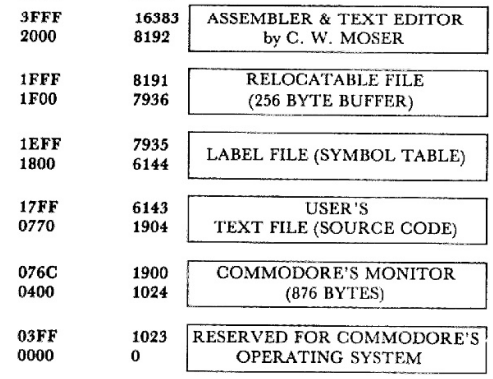

Table I

MAJOR ASSEMBLER ROUTINES

HEX	DEC	ROUTINE	
2033	8243	**CLEAR**	user's text file
208A	8330	**BREAK**	to monitor
2098	8344	**AUTO**	line number
20A0	8352	**GET**	program from tape
20A6	8358	**FORMAT**	text file
20B6	8374	**MANUSCRIPT**	line numbers output/ not output
20C6	8390	**ASSEMBLE**	source code
20FF	8447	**RUN**	program previously assembled
2116	8470	**PRINT**	text file
2AFB	11003	**OUTPUT**	create a relocatable object file
2E52	11858	**LABELS**	prints out label file
31EE	12782	**PASS**	execute the second pass of assembly
333E	13118	**NUMBER**	re-number text file
3467	13415	**PUT**	program out to tape
3559	13657	**FIND**	character string specified
355F	13663	**EDIT**	change source code
3844	14404	**HARD**	print routine (not functional on PET)
3873	14451	**COPY**	lines of text
39B9	14777	**MOVE**	lines of text

39C2	14786	DELETE	lines of text
39EF	14831	SET	boundaries of text file, label file & buffer
3A80	14976	DUPLICATE	files from tape 1 to tape 0
3AB6	15030	ENTER	file name in the diskette directory
3AC7	15047	LOOK UP	file name in the diskette directory
3B50	15184	SHIFT	upper/lower case

Table II

PSEUDO OPCODE ROUTINES

HEX	DEC	ROUTINE	
2919	10521	.DS	Designate Storage
2964	10596	.EJ	Eject
297B	10619	.RS	Resolve address & Store
2980	10624	.CE	Continue with Errors
2985	10629	.OS	Object Store option
298A	10634	.OC	Object store option Clear
298F	10639	.CT	Continues on Tape
2994	10644	.LS	List option Set
2999	10649	.LC	List Option Clear
299F	10655	.SI	Store Internal address
29A8	10664	.SE	Store External address
29B3	10675	.BA	Beginning Address
29F3	10739	.MC	Move Code
2A1D	10781	.BY	Bytes
2A57	10839	.DI	Designate Internal
2A60	10848	.DE	Designate External
2AB7	10935	.EN	End
3378	13176	.RC	Resolve Code
3D1E	15646	.ES	Output macro generated object code
3D23	15651	.EC	Supress macro generated object code
3D6A	15722	.MD	Macro Definition
3E0C	15884	.ME	Macro End

Table III

INTERESTING AREAS

HEX	DEC	ROUTINE
2000	8192	Cold start of ASSM/TED 1.0
203F	8255	Command Line Interpreter
207A	8314	Initializes Pointer for Text File
2090	8336	Warm start of ASSM/TED 1.0
2190	8592	Same as 8599 + carriage return
2197	8599	Prints out the double slash after listing
2602	9730	Reads remainder of entered command - For Example: PRINT 100 200 or FORMAT CLEAR
26AB	9899	Jump Table for Major Assembler Routines (Commands)
271C	10012	Pseudo Opcode Table
27AA	10154	Mnemonics Table
2E89	11913	Xfers Pointer for Lable File to Zero Page Initialize Pointer for Lable File
2F96	12182	Stores a Zero Pointer +2
32DB	13019	Prints character that is in accumulator (same function as 65490 in BASIC ROM)
330B	13067	Prints carriage return
331A	13082	Prints 2 spaces
331D	13085	Prints 1 space
3323	13091	Converts accumulator to Hex & prints it
354F-3558	13647-13656	Permanent Copy of Value of Boundaries for Text, Label & Buffer (See also 14889)
37E2	14306	Moser suggests this location for a JSR to a line printer routine written by the user. The routine at 13019 would call this subroutine.
3A29	14889	Prints out the boundaries & the present end of data (See also 13647)
3F00-3FFF	16128-16383	Relocated Page 1 variables
3F35-3F85	16181-16261	Keyboard Buffer