Motorola 68008 Simulator User Manual

The 68k/Sim Team

May 8, 2002

Contents

Chapter 1

Introduction

This manual is a guide on how to use the M68k Simulator program, and what tools are necessary for its use.
Use of the program assumes some prior knowledge of Assembly Language, and allows the user to step through compiled programs only. The user can upload a h68 file by pressing the open button, and selecting the required file. The program is meant as a teaching tool, and not only as a simulator. The display and help features of the program will be discussed below.

Chapter 2

Installation

2.1 System Requirements

The program is best viewed using Linux, but can be run on almost any operating system which has Qt libraries. It is available on CD and was written in C++. The graphics were done using Qt, so that it would be fully compatible with all operating systems with Qt libraries available.

2.2 Running the Program

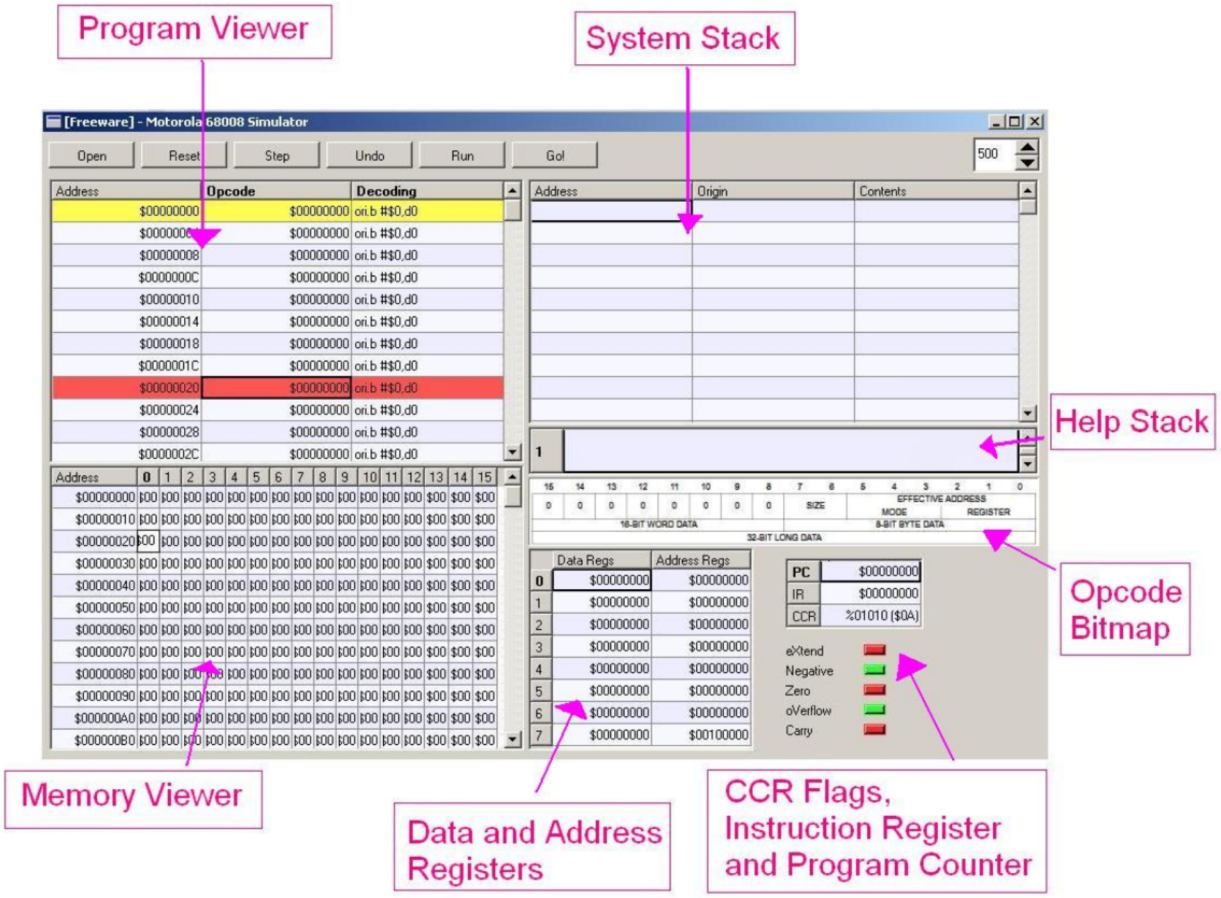
Once the program has been installed, it can be run by starting '68kasm' on *NIX systems or by clicking on the "Motorola 68008 Simulator" entry in the Start Menu in Windows.

