Boot Linux Kernel From NFS (NFSboot)-buildroot

Linux

Exported on 06/16/2020

Table of Contents

1 Introduction

This document discusses how to boot the Linux Kernel form NFS (Network File System) using the **DTB**, **vmImage** and the **rootfs** , a brief summary is shown as below:

- **DTB (15KB):** Board DTB file for Linux booting, user could load DTB file via TFTP
- **vmImage (5.2M):** Linux kernel file, user could load the vmImage file via TFTP
- **rootfs(38M):** Root filesystem image, in the page, rootfs loading isn't required , target board can mount the **rootfs folder**(in the host PC) through NFS

2 Configure and Build the Source Code

Configure the Linux Kernel configuration to enable the Network File Systems(NFS) support and generate a vmImage without the the RAM File System(RAMFS).

2.1 Configure Linux Kernel

Run **make linux-menuconfig** and configure the kernel as follows:

• 2.1.1 Enable The NFS support in Linux Kernel

```
File systems  --->
   [*] Network File Systems  --->
       <*>   NFS client support
       <*>     NFS client support for NFS version 2
       <*>     NFS client support for NFS version 3
       [*]   Root file system on NFS
```

• 2.1.2 Separate RAMFS form Linux Kernel Image

```
General setup  --->
   [N] Initial RAM filesystem and RAM disk (initramfs/initrd) support
```

Note: As the ARM File System will be linked into Linux Kernel Image in default configuration, We should disable this feature manually to generate a Image file without the RAMFS.

2.2 Configure Buildroot

Use the default buildroot configuration.

2.3 Build the Source code

Run "make" in buildroot directory, the target files (DTB, vmImage and rootfs) will be generated in the directory /output/images/rootfs.tar sc589-mini.dtb vmImage

3 Set Up NFS and TFTP Server

NFS and TFTP Server should be prepared before booting the linux kernel via NFS.

3.1 Set up the TFTP Server

Refer to the section **"Set Up the TFTP Server"** in SC5xx ezkit Linux quick start guide[1] to prepare the TFTP server.

3.2 Set up the NFS Server

- **Run the following command to install the NFS server in Ubuntu:**

```
$ sudo apt-get install nfs-kernel-server
```

- **Set up the NFS shared directory: NFS Root File Systems**

```
$ sudo mkdir /rootfs/
$ sudo chmod 777 /rootfs/
```

- **Add the following configuration field in /etc/exports.**

```
$ sudo vim /etc/exports
/rootfs *(rw,sync,no_root_squash,no_subtree_check)
```

- **Save the "exports" and restart the NFS service.**

```
$ sudo service nfs-kernel-server restart
```

Now the NFS shared directory has been prepared valid.

- **Copy the Linux Kernel RootFS to the NFS shared directory.**

```
$ cd /opt/analog/cces-linux-add-in/1.3.0/buildroot-sc5xx-1.3.0/images/
$ sudo tar -C /rootfs/ -xf rootfs.tar
$ ls /rootfs/
bin  dev  etc  home  init  lib  lib32  media  mnt  opt  proc  root  run  sbin  sys  tmp  usr  var
```

Use the "ls /rootfs/" command and the files of Linux kernel rootfs will be come-to-able as shown above.

1 https://download.analog.com/tools/LinuxAddInForCCES/documentation/linux_add_in_user_guide_1.3.1.pdf

4 Configure U-boot to NFS Boot

The default U-Boot boot sequence looks for the boot image files on the Ramboot. The boot command is modified to load a *bootscript* using the built-in NFS boot command.
Boot into the U-boot and stopped.

```
sc# set nfsfile vmImage
sc# set rootpath /rootfs
sc# set ipaddr 10.100.4.50
sc# set serverip 10.100.4.174
sc# set nfsargs 'set bootargs root=/dev/nfs rw nfsroot=${serverip}:${rootpath} proto=tcp nfsvers=3
clkin_hz=(25000000) earlyprintk=serial,uart0,57600 console=ttySC0,57600 mem=224M'
sc# set nfsboot 'tftp ${loadaddr} ${nfsfile};tftp ${dtbaddr} ${dtbfile};run nfsargs;run addip;bootm $
{loadaddr} - ${dtbaddr}'
sc# set bootcmd run nfsboot
sc# saveenv
```

```
/rootfs        /*  rootfs is the NFS shared directory */
proto=tcp      /*  proto=tcp means the network of NSF booting using the
TCP/IP networking support */
nfsvers=3      /*  nfsvers=3 means NFS client support for NFS version 3,
which depends on the linux-kernel NFS configuration */
```

5 Boot Linux using NFS Boot method

Use the command "boot" in u-boot and the booting log will be shown as below:

NFS Booting log:

```
U-Boot 2015.01 ADI-1.3.0 (Sep 11 2018 - 12:01:02)

CPU:   ADSP ADSP-SC589-0.1 (Detected Rev: 1.1) (spi flash boot)
VCO: 450 MHz, Cclk0: 450 MHz, Sclk0: 112.500 MHz, Sclk1: 112.500 MHz, DCLK: 450 MHz
OCLK: 150 MHz
I2C:   ready
DRAM:  224 MiB
MMC:   SC5XX SDH: 0
SF: Detected N25Q512 with page size 256 Bytes, erase size 4 KiB, total 64 MiB
In:    serial
Out:   serial
Err:   serial
other init
Net:   dwmac.3100c000
Hit any key to stop autoboot:  0
sc # boot
dwmac.3100c000 Waiting for PHY auto negotiation to complete. done
Speed: 1000, full duplex
Using dwmac.3100c000 device
TFTP from server 10.100.4.174; our IP address is 10.100.4.50
Filename 'vmImage'.
Load address: 0xc2000000
