executed. If there aren't many in the source code to begin with, it doesn't seem to matter much.

4. Short relative instructions are higher in dynamic frequency and long relative are lower.

These conclusions are pretty solid and I feel they could be used to extrapolate static data into dynamic data.

Indexed addressing occurs more frequently in the dynamic case. Immediate and extended occur less frequently. For the indexed subclasses, auto increment and the register offsets occur more frequently and the constant and no offset submodes occur less frequently

The addressing mode conclusions are not very solid and probably could not be used to extrapolate static into dynamic data.

8.7. HOW WELL DID I DO?

When I started this project I had three major objectives. They were:

1. To do an extensive analysis of the M6809 and provide sufficient raw data for future architects.

2. To test the decisions that were made in the M6809 design (i.e., to learn from my past successes and

failures).

3. To determine whether static data can be used to determine dynamic characteristics in an M6809-like architecture.

I do feel I provided a great deal of good, representative M6809 data. I also feel the following areas could be improved upon by me or someone else at a later date:

a. Figure out a reasonable way to take some dynamic data where the number of cycles per instruction is saved. This would allow the average percent of time an opcode takes to be calculated.

b. More dynamic addressing mode data is needed before I will feel comfortable with the addressing mode results.

c. The shifts showed some peculiar characteristics that I cannot completely explain. I suspect that they are being used for multi-bit or multi-precision shifts, but I don't know which. In the future it would be nice to gather some more data in this area.

For question 2 above -- How good were our decisions? The ledger seems to be on the positive side. It seems we did an excellent job of designing the postbyte for the indexed

addressing mode. It only requires .17 bytes more per indexed instruction than the M6800 and only takes about .2 more cycles.

We did a good job of estimating what type of programs would be written for the M6809. They are position independent, modular, and use a great deal of stack addressing just as we had hoped.

We also made good tradeoffs when we decided which instructions should be on page 1, 2 and 3. (A last minute decision in the M6809 design to move lbra and lbsr from page 2 to page 1 saved us here.)

The new instructions we added are very heavily used. Three of them are the most frequently executed single opcodes.

Into each life some bits must fall. The decision to include the direct page register and to expand the direct addressing mode to include read-modify-write instructions was a big mistake. The M6800 data told us this was a useful extension. Our gut told us differently. In this case we went with the data, and the result is that we designed in a feature that was losing popularity due to lower memory prices.

Indirect addressing was another mistake. In this case we followed our gut and ignored the existing data that suggested that indirect addressing was not used. Fortunately,

this mistake didn't cost a lot of silicon (silicon = $).

I don't feel I completely answered the third question (Can static data be used to predict dynamic characteristics?). I feel that static instruction set data can be extrapolated into dynamic characteristics, at least for an architecture similar to the M6809. However, I am not sure it would apply to a greatly differing architecture. More work needs to be done here. Further, I don't feel confident that the static addressing mode data (especially indexed) can be used to determine the dynamic performance. More data needs to be taken here too.

