SNASM68K Console Notes

Overseas Customers

We are only able to supply power supplies for 240V 50Hz operation so you will need to obtain a 9V 1A PSU if you are outside the UK. The unit has an internal bridge rectifier so the polarity of the PSU is unimportant.

Installation

If you haven't used SNASM before you should follow the standard instructions for installing the PC hardware and software.

Whenever connecting the cable or changing any switch positions it is important that power is disconnected from both the target systems and Console interface.

CAREFULLY plug the small, black interface unit into the target interfaces 68000 socket making sure that the notches on both sockets are aligned (Pin 1 is marked on the case of the black interface unit). Severe damage to both devices can result from incorrect orientation.

Connect the ribbon cable between the main unit and the interface unit. Plug the SCSI cable into both the PC and the Console interface. Then, with any RAM or EPROM that may occupy the lower 4Mb on the target deactivated or removed, and the main units from panel switch set to '1' connect the power supply to the main unit and then to the target interface. When the target is turned on the write-prot light on the main unit should light for about a second then go off. The main unit should then have active, pwr-on and dc-pwr LEDs all lit and you should be able to communicate with it from the PC using the link test utility (LTU.EXE).

This unit contains 1Mb of static RAM for EPROM cartridge emulation, and 32K of battery backed RAM, the top 1K of which is used by the link software. The main 1Mb of RAM can be mapped in on any 1Mb boundary. The static RAM is mapped at $200000 and cannot be moved (Panther users see note below).

Whilst the processor is running code in the units internal EPROMs the main RAM is write enabled and the write-prot LED on the front of the unit is extinguished. Whilst user code is being run from the main RAM this RAM is write protected and the write-prot LED lights up.

If the front panel switch is used to de-activate the main unit you will be able to use the target machine as if the interface wasn't connected. When the switch is moved to the enable position the target machine will have to be reset to start the SCSI software.

There is a file on the SNASMZ80+UTILS disk called startup.68k which should be placed in your program before any other code. This piece of code sets up the 68000 vectors to point at either handlers within your code or to error handlers in the Console interface's ROM depending on the value of the Keep Traps variable which you should set at the start of your code. The file called except.68k contains the definitions of the exception labels which in the finished product are usually placed at all the same address and a piece of code is usually placed there to cause a reboot. If you want to use any interrupts in your program remove the label for this exception from this list. Panther users do not need this extra disk and can ignore this information.

There are some services provided by the software in the interface which can be called from within your code to give you further control over the debugging environment.

The entry point for these is 16 bytes into the interface workspace so you need to do a jsr to the base address plus 16 bytes to use these services.

D0 holds the service number. Registers are as below

Service 0 - Handle more exceptions
D1 - vector number at which to start
D2 - number of vectors to change

This causes further exceptions to switch execution to the SCSI software. The following piece of code causes allows you to enter the SCSI software by patching traps 1-F.

```
moveq.   #0, d0          ; user service 0
move.l   #33,d1          ; start at offset 33 (trap1)
move.1   #15,d2          ; set 15 vectors

jsr        $208000+16    ;$c00000+16 on Panther
bne        Error         ;zero flag set on success
                         ;could also check for d0 set to 0
```

Service 1 - Change status register marks
D1 - low word=ExSROrMask
high word=ExSRAnd Mark
D2 - low word=TpSROrMask
high word=TpSRAndMask

At SCSI software uses two words designated ExSRAndMask and ExSROrMask respectively. On entry to the exception handler routine these are 'anded' and 'ored' with the status register, thus allowing manipulation of the sr at that point by setting these flags prior to an exception. Similarly there are two words designated TpSRAndMask and TpSROrMask which perform the same function upon the execution of a trap0.

By default all these variables are initialised on entry to the downloader to ensure that when the downloader is invoked all interupts are switched off.

When you come to produce an image of your finished product you should set the KeepTraps variable false and assemble your code with the pure binary mode wet which will give you a binary image of the final ROM.

The lower eight bytes of the memory map are moved into the SSP and PC at reset when the code is in ROM so you should set up these and also, whilst testing you should ensure that the initial values of these registers are set using the REGS directive as shown in the SNASM manual.

Configuration File

The file snbug658.cf? (where ? is the target number, usually 7) contains information on which areas of memory can be read and written without causing problems. You should edit this file to contain the memory ranges for all of the target machines ROM and RAM areas. There is more information on this in the manual in the section on the debugger.

DIP Switches

A 1-4 Start 1Mb address of main RAM (1->A23,4->A20)
5-8 End 1Mb address of main RAM (1->A23, 4->A20)

If switched 4 and 8 are moved to the up position the main RAM will reside at 1Mb so the unit can be operated with target RAM or EPROM in the lower 1Mb with the main unit enabled. Be very careful when using the facility, to setup the switched and insert the cartridge before powering up either device.

B	1-3	SCSI ID (7 by default)
	4-5	Unused
	6	Power on reset (On by default)
	7	Enable IO block (On by default)
	8	Unused

The default positions of the switches are:-

A	1-8	All on (Ram at 0 for 1Mb)
B	1-3	All on (SCSI ID = 7)
	4-5	Off
	6	On
	7	On
	8	Off

Z80 Notes

The Z80 assembler is fully functional but does not currently support linking.

The syntax is mostly the same as the 68000 assembler so read the manual for information on pseudo ops, macros, etc. here are some exceptions:

- Local labels start with an at (@) but the /I switch changes this to a $
- Both $ and * can be used to represent the current PC value.
- Hex numbers are denoted by a trailing H and binary by a trailing B.
- You can instead prefix hex numbers with a hash (#)

Extras:

- You can abbreviate EX AF,AF to EXAF, JR NZ to JNZ
- PUSH and POP can take multiple registers
- You can omit the initial A in ADD,ADC and SBC
- Opt hd64180+ turns on the extra 64180 instructions.

Note:

- DB and DW are the only define bytes/words directives

To assemble PROG.Z80 to a pure binary file you would use :-
snasmz80 /p prog.z80,z80prog.bin

For a list of Z80 switches and options to snasmz80 with any options.