2.3 Installing the Program

To install the program, simply insert the CD into your CD-ROM drive. A page with explanation should come up automatically (Windows only). If it doesn't, open index.html on the CD-ROM manually instead.

Chapter 3

Display Features



3.1 Memory Viewer

This shows the contents of memory at all times. The user may change the values in memory directly by double clicking on a memory location, and typing the desired value. It is possible to scroll through the memory to see it's contents, or select a particular address by double clicking the address pointer and entering the required address.

3.2 Program Viewer

This displays the actual written program itself. The first line will display the instruction pointed to by the memory location in the program counter(see 'Section 3.6'). The by-product of this feature is that it emphasises the fact that memory doesn't distinguish between data and code. So if the program counter is not pointing to the desired origin of the program, it will indiscriminately display the associated instruction of the data at the memory location in the pc. The current executing instruction is highlighted by a yellow line and a breakpoint by a red line. A breakpoint is inserted by pressing control and left-clicking on the desired line.

3.3 Help Stack

This feature indicates possible mistakes in the program by anticipating common errors when using assembly language. It displays a suggestion on the possible cause of the mistake and a solution to the problem. It should be noted that the helpstack only guesses the cause of the mistake and sometimes guesses incorrectly. It should also be noted that the helpstack will not halt the flow of the program.

3.4 System Stack

This displays the address of the data, the data itself and the origin of the data on the stack. The stack will always be empty when the program begins and the reset button clears the stack.

3.5 Address and Data Registers

These are fairly self-explanatory. They display the contents of the data registers and the address registers and can be changed,manually, by the user by double-clicking.

3.6 Program Counter and Status Bits

In the extreme, bottom-right-hand corner, the program counter(pc), instruction register(ir) and condition code register(ccr) are displayed. The ccr is displayed both numerically(in decimal and hexadecimal) and visually. You can click on the visual ccr flags and hence manipulate them. You can also double-click on the pc text box and input an address value. This is the means by which the user specfies the starting address(origin) of the program.

3.7 Opcode Window

This shows a pictorial template of how the instruction is encoded according to the m68k manual. It shows the template of the current instruction pointed to by the program counter, and updates itself automatically inline with the pc.

Chapter 4

Buttons

4.1 Open

This button opens the h68 file. This file is generated when an assembly program is successfully compiled. When a h68 file is stored in any directory, it is accessible via the open button.

4.2 Reset

The user uses this button to 'reset' the pc to the start of the program. This clears all the registers and contents of the stack and memory.

4.3 Step

This button is used for debugging the program. It executes one instruction at a time, allowing the user to see the result of each operation after it's executed.

4.4 Undo

A special feature of this program is it's undo capabilities. Whilst stepping through the program, the user is able to step back or 'undo' any number of instructions. This also undoes register, memory and stack operations. As a teaching tool, this is useful as it allows the user to see the effect of the flags. For example, the user can observe the behavior of branching on a status flag by setting the particular flag, executing the instruction and then stepping back, changing the flag and once again, executing the instruction.

4.5 Run

This button allows the user to run through the program at various speeds controlled by a spin box on the far right. It is an alternative to the step button. The user can watch the effects of the program without continuously pressing the step button. For very fast speeds, the user should use the 'go' button.(explained in Section 4.6)As a teaching tool, it is best to run the program slowly so one can monitor what's going on.

4.6 Go

The go button should be used when it is not necessary for the user to see the effects of each instruction. It warns the user that the screen will not be updated while the program is running(due to the high speed).However, as mentioned above, it would be wise to run the program at low speeds when unconfident with assembler in general. Therefore the 'go' feature will only be useful for those uninterested in the individual effects of each instruction.