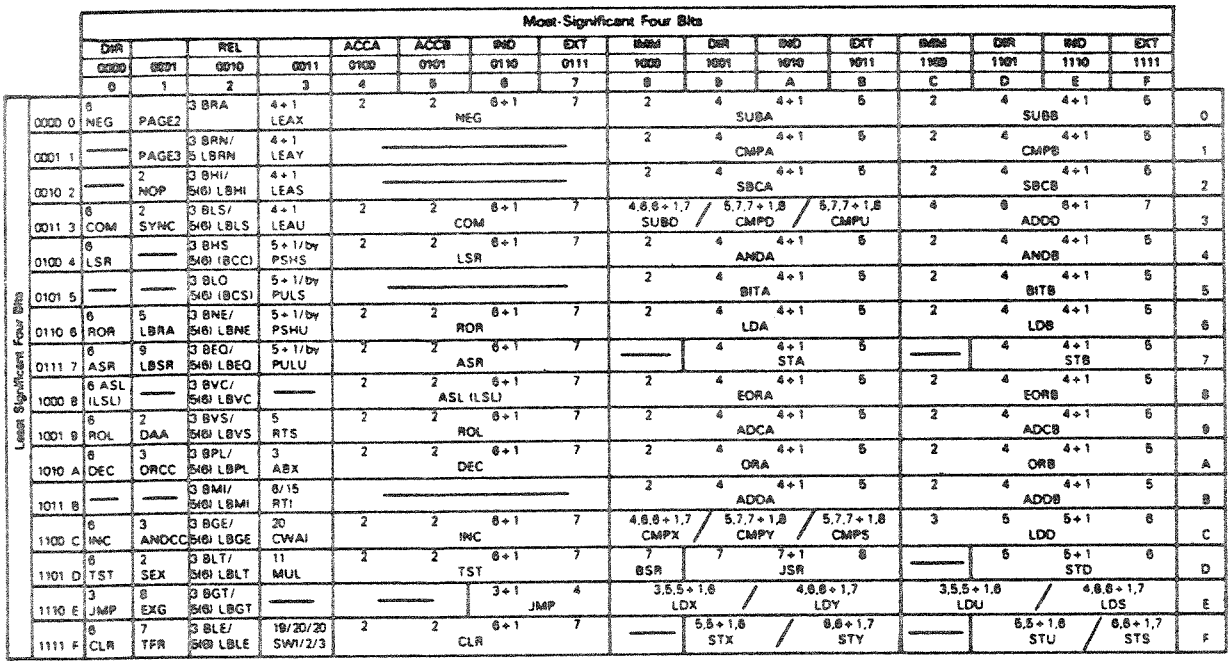
APPENDIX I -- THE M6809 INSTRUCTION SET

This appendix contains a brief description of the instruction set of the M6809.

ABX	Add the B register to the X register	
ADCx	Add carry bit and memory to accumulator A or B	
ADDx	Add memory to accumulator A, B or D	
ANDx	And memory with accumulator A or B	
ANDCC	And immediate value with condition code register	
ASL	Arithmetic shift left memory or accumulator A or B	
ASR	Arithmetic shift right memory or accumulator A or B	
(L)BCC	Branch if carry is clear	
(L)BCS	Branch if carry is set	
(L)BEQ	Branch if equal	
(L)BGE	Branch if greater than or equal	(signed)
(L)BGT	Branch if greater than	(signed)
(L)BHI	Branch if higher	(unsigned)
(L)BHS	Branch if higher or same	(unsigned)
(L)BLE	Branch if less than or equal	(signed)
(L)BLO	Branch if lower	(unsigned)
(L)BLS	Branch if lower or same	(unsigned)
(L)BLT	Branch if less than	(signed)
(L)BMI	Branch if minus	
(L)BNE	Branch if not equal	
(L)BPL	Branch if plus	
(L)BRA	Branch always	
(L)BSR	Branch to subroutine	
(L)BVC	Branch if v clear	
(L)BVS	Branch if v set	
BITx	And memory with A or B (don't store result)	
CLR	Clear memory or accumulator A or B	
CMPx	Compare memory with A,B,D,X,Y,U, or S	
COM	Complement memory or accumulator A or B	
CWAI	Clear interrupt mask; wait for interrupt	
DAA	Decimal adjust accumulator A (follows an ADD)	
DEC	Decrement memory or accumulator A or B	
EOR	Exclusive or memory with accumulator A or B	
EXG	Exchange R1 with R2	
INC	Increment memory or accumulator A or B	
JMP	Jump	
JSR	Jump to subroutine	
LDx	Load A,B,D,X,Y,U, or S from memory	
LEAx	Load effective address into X,Y,U or S	
LSL	Logical left shift memory or accumulator A or B	
LSR	Logical right shift memory or accumulator A or B	
MUL	A * B --> D	
NEG	Negate memory or accumulator A or B	

NOP	No-operation
ORx	Or memory with accumulator A or B
ORCC	Or immediate value with condition code register
PSHx	Push register(s) on U or S stack
PULx	Pull register(s) from U or S stack
ROL	Rotate left thru carry memory or accumulator A or B
ROR	Rotate right thru carry memory or accumulator A or B
RTI	Return from interrupt
RTS	Return from subroutine
SBCx	Subtract carry bit and memory from accumulator A or B
SEX	Sign extend B accumulator thru A
STx	Store A,B,D,X,Y,U or S to memory
SUBx	Subtract memory from accumulator A, B or D
SWIn	Software interrupt 1, 2 or 3
SYNC	Synchronize with external event
TFR	Transfer R1 to R2
TST	Test memory or accumulator A or B for zero

APPENDIX II -- OPCODE MAP AND INDEXING FORMATS



Least Significant Four Bits	Most Significant Four Bits: 0 (0000) DIR	1 (0001)	2 (0010) REL	3 (0011)	4 (0100) ACCA	5 (0101) ACCB	6 (0110) IND	7 (0111) EXT	8 (1000) IMM	9 (1001) DIR	A (1010) IND	B (1011) EXT	C (1100) IMM	D (1101) DIR	E (1110) IND	F (1111) EXT	Least Significant Four Bits
0000 0	6 NEG	PAGE2	3 BRA	4+1 LEAX	2	2	6+1 NEG	7	2	4	4+1 SUBA	5	2	4	4+1 SUBB	5	0
0001 1	—	PAGE3	3 BRN/ 5 LBRN	4+1 LEAY	—				2	4	4+1 CMPA	5	2	4	4+1 CMPB	5	1
0010 2	—	2 NOP	3 BHI/ 5(6) LBHI	4+1 LEAS	—				2	4	4+1 SBCA	5	2	4	4+1 SBCB	5	2
0011 3	6 COM	2 SYNC	3 BLS/ 5(6) LBLS	4+1 LEAU	2	2	6+1 COM	7	4,6,6+1,7 SUBD	5,7,7+1,8 CMPD /	5,7,7+1,8 CMPU		4	6	6+1 ADDD	7	3
0100 4	6 LSR	—	3 BHS 5(6) (BCC)	5+1/by PSHS	2	2	6+1 LSR	7	2	4	4+1 ANDA	5	2	4	4+1 ANDB	5	4
0101 5	—	—	3 BLO 5(6) (BCS)	5+1/by PULS	—				2	4	4+1 BITA	5	2	4	4+1 BITB	5	5
0110 6	6 ROR	5 LBRA	3 BNE/ 5(6) LBNE	5+1/by PSHU	2	2	6+1 ROR	7	2	4	4+1 LDA	5	2	4	4+1 LDB	5	6
0111 7	6 ASR	9 LBSR	3 BEQ/ 5(6) LBEQ	5+1/by PULU	2	2	6+1 ASR	7	—	4	4+1 STA	5	—	4	4+1 STB	5	7
1000 8	6 ASL (LSL)	—	3 BVC/ 5(6) LBVC	—	2	2	6+1 ASL (LSL)	7	2	4	4+1 EORA	5	2	4	4+1 EORB	5	8
1001 9	6 ROL	2 DAA	3 BVS/ 5(6) LBVS	5 RTS	2	2	6+1 ROL	7	2	4	4+1 ADCA	5	2	4	4+1 ADCB	5	9
1010 A	6 DEC	3 ORCC	3 BPL/ 5(6) LBPL	3 ABX	2	2	6+1 DEC	7	2	4	4+1 ORA	5	2	4	4+1 ORB	5	A
1011 B	—	—	3 BMI/ 5(6) LBMI	6/15 RTI	—				2	4	4+1 ADDA	5	2	4	4+1 ADDB	5	B
1100 C	6 INC	3 ANDCC	3 BGE/ 5(6) LBGE	20 CWAI	2	2	6+1 INC	7	4,6,6+1,7 CMPX	5,7,7+1,8 CMPY /	5,7,7+1,8 CMPS		3	5	5+1 LDD	6	C
1101 D	6 TST	2 SEX	3 BLT/ 5(6) LBLT	11 MUL	2	2	6+1 TST	7	7 BSR	7	7+1 JSR	8	—	5	5+1 STD	6	D
1110 E	3 JMP	8 EXG	3 BGT/ 5(6) LBGT	—	—		3+1 JMP	4	3,5,5+1,6 LDX	/	4,6,6+1,7 LDY		3,5,5+1,6 LDU	/	4,6,6+1,7 LDS		E
1111 F	6 CLR	7 TFR	3 BLE/ 5(6) LBLE	19/20/20 SWI/2/3	2	2	6+1 CLR	7	—	5,5+1,6 STX	/	6,6+1,7 STY	—	5,5+1,6 STU	/	6,6+1,7 STS	F

Figure 10-1: The M6809 Opcode Map.

The small numbers in the map are the execution cycles for each opcode. For indexed addressing the number of cycles from the indexed format table must be added to the base cycle count.

Type	Forms	Non Indirect: Assembler Form	Non Indirect: Postbyte OP Code	Non Indirect: + —	Non Indirect: + #	Indirect: Assembler Form	Indirect: Postbyte OP Code	Indirect: + —	Indirect: + #
Constant Offset From R (twos complement offset)	No Offset	,R	1RR00100	0	0	[,R]	1RR10100	3	0
	5 Bit Offset	n, R	0RRnnnnn	1	0	defaults to 8-bit			
	8 Bit Offset	n, R	1RR01000	1	1	[n, R]	1RR11000	4	1
	16 Bit Offset	n, R	1RR01001	4	2	[n, R]	1RR11001	7	2
Accumulator Offset From R (twos complement offset)	A — Register Offset	A, R	1RR00110	1	0	[A, R]	1RR10110	4	0
	B — Register Offset	B, R	1RR00101	1	0	[B, R]	1RR10101	4	0
	D — Register Offset	D, R	1RR01011	4	0	[D, R]	1RR11011	7	0
Auto Increment/Decrement R	Increment By 1	,R+	1RR00000	2	0	not allowed			
	Increment By 2	,R++	1RR00001	3	0	[,R++]	1RR10001	6	0
	Decrement By 1	,-R	1RR00010	2	0	not allowed			
	Decrement By 2	,--R	1RR00011	3	0	[,--R]	1RR10011	6	0
Constant Offset From PC (twos complement offset)	8 Bit Offset	n, PCR	1XX01100	1	1	[n, PCR]	1XX11100	4	1
	16 Bit Offset	n, PCR	1XX01101	5	2	[n, PCR]	1XX11101	8	2
Extended Indirect	16 Bit Address	—	—	—	—	[n]	10011111	5	2

R = X, Y, U or S X = 00 Y = 01
X = Don't Care U = 10 S = 11

+ and + indicate the number of additional cycles and bytes for the particular variation

Figure 10-2: Indexed Addressing Mode Formats.

APPENDIX III -- STATIC DATA

The following represents the concatenation of all the static data. The data for the individual program classes and the individual programs is available from the author.

M6809 Static Statistics

Sorted by Opcode Frequency

Total number of instructions = 26330

Total number of bytes = 61977

Average size of instruction = 2.35 bytes

Total number of page 1 instructions = 24960

Percent of total = 94.80

Total number of page 1 bytes = 56863

Percent of total = 91.75

Opcode(hex)	Count	Percent	Bytes	Percent
17	2307	8.76	6921	11.17
30	922	3.50	2653	4.28
34	910	3.46	1820	2.94
86	906	3.44	1812	2.92
20	877	3.33	1754	2.83
8e	862	3.27	2586	4.17
26	804	3.05	1608	2.59
27	800	3.04	1600	2.58
ed	739	2.81	1584	2.56
cc	722	2.74	2166	3.49
ec	663	2.52	1451	2.34
35	645	2.45	1290	2.08
ae	613	2.33	1272	2.05

