

Configure UC5282 NVRAM

Configure uC5282 NVRAM

Overview of Steps

- Create a 'cramfs' file system image of the netboot binary image. This is a compressed version of the netboot binary image.

Note: Alternately, with more recent versions of rtems (Example: 4.9.x) , you can directly flash the netboot.flashing.bin file.

- Store this on a machine with a tftp server that can access your target
- Connect the target (Example: a BPM chassis that has the Coldfire module in it) to your desktop or laptop.
- Set the IP address environment variable and run the tftp server.
- Transmit the file system image to the target.
- Program it into flash memory
- Using the uC5282 uCbootloader, set the environment variables
- Boot the image.

(Much of this document is from an email from Till, 13 March 2008)

Commands

The syntax to set variables is:
B\$ setenv ENVVAR value

To clear an environment variable:
B\$ setenv ENVVAR

To show a list of all variables:
B\$ printenv
To show the value of a single variable:
B\$ printenv ENVVAR

Detailed Steps

1 Generate directory hierarchy on host computer. Simplest case - just a root directory containing the 'netboot' binary image:

```
mkdir cramfs_top  
cp /afs/slac/package/rtems/<version>/target/ssrApps<lvl>/m68k-rtems/uC5282/img/netboot.flashing.bin cramfs_top/netboot.bin
```

Note: If your board is a 80MHz version of Coldfire, then you want to use netboot and rtems from 4.9.4/target/rtems_p1/ssrApps_p3/

2 Generate cramfs image (need mkcramfs utility) on host computer

```
mkcramfs cramfs_top cramfs.img
```

NOTE: in spite of the coldfire being a big-endian machine and cramfs internal data structures being susceptible to endianness, uCbootloader seems to expect a LITTLE-endian cramfs image (which is directly created by mkcramfs on a little-endian machine such as a x86 box)

Note: For more recent versions of the rtems (for example 4.9.x), one can tftp in netboot.flashing.bin directly, without having to compress it first.

3 Connect to serial port

Connect serial port (Example: on BPM chassis) to serial port on your computer. (It must be a straight-through cable.) Connect to your local serial port using a terminal emulation program (Tera Term Pro on Windows works).

Note: If connecting through DIGI Terminal Port Server, connect the serial port of device/chassis containing Coldfire module, to one of the serial ports of the DIGI.

To use iocConsole program to do serial communications with the DIGI:

(a) Add your Coldfire EIOC to screeniocs file, the configuration file for iocConsole.

For development environment, this file is in /afs/slac/g/lcls/epics/ioc/iocCommon/screeniocs.

This file lets association of a DIGI Serial Port number with your <coldfire_eioc_nodename>

(b) From a development linux host, such as lcls-dev2:

```
> source /afs/slac/g/lcls/tools/script/ENVS.bash
```

```
> iocConsole <coldfire_eioc_nodename>
```

Example: iocConsole 134.79.219.81

Power up the Coldfire device/chassis (Example: BPM chassis).

4 At the uCbootloader prompt B\$, set the IP address environment variable, for example:

```
B$ setenv IPADDR0 134.79.219.81
```

Start the tftp program:

```
B$ tftp
uCTFTP Console 1.0 is running ...
```

5 Copy the cramfs.img to uc5282 target. (Your host must have a tftp server and must be able to access your target.)

(These instructions are for using tftp. To use the serial connection, use the uCdimmm Coldfire 5282 manual)

From your host:

```
hostname$tftp 134.79.219.81
tftp> verbose
Verbose mode on.
tftp> binary
mode set to octet
tftp> tra
Packet tracing on.
tftp> put cramfs.img
```

NOTE: Use 'put netboot.flashimg.bin' if you are directly tftp'ing the binary image (and not the compressed image) per Step 2.

You will see messages like these in your host session:

```
putting cramfs.img to 134.79.219.81:cramfs.img octet
Sent 270336 bytes in 20.2 seconds []
tftp>
```

You will see messages like these on your serial session:

```
.....
done
```

It sometimes times out and takes a few tries before it works. In your serial session, press ESC to exit tftp and return to the B\$ prompt

6 Program the image into flash memory using 'program' command at B\$ prompt:

```
B$ program
erase... done
write... done
```

Select boot file:

```
B$ setenv KERNEL 0:netboot.flashimg.bin
```

The board has now been flashed. Next you can do one of three things (four things counting nothing):

- (7a) Test that you can boot the netboot image.*
- (7b) Set up for production. (Do this only if you feel confident that the flashing worked.)*
- (7c) Boot up to test on development.*

