Stevens Institute of Technology

Research in the 68000 Assembly Language

Course: CS550 Computer Organization and Programming

Junjie Chen

Professor: Edward Banduk

October 2021

1. The 68000 Processor

A microprocessor is a computer processor in which data processing logic and control are contained in a single integrated circuit or in a small number of integrated circuits. Physically, it is a semiconductor integrated circuit chip that integrates a large number of micro transistors and other electronic components.

The Motorola's 68000 Processor is a 16/32-bit complex instruction set computer (CISC) microprocessor and the most significant of the 32-bit designs of microprocessors, having been released in 1979. It was the most popular CPU design of its class due to its combination of great performance, large memory space (16 megabytes), and relatively inexpensive cost.

Memory is logically addressed in 8-bit bytes, 16-bit words, or 32-bit long words. It requires the alignment of word and long-word data. In bit manipulation instructions, bits are independently addressable.

There are several addressing modes are available:

a) Inherent

   Register or memory direct or indirect addressing.

b) Immediate

   It can be divided into immediate and quick immediate, which are the next instruction word and the subfield of the instruction, respectively.

c) Relative

   The operand address in the instruction is the contents of the specified address register plus a 16-bit signed displacement.

d) Extended

Register direct or indirect addressing with offset.

e) Indexed

The operand address is made up of the contents of the given address register plus the contents of an additional data or address register indicated in the instruction with an 8-bit signed displacement.

2. The 68000 Assembly Language

As we all know, computers use binary to work internally. Assembly language is any low-level language used in electronic computers, microprocessors, microcontrollers, or other programmable devices. Assembly languages relate to different sets of machine language instructions in different devices, and each is dedicated to a specific computer system architecture. The 68000 Assembly Language is the specific assembly language used in Motorola 68K-series microprocessors.

Most assembly language uses a mnemonic to represent each low-level machine instruction or opcode, typically also each architectural register, flag, etc. Figure 1 lists instruction mnemonics of the 68000 Assembly Language.

Mnemonic	Description	Mnemonic	Description
ABCD	Add Decimal with Extend	MOVE	Move
ADD	Add	MOVEM	Move Multiple Registers
AND	Logical AND	MOVEP	Move Peripheral Data
ASL	Arithmetic Shift Left	MULS	Signed Multiply
ASR	Arithmetic Shift Right	MULU	Signed Multiply
B_{CC}	Branch Conditionally	NBCD	Negate Decimal with Extend
BCHG	Bit Test and Change	NEG	Negate
BCLR	Bit Test and Clear	NOP	No Operation
BRA	Branch Always	NOT	One's Complement
BSET	Bit Test and Set	OR	Logical OR
BSR	Branch to Subroutine	PEA	Push Effective Address
BTST	Bit Test	RESET	Reset External Devices
CHK	Check Register Against Bounds	ROL	Rotate Left without Extend
CLR	Clear Operand	ROR	Rotate Right without Extend
CMP	Compare	ROXL	Rotate Left with Extend
DB_{CC}	Tst Cond, Decrement and Branch	ROXR	Rotate Right with Extend
		RTE	Return from Exception
DIVS	Signed Divide	RTR	Return and Restore
DIVU	Unsigned Divide	RTS	Return from Subroutine
EOR	Exclusive OR	SBCD	Subtract Decimal with Extend
EXG	Exchange Registers	S_{CC}	Set Conditional
EXT	Sign Extend	STOP	Stop
JMP	Jump	SUB	Subtract
JSR	Jump to Subroutine	SWAP	Swap Data Register Halves
LEA	Load Effective Address	TAS	Test and Set Operand
LINK	Link Stack	TRAP	Trap
LSL	Logical Shift Left	TRAPV	Trap on Overflow
LSR	Logical Shift Right	TST	Test
		UNLK	Unlink

Figure 1. Instruction Mnemonics of the 68000 Assembly Language

We can see that most instructions include dot-letter suffixes, allowing for operations on 8-bit bytes (".b"), 16-bit words (".w"), and 32-bit longs (".l") (".l"). And the majority of instructions are dyadic, meaning they have a source and a destination, and the destination can be altered.

3. Assemblers

The primary function of assembler is to convert mnemonic operation codes into binary equivalents. The assembler completes this work with the use of a fixed table, much like the table in figure 1.

However, the assembler must do more than simply translate the operation codes. It must also figure out how many and what kind of operands the instruction requires. This can be complicated since certain instructions (such as Stop) have no operands, while others (such as a Jump instruction) have one and others (such as a register transfer or a multiple-bit shift) require two. Some instructions may also provide options; for example, some computers contain instructions (such as Shift or Clear) that can be applied to a CPU register or a memory address.

Some instructions in assembly language are not directly converted into machine language instructions. These are assembler commands that assign the program to specific memory locations, create symbols, designate memory spaces for data storage, place tables or other fixed data in memory, allow references to other programs, and perform minor housekeeping (like memory cleaning) tasks.

4. The 68000 Simulator

68000 Simulator creates a virtual 68000 microprocessor that runs on your PC. It allows you to run 68000 programs without any 68000 hardware. Among the significant ones are: EASy68K, Crossware 68000 Simulator for Windows.

References

[1] Stritter, E., & Gunter, T. (1979). Microsystems a Microprocessor Architecture for a Changing World: The Motorola 68000. Computer, 12(02), 43-52.

[2] Microprocessor - Wikipedia. (2021). Retrieved 3 October 2021, from https://en.wikipedia.org/wiki/Microprocessor

[3] 68000 Addressing Modes | Direct Register | Direct Memory Addressing. (2021). Retrieved 3 October 2021, from https://www.eeeguide.com/68000-addressing-modes/

[4] Assembly language - Wikipedia. (2021). Retrieved 3 October 2021, from https://en.wikipedia.org/wiki/Assembly_language

[5] Kane, G., Hawkins, D., & Leventhal, L. (1981). 68000 assembly language programming. Berkeley: Osborne/McGraw-Hill.

[6] Motorola 68000 - Wikipedia. (2021). Retrieved 3 October 2021, from https://en.wikipedia.org/wiki/Motorola_68000