bd	605	2.30	1815	2.93
a6	588	2.23	1261	2.03
c6	564	2.14	1128	1.82
81	517	1.96	1034	1.67
36	516	1.96	1032	1.67
1f	486	1.85	972	1.57
39	434	1.65	434	0.70
e6	425	1.61	907	1.46
a7	417	1.58	889	1.43
32	405	1.54	955	1.54
16	403	1.53	1209	1.95
e7	366	1.39	886	1.43
8d	347	1.32	694	1.12
b7	293	1.11	879	1.42
de	255	0.97	510	0.82
af	251	0.95	534	0.86
31	241	0.92	616	0.99
b6	202	0.77	606	0.98
c1	183	0.70	366	0.59
3f	167	0.63	198	0.32
5f	150	0.57	150	0.24
84	141	0.54	282	0.46
33	140	0.53	405	0.65
6f	131	0.50	273	0.44
2b	127	0.48	254	0.41
4f	123	0.47	123	0.20
5a	117	0.44	117	0.19
be	111	0.42	333	0.54
25	110	0.42	220	0.35
24	109	0.41	218	0.35
6c	105	0.40	271	0.44
96	101	0.38	202	0.33
83	99	0.38	297	0.48
69	89	0.34	178	0.29
6d	89	0.34	183	0.30
9e	88	0.33	176	0.28
8a	87	0.33	174	0.28
a1	84	0.32	191	0.31
85	83	0.32	166	0.27
1d	82	0.31	82	0.13
c3	82	0.31	246	0.40
9f	81	0.31	162	0.26
bf	81	0.31	243	0.39
aa	78	0.30	158	0.25
2d	76	0.29	152	0.25
7f	73	0.28	219	0.35
2a	72	0.27	144	0.23
37	71	0.27	142	0.23
2c	69	0.26	138	0.22
fc	68	0.26	204	0.33

8c	65	0.25	195	0.31
f6	64	0.24	192	0.31
fd	63	0.24	189	0.30
e3	62	0.24	126	0.20
66	61	0.23	122	0.20
7e	54	0.21	162	0.26
97	53	0.20	106	0.17
5c	52	0.20	52	0.08
ac	52	0.20	110	0.18
6a	51	0.19	136	0.22
c4	51	0.19	102	0.16
f7	51	0.19	153	0.25
c	48	0.18	96	0.15
4c	48	0.18	48	0.08
58	47	0.18	47	0.08
ab	46	0.17	104	0.17
49	45	0.17	45	0.07
6e	45	0.17	121	0.20
a3	45	0.17	93	0.15
f	44	0.17	88	0.14
7d	44	0.17	132	0.21
2f	42	0.16	84	0.14
c5	42	0.16	84	0.14
ea	42	0.16	84	0.14
1c	41	0.16	82	0.13
1e	40	0.15	80	0.13
44	40	0.15	40	0.06
2e	39	0.15	78	0.13
d7	39	0.15	78	0.13
a	38	0.14	76	0.12
4a	38	0.14	38	0.06
ce	37	0.14	111	0.18
1a	35	0.13	70	0.11
22	35	0.13	70	0.11
b1	34	0.13	102	0.16
cb	34	0.13	68	0.11
ee	34	0.13	70	0.11
43	33	0.13	33	0.05
e1	33	0.13	74	0.12
ef	32	0.12	64	0.10
d	31	0.12	62	0.10
a4	31	0.12	64	0.10
ad	30	0.11	90	0.15
e4	30	0.11	60	0.10
dd	29	0.11	58	0.09
40	28	0.11	28	0.05
23	24	0.09	48	0.08
3d	24	0.09	24	0.04
c0	24	0.09	48	0.08
48	23	0.09	23	0.04

bb	23	0.09	69	0.11
eb	23	0.09	52	0.08
3b	21	0.08	21	0.03
4d	21	0.08	21	0.03
50	19	0.07	19	0.03
80	19	0.07	38	0.06
53	18	0.07	18	0.03
d6	18	0.07	36	0.06
7c	17	0.06	51	0.08
ca	17	0.06	34	0.05
46	16	0.06	16	0.03
9c	16	0.06	32	0.05
5d	15	0.06	15	0.02
63	15	0.06	30	0.05
54	14	0.05	14	0.02
8b	14	0.05	28	0.05
56	12	0.05	12	0.02
e0	12	0.05	24	0.04
59	11	0.04	11	0.02
12	10	0.04	10	0.02
3a	10	0.04	10	0.02
88	10	0.04	20	0.03
47	9	0.03	9	0.01
98	9	0.03	18	0.03
ba	9	0.03	27	0.04
c8	9	0.03	18	0.03
dc	9	0.03	18	0.03
82	8	0.03	16	0.03
a8	8	0.03	23	0.04
fe	8	0.03	24	0.04
60	7	0.03	14	0.02
db	7	0.03	14	0.02
f1	7	0.03	21	0.03
7a	6	0.02	18	0.03
9b	6	0.02	12	0.02
bc	6	0.02	18	0.03
d3	6	0.02	12	0.02
e	5	0.02	10	0.02
89	5	0.02	10	0.02
a0	5	0.02	10	0.02
ff	5	0.02	15	0.02
b3	4	0.02	12	0.02
3	3	0.01	6	0.01
64	3	0.01	6	0.01
68	3	0.01	6	0.01
70	3	0.01	9	0.01
73	3	0.01	9	0.01
79	3	0.01	9	0.01
a9	3	0.01	6	0.01
df	3	0.01	6	0.01

f0	3	0.01	9	0.01
f3	3	0.01	9	0.01
19	2	0.01	2	0.00
91	2	0.01	4	0.01
93	2	0.01	4	0.01
d8	2	0.01	4	0.01
fb	2	0.01	6	0.01
28	1	0.00	2	0.00
57	1	0.00	1	0.00
76	1	0.00	3	0.00
9a	1	0.00	2	0.00
b4	1	0.00	3	0.00
d0	1	0.00	2	0.00

Total number of page 2 instructions = 1354

Percent of total = 5.14

Total number of page 2 bytes = 5058

Percent of total = 8.16

Opcode(hex)	Count	Percent	Bytes	Percent
8e	341	1.30	1364	2.20
ae	182	0.69	571	0.92
26	126	0.48	504	0.81
27	111	0.42	444	0.72
83	106	0.40	424	0.68
a3	75	0.28	248	0.40
af	66	0.25	210	0.34
be	63	0.24	252	0.41
8c	58	0.22	232	0.37
9e	44	0.17	132	0.21
2d	29	0.11	116	0.19
25	24	0.09	96	0.15
ac	16	0.06	48	0.08
2c	14	0.05	56	0.09
9f	12	0.05	36	0.06
2e	10	0.04	40	0.06
bf	10	0.04	40	0.06
24	9	0.03	36	0.06
2b	7	0.03	28	0.05
2f	7	0.03	28	0.05
3f	7	0.03	14	0.02
b3	7	0.03	28	0.05
ee	5	0.02	16	0.03
ce	4	0.02	16	0.03

fe	4	0.02	16	0.03
22	3	0.01	12	0.02
bc	3	0.01	12	0.02
23	2	0.01	8	0.01
29	2	0.01	8	0.01
9c	2	0.01	6	0.01
ff	2	0.01	8	0.01
de	1	0.00	3	0.00
df	1	0.00	3	0.00
ef	1	0.00	3	0.00

Total number of page 3 instructions = 16

Percent of total = 0.06

Total number of page 3 bytes = 56

Percent of total = 0.09

Opcode(hex)	Count	Percent	Bytes	Percent
83	7	0.03	28	0.05
a3	7	0.03	21	0.03
93	1	0.00	3	0.00
b3	1	0.00	4	0.01

*** Instruction Classes ***

op	count	%	bytes	%
ld16	4114	15.62	11291	18.22
ld	2868	10.89	6144	9.91
lbsr	2307	8.76	6921	11.17
lea	1708	6.49	4629	7.47
psh	1426	5.42	2852	4.60
st16	1376	5.23	3155	5.09
st	1219	4.63	2991	4.83
bra	877	3.33	1754	2.83
cmp	860	3.27	1792	2.89
bne	804	3.05	1608	2.59
beq	800	3.04	1600	2.58
pul	716	2.72	1432	2.31
jsr	635	2.41	1905	3.07
clr	521	1.98	853	1.38
tfr	486	1.85	972	1.57
rts	434	1.65	434	0.70

cmpl6	422	1.60	1409	2.27
lbra	403	1.53	1209	1.95
bsr	347	1.32	694	1.12
inc	270	1.03	518	0.84
and	254	0.96	511	0.82
dec	250	0.95	385	0.62
or	234	0.89	479	0.77
tst	200	0.76	413	0.67
swi	174	0.66	212	0.34
add	155	0.59	353	0.57
add16	153	0.58	393	0.63
sub16	150	0.57	406	0.66
rol	148	0.56	243	0.39
bmi	127	0.48	254	0.41
lbne	126	0.48	504	0.81
bit	125	0.47	250	0.40
lbeq	111	0.42	444	0.72
bcs	110	0.42	220	0.35
bcc	109	0.41	218	0.35
jmp	104	0.39	293	0.47
ror	90	0.34	153	0.25
sex	82	0.31	82	0.13
blt	76	0.29	152	0.25
asl	73	0.28	76	0.12
bpl	72	0.27	144	0.23
com	72	0.27	96	0.15
bge	69	0.26	138	0.22
sub	64	0.24	131	0.21
lsr	57	0.22	60	0.10
neg	57	0.22	70	0.11
ble	42	0.16	84	0.14
andcc	41	0.16	82	0.13
exg	40	0.15	80	0.13
bgt	39	0.15	78	0.13
eor	38	0.14	83	0.13
bhi	35	0.13	70	0.11
orcc	35	0.13	70	0.11
lblt	29	0.11	116	0.19
bls	24	0.09	48	0.08
lbcs	24	0.09	96	0.15
mul	24	0.09	24	0.04
rti	21	0.08	21	0.03
lbge	14	0.05	56	0.09
abx	10	0.04	10	0.02
asr	10	0.04	10	0.02
lbgt	10	0.04	40	0.06
nop	10	0.04	10	0.02
lbcc	9	0.03	36	0.06
adc	8	0.03	16	0.03
sbc	8	0.03	16	0.03

lble	7	0.03	28	0.05
lbmi	7	0.03	28	0.05
lbhi	3	0.01	12	0.02
daa	2	0.01	2	0.00
lbls	2	0.01	8	0.01
lbvs	2	0.01	8	0.01
bvc	1	0.00	2	0.00
brn	0	0.00	0	0.00
bvs	0	0.00	0	0.00
cwai	0	0.00	0	0.00
illop	0	0.00	0	0.00
lbpl	0	0.00	0	0.00
lbrn	0	0.00	0	0.00
lbvc	0	0.00	0	0.00
sync	0	0.00	0	0.00

*** Larger Instruction Classes ***

Class	count	%	bytes	%
load	6982	26.52	17435	28.13
call	3289	12.49	9520	15.36
cond_br	2652	10.07	5992	9.67
store	2595	9.86	6146	9.92
psh_pul	2142	8.14	4284	6.91
addr	1708	6.49	4629	7.47
cmp_tst	1482	5.63	3614	5.83
xfr	1384	5.26	3256	5.25
arith	538	2.04	1315	2.12
logical	526	2.00	1073	1.73
inc_dec	520	1.97	903	1.46
shifts	378	1.44	542	0.87
tot	22234	84.44	58709	94.73

*** Addressing Mode Usage ***

Addressing mode	count	percent
indexed	7371	27.99
immediate	5132	19.49
short_relative	3532	13.41
inherent	3466	13.16
long_relative	3054	11.60
extended	1937	7.36
direct	958	3.64

accumulator_b	456	1.73
accumulator_a	424	1.61
indirect	175	0.66

Indexed Addressing Statistics:

subgroup	addr mode	number	% of total	% of subgrp.
inc/dec	+	286	3.88	39.61
inc/dec	++	120	1.63	16.62
inc/dec	-	43	0.58	5.96
inc/dec	--	273	3.70	37.81
offset	5	3940	53.45	73.32
offset	8	631	8.56	11.74
offset	16	572	7.76	10.64
offset	pc8	92	1.25	1.71
offset	pc16	139	1.89	2.59
reg_offset	a	85	1.15	28.62
reg_offset	b	99	1.34	33.33
reg_offset	d	113	1.53	38.05
no_offset	0	961	13.04	100.00
ext_indirect		17	0.23	100.00

Average additional bytes for indexed = 1.17

Index register usage:

reg	number	%
u	2411	32.71
s	1801	24.43
x	1762	23.90
y	1149	15.59
pc	231	3.13

Total number of relative instructions = 6586

Total short = 3532 53.63 percent

Total long = 3Ø54 46.37 percent

*** Push/Pull Statistics ***

Total number of push/pulls = 2142

Average number of registers push/pulled per instr. = 2.25

Register push/pulled	count	percent
a	1216	25.23
x	1Ø91	22.63
b	844	17.51
y	837	17.37
pc	442	9.17
u	299	6.2Ø
cc	62	1.29
dpr	27	Ø.56
s	2	Ø.Ø4

*** Direct Page Register Statistics ***

Number of loads of the dpr = 9

BIBLIOGRAPHY

[BON1] Boney, Joel and Ritter, Terry, "A Microprocessor For the Revolution, Part 1: Design Philosophy", Byte, Vol. 4, No. 1, January 1979, pp. 14-42.

[BON2] Boney, Joel and Ritter, Terry, "A Microprocessor For the Revolution, Part 2: Instruction Set Dead Ends, Old Trails, and Apologies", Byte, Vol. 4, No. 2, February 1979, pp. 32-42.

[BON3] Boney, Joel and Ritter, Terry, "A Microprocessor For the Revolution, Part 3: Final Thoughts", Byte, Vol. 4, No. 3, March 1979, pp. 46-52.

[BON4] Boney, Joel, "M6800 Instruction Analysis", Internal Motorola memo, March 2, 1977. (Available from author)

[KER] Kernighan, Brian W. and Plauger, P.J., Software Tools, Yourdon Inc., 1976, pp. 163-219.

[LUN] Lunde, Amund, "Empirical Evaluation of Some Features of Instruction Set Processor Architectures", Communications of the ACM, Vol. 20, No. 3, March 1977, pp. 143-152.

[MAR] Marathe, Madhav, Excerpt from his thesis on the PDP-11 instruction mix (personal correspondence to Dr. Skip Stritter), August 8, 1977.

[MOT1] MC6801 8-Bit Single-Chip Microcomputer Reference Manual, MC6801RM(AD), Motorola Literature Distribution Center, 1980.

[MOT2] MC6809 - MC6809E Microprocessor Programming Manual, M6809PM(AD), Motorola Literature Distribution Center, March 1, 1981.

[PEU] Peuto, Bernard L., and Shustek, Leonard J., "An Instruction Timing Model of CPU Performance", 4th Symposium on Computer Architecture, March 1977, pp. 165-177.

[SHU] Shustek, Leonard Jay, "Analysis and Performance of Computer Instruction Sets", Stanford Linear Accelerator Center Report No. 205, STAN-CS-78-658, January 1978.

[STO] Stone, Harold S., editor, Introduction to Computer Architecture, Science Research Associates, 1975, pp. 525-528.