

Analysis of the Use of the 6502's Opcodes

Despite its remarkably sparse (151) set of opcodes, the 6502 CPU is considered to be more powerful than its cousin, the 6800. This is generally ascribed to its "post-indexed" addressing mode, represented mnemonically by "LDA (zpg),Y". This instruction computes the effective address of its operand by an unsigned addition of the Y register (8 bits) to a 2-byte zero page pointer (at zpg & zpg+1).

I have written a simple program that analyzes the Commodore PET's usage of the 6502 opcodes. I was particularly interested in the frequency of usage of the above type of indirection in this reasonably big piece of software. I had noticed in my own work that the other type of indirection on the 6502, "pre-indexed" indirection, was almost never of use and wondered whether this was just peculiar to my programming habits. The program written merely steps through the BASIC system, from its start at \$C000 to its end at \$FFFF, avoiding data tables, messages and other non-instructional areas, and counts the number of times each opcode is used. Here is what I find:

The PET BASIC interpreter occupies a little less than 14K of instruction memory. It consists of 6355 instructions, give or take a few. The "average" instruction is thus about two bytes long. The 6502 CPU possesses 151 discrete one-byte opcodes; the remaining 105 are not implemented. Their frequency of use in the PET ranges from very heavy (the JSR instruction itself accounts for more than 10% of the total) to not at all. Table 1 summarizes the situation for those twenty percent of the instructions that are most frequently used. We see that

- 1) The 80/20 rule is followed closely. Thirty opcodes (20% of the total) account for 5131 (81%) of the total 6355 used.
- 2) There are only three instructions in Table 1 that employ 16-bit operands. Two of these, JSR and JMP, have no substitute on the 6502. The third is store accumulator, STA adr16, but it

occurs only 72 times. Its complement, LDA adr16, is interestingly used only 38 times. Compare this with the zero page STA and LDA instructions that are used 511 and 437 times respectively. If we assume there are about 13 subroutines (136 RTS's), the "average" subroutine is called five times (684 JSR's).

- 3) The most popular instructions (#3-20) are described fairly well by:

$$\text{Frequency} = 28 + 1325/X$$

where X is the ranking, with a correlation coefficient of 0.978. This is similar to Zipf's law, which states that the *n*th most frequent word is encountered $1/n$ as frequently as the most common.

Table 1

The thirty most-used instructions in PET BASIC and their frequency of use.

FREQUENCY	INSTRUCTION	FREQUENCY	INSTRUCTION
648	JSR adr16	116	STX zpg
511	STA zpg	110	LDX #data
437	LDA zpg	109	LDA (zpg),Y
388	BNE loc	96	LDY zpg
320	LDA #data	92	LDX zpg
267	BEQ loc	75	BPL loc
221	JMP adr16	73	STA adr16
197	LDY #data	72	STA (zpg),Y
153	CMP #data	72	INC zpg
144	ADC #data	67	BCS loc
136	TRS	60	TAX
135	STY zpg	56	CLC
127	BCC loc	55	ADC zpg
127	PHA	55	ADC #data
121	INY	54	BMI loc

Table 2

The forty-one least-used (i.e., never) instructions, in numerical order.

BRK	AND adr16,X	ADC adr16,X
ORA (zpg,X)	AND adr16,X	ROR adr16,X
ASL adr16	ROL adr16,X	STX zpg,Y
ORA (zpg),Y	EOR (zpg,X)	LDA (zpg,X)
ORA zpg,X	EOR adr16	LDX zpg,Y
ORA adr16,Y	LSR adr16	CLV
ORA adr16,X	EOR (zpg),Y	LDY adr16,X
ASL adr16,X	EOR zpg,X	CPY adr16
AND (zpg,X)	EOR adr16,Y	CMP zpg,X
AND adr16	EOR adr16,X	DEC adr16,X
ROL adr16	LSR adr16,X	SBC (zpg,X)
AND (zpg),Y	ADC (zpg,X)	SED
AND zpg,X	ROR adr16	INC adr16,X
ROL zpg,X	ADC zpg,X	

by **Howard W. Whitlock, Jr.**

Howard W. Whitlock, Jr., 1614 Norman Way, Madcity, WI 53705.

Table 3

All used opcodes and frequency; in decreasing order. Those opcodes employing post-indexed addressing are flagged with a *; those using pre-indexing with a †.

OPCODE: 20	COUNT= 684	OPCODE: 4A	COUNT= 19
OPCODE: 85	COUNT= 511	OPCODE: 45	COUNT= 18
OPCODE: A5	COUNT= 437	OPCODE: E4	COUNT= 17
OPCODE: D0	COUNT= 388	OPCODE: D1 *	COUNT= 17
OPCODE: A9	COUNT= 320	OPCODE: C0	COUNT= 17
OPCODE: F0	COUNT= 267	OPCODE: 66	COUNT= 16
OPCODE: 4C	COUNT= 221	OPCODE: 99	COUNT= 15
OPCODE: A0	COUNT= 197	OPCODE: EA	COUNT= 14
OPCODE: C9	COUNT= 154	OPCODE: 46	COUNT= 14
OPCODE: 68	COUNT= 144	OPCODE: 58	COUNT= 13
OPCODE: 60	COUNT= 136	OPCODE: 78	COUNT= 12
OPCODE: 84	COUNT= 135	OPCODE: 2A	COUNT= 12
OPCODE: 90	COUNT= 127	OPCODE: 26	COUNT= 11
OPCODE: 48	COUNT= 127	OPCODE: 05	COUNT= 11
OPCODE: C8	COUNT= 121	OPCODE: 28	COUNT= 10
OPCODE: 86	COUNT= 116	OPCODE: 9A	COUNT= 9
OPCODE: A2	COUNT= 110	OPCODE: 08	COUNT= 9
OPCODE: B1 *	COUNT= 109	OPCODE: B4	COUNT= 8
OPCODE: A4	COUNT= 96	OPCODE: 50	COUNT= 8
OPCODE: A6	COUNT= 92	OPCODE: 25	COUNT= 8
OPCODE: 10	COUNT= 75	OPCODE: 06	COUNT= 8
OPCODE: 8D	COUNT= 73	OPCODE: DD	COUNT= 7
OPCODE: E6	COUNT= 72	OPCODE: BA	COUNT= 7
OPCODE: 91 *	COUNT= 72	OPCODE: 94	COUNT= 7
OPCODE: B0	COUNT= 67	OPCODE: 8E	COUNT= 7
OPCODE: AA	COUNT= 60	OPCODE: 70	COUNT= 6
OPCODE: 18	COUNT= 56	OPCODE: 6C	COUNT= 6
OPCODE: 69	COUNT= 55	OPCODE: 76	COUNT= 5
OPCODE: 65	COUNT= 55	OPCODE: F5	COUNT= 4
OPCODE: 30	COUNT= 54	OPCODE: AE	COUNT= 4
OPCODE: E8	COUNT= 49	OPCODE: 79	COUNT= 4
OPCODE: C6	COUNT= 48	OPCODE: F1 *	COUNT= 3
OPCODE: 8A	COUNT= 48	OPCODE: CD	COUNT= 3
OPCODE: 88	COUNT= 48	OPCODE: 8C	COUNT= 3
OPCODE: 98	COUNT= 46	OPCODE: F6	COUNT= 2
OPCODE: A8	COUNT= 45	OPCODE: EE	COUNT= 2
OPCODE: CA	COUNT= 43	OPCODE: D9	COUNT= 2
OPCODE: C5	COUNT= 43	OPCODE: D8	COUNT= 2
OPCODE: 38	COUNT= 40	OPCODE: AC	COUNT= 2
OPCODE: 29	COUNT= 40	OPCODE: 71 *	COUNT= 2
OPCODE: AD	COUNT= 38	OPCODE: 56	COUNT= 2
OPCODE: 49	COUNT= 37	OPCODE: 40	COUNT= 2
OPCODE: BD	COUNT= 35	OPCODE: 16	COUNT= 2
OPCODE: 09	COUNT= 31	OPCODE: FD	COUNT= 1
OPCODE: E5	COUNT= 30	OPCODE: F9	COUNT= 1

(Continued on next page)

An amazing 27% of the 6502's opcodes are not used at all! These are contained in Table 2. We see that

- 1) The not-used instructions are heavily represented by logical (especially XOR) instructions.
- 2) While there are no pre-indexed, indirect memory references in the top 20%, they are fairly common (six of eight) in the never-used table. The only ones used at all (see Table 2) are STA (zpg,X) and CMP (zpg,X). These are used in conjunction with a zero-page string descriptor stack.
- 3) Neither BRK nor SED (set decimal mode for arithmetic) are used.
- 4) The only post-indexed instructions never used are the logical ones, ORA, AND, and EOR. I have included all used opcodes as Table 3.

I have also counted all adjacent opcode pairs drawn from the instructions of Table 1. The ten most frequently used doublets are in Table 4. Almost all of the doublets are encountered more frequently than expected in a statistical sense (Fa*Fb/6355). There are some notable exceptions, however. The two compare-and-branch pairs, CMP #data: BNE loc, and CMP #data: BEQ loc, are much more common than expected. On the other hand, the doublet BNE loc: JSR loc is (relatively) uncommon. Rather than delving into the Freudian implications of these and related observations, I invite interested readers to send me a SASE and \$1.00 for the complete printout.

Conclusions

My original suspicions concerning the worthlessness of the 6502's pre-indexed instructions seem confirmed. On the other hand, post-indexed indirection, although much more common than pre-indexing, is not nearly as common as I had assumed. The small number of instructions actually used is surprising. The absence of any "super" two-op constructs is apparent from Table 4.

■ ■

Synertek — long dedicated to the 6502 — has designed a new 6809-based SYM SBC. Although the brand-new Osborne-1 system is Z80-based, the only possible rationale for that is the existence of much complex, tested Z80 software; my guess is that the Osborne-2 will bypass the 6809 level and upgrade either to the 68000 or the coming 32-bit Intel processor (again depending on the availability of software). No older processor can compete in performance with the 6809. Its problems are lack of software, reluctance of owners of working systems to upgrade it (partly because of the software cost), and the possibility that more powerful processors may attain a superior price/performance ratio.

It must be the difficulty of equalling the performance of the 6809 that has already aborted the projected 6516 upgrade of the 6502 (and no doubt other upgrades or new designs). This leaves the field wide-open to Japanese designers, who are unwisely scorned by some notable Americans. American companies may be unwilling to gamble, compete with Motorola (at this level), or learn how to handle designers of genius. The Japanese may.

My personal reaction — significant only to the extent that it may prove typical of 6502 *aficionados* — is that there's no need for a rush move to the 6809. It should be 6800 system users who adopt it first, do the spadework, and prove its programming speed, efficiency, and elegance. It will surely excel, but by what margin? Even 6800 users have not stampered to its upgrade, proving that the machine-code-compatibility (with the older design) played a major role in the quick adoption of the Z80 — however grotesque it may seem to a logical mind. The Motorola 6809 assembler can "translate" 6800 assembly-language into the 6809 equivalents, and similar translators could be written for other micro assembly-languages. However, this is not likely to yield optimal code, using the full power of the 6809. Only the human mind can do that. Although I do not admire the 6800 set, regret that so much of it was retained, and (as a first impression) feel that not all the 6809 innovations are ideal, its overall superiority is so great that I intend to get the SYM version.

H. T. Gordon
College of Natural Resources
U. C. Berkeley, Berkeley, CA 94702

More on N-Logs

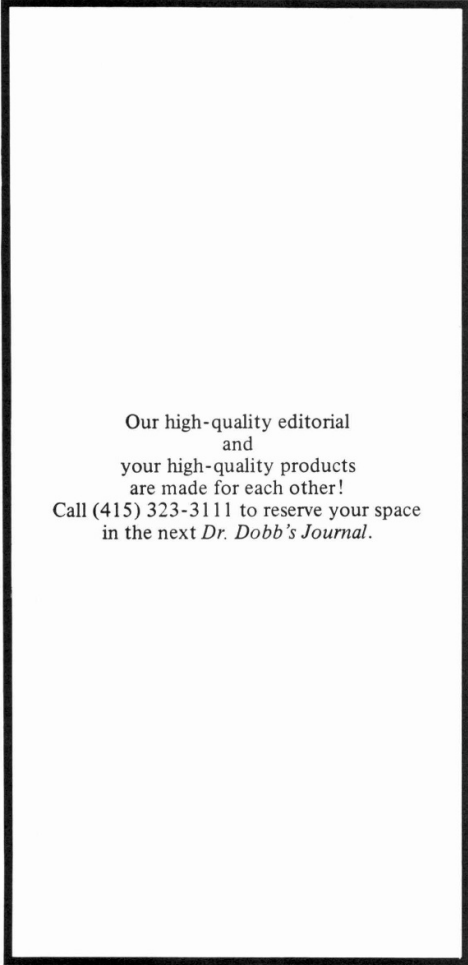
Dear Editor,

The comments of Mr. Mikes in *DDJ* #52 regarding my article on N-Logs were greatly appreciated. His criticisms fall into two groups, the first relating to history, the second having to do with the

meaning of words and symbols. I will try to respond.

His remarks as to the origin of 'e', the base of what are now called 'Natural' or 'Naperian' logarithms are quite correct. I was deceived by the all-too-common

(Continued on page 22)



Our high-quality editorial
and
your high-quality products
are made for each other!
Call (415) 323-3111 to reserve your space
in the next *Dr. Dobb's Journal*.