7a To boot the netboot image, set the necessary environment variables and type 'go' at the B\$ prompt:

```
setenv KERNEL 0:netboot.flashimg.bin
setenv BP_SRVR 134.79.19.29
setenv BP_FILE afsnfs2:/afs/slac:package/rtems/4.9.1/target/ssrlApps/m68k-rtems/uC5282/bin/rtems.ralf
setenv NETMASK 255.255.252.0
```

```
B$ printenv
FACTORY=Arcturus Networks Inc.
REVISION=uC5282 Rev 1.1 16MB External Flash
SERIAL=X49E5FA37-01080
CONSOLE=ttyS0
KERNEL_ARGS=root=/dev/mtdblock0 quiet
HWADDR0=00:06:3B:00:F0:80
FW_VERSION=010808012
_0=10000000:1000000:RW
CACHE=on
KERNEL=0:netboot.flashimg.bin
BP_SRVR=134.79.19.29
BP_FILE=afsnfs2:/afs/slac:package/rtems/4.9.1/target/ssrlApps/m68k-rtems/uC5282/bin/rtems.ralf
IPADDR0=134.79.219.113
NETMASK=255.255.252.0
```

Note: There is a slight difference in setting up the flash partition for a 4 M Coldfire and a 16 M Coldfire:

For 4M Coldfire, the partition as above will work - so nothing needs to be set up specifically.

For 16M Coldfire, the partition must be setup as follows:

`_0=10000000:100000:RW`

`_1=10100000:F00000:RW`

Read the manual /afs/slac/package/rtems/<version>/src/ssrlApps<lv>/netboot/README.uC5282 which explains how to deal with 16M of flash.

```
B$ go
uncompress... go! 0x40000
```

```
RTEMS bootloader by Till Straumann <strauman@slac.stanford.edu>
$Id: netboot.c,v 1.33 2009/02/05 08:39:20 strauman Exp $
CVS tag $Name: rtems-4-9-1-p2 $
```

Press 's' for showing the current NVRAM configuration
Press 'b' for manually entering filename/cmdline parameters only
Press '@' for continuing the netboot (BOOTP flag from NVRAM)
Press 'd' for continuing the netboot; enforce using BOOTP
Press 'p' for continuing the netboot; enforce using BOOTP
but use file and cmdline from NVRAM
Press 'm' for continuing the netboot; enforce using NVRAM config
Press 'R' to reboot now (you can always hit <Ctrl>-x to reboot)
Press any other key for this message

7b To set up for production and using DHCP/BOOTP, set these environment variables:

```
B$ printenv
FACTORY=Arcturus Networks Inc.
REVISION=uC5282 Rev 1.1 16MB External Flash
SERIAL=X4747745D-01DC9
CONSOLE=ttyS0
KERNEL=0:netboot.flashimg.bin*
KERNEL_ARGS=root=/dev/mtdblock0 quiet
HWADDR0=00:06:3B:00:AD:C9
FW_VERSION=010808004
_0=10000000:100000:RW
CACHE=on
(next three are optional; depends whether you want the ioc to boot up on its own the first time; also depends on whether or not dhcp will be set up in advance)
DO_BOOTP=Y
AUTOEXEC=2
AUTOBOOT=2
```

7c To boot up and test on dev, at the uCbootloader prompt B\$, set the rest of your environment variables.

```
B$ printenv
FACTORY=Arcturus Networks Inc.
REVISION=uC5282 Rev 1.1 16MB External Flash
SERIAL=X4747745D-01DC9
CONSOLE=ttyS0
KERNEL=0:netboot.flashimg.bin
KERNEL_ARGS=root=/dev/mtdblock0 quiet
HWADDR0=<macaddress>
FW_VERSION=010808004
_0=10000000:100000:RW
CACHE=on
IPADDR0=<ipaddress>
DNS_SERVER=134.79.151.12
NTP_SERVER=134.79.151.13
NETMASK=255.255.252.0
GATEWAY=134.79.219.1
HOSTNAME=<hostname>
BP_PARM=INIT=/boot/g/lcls/epics/iocTop/<appname>/<tagname>/iocBoot/<eiocname>/st.cmd
LOGHOST=syslog-host
DNS_DOMAIN=slac.stanford.edu
BP_FILE=afsnfs2:/afs/slac:package/rtems/4.9.1/target/ssrlApps/m68k-rtems/uC5282/bin/rtems.ralf
BP_SRVR=134.79.19.148
DO_BOOTP=N
```

Appendix (Keep it alive!)

Here are some useful links that discuss flashing 16M Coldfire and 80 MHz Coldfire:

- [Till's notes for supporting 80 MHz Coldfire](#)
- More of [Till's release notes for supporting 80 MHz Coldfire](#)
- Till's uC5282 Readme file in /afs/slac/package/rtems/4.9.4/src/ssrlApps_p3/netboot/README.uC5282

Share your Coldfire experiences, troubles, gotchas here: