Testing your emulator

This manual contains instructions on how to compile 6502 instructions into byte-code, and how to import them into your emulator.

Installing DASM

We will use the DASM assembler to assemble 6502 instructions. It is available on Blackboard, and on `http://www.atari2600.org/DASM/`. The archive contains several interesting files:

`bin/Linux/dasm.linux-i386-elf-glibc22`	Assembler
`DASM/doc/DASM.txt`	Manual

The assembler needs to be set to 'executable' before it can be used. Also, it is practical to move it to an easier location. For instance:

```
mv bin/Linux/dasm.linux-i386-elf-glibc22 dasm
chmod a+x dasm
```

The last command sets the 'executable' bit. You now have the DASM assembler in the form of an executable called `dasm`. You can run it by typing:

```
./dasm
```

Using DASM

Also available on Blackboard is `DASMextras.zip`. This archive contains examples (`*.c64`) as well as an x86-assembly routine to read the assembled bytecode into a 64k memory block (`readimage.s`). The follow code is in `basic.c64`:

```
processor 6502

; start at 80:00
org $8000

; insert your code here

; STP
dc $de
```

The above program performs the following:

1. `processor 6502` informs DASM that we will use 6502 assembly.
2. `org $8000` will skip the first 8000h bytes to end up in the user section (see manual). We will use the convention that the first byte defined is where the program execution (PC) will start. In this program, that's the STP instruction.

3. `dc $de` is the STP instruction, which the 6502 officially doesn't support.

To convert this program into byte-code, run DASM as follows:

```
./dasm basic.c64 -oimage.bin
```

which will create `image.bin`. The format is described in the DASM manual, but the following section will explain how to use `readimage.s` to read this format for you.

Importing into your emulator

When you link along `readimage.s`, you gain access to the 'readimage' routine. Use it as follows:

```
.bss
mem:  .skip 65536

.data
filename:  .asciz "image.bin"

.text
pushl $mem
pushl $filename
call readimage
addl $-8,%esp
```

and 'mem' will be initialised with the binary image and the PC entry point (FF:FC/FF:FD) will point to the origin used (80:00 in the example of the previous section).

More examples

Other examples are included, and may require slight modifications to get working. For instance, `root.c64` contains a routine to calculate the square root of a number, but you will need to define the input values and let your emulator print the result.

More examples can be downloaded off the web, but may need similar patching as well. Also, downloaded examples often need the `processor 6502` line added, as well as a `STP` instruction.

Finally, you can of course write your own examples and small test routines by extending `basic.c64`. The DASM manual contains any syntax details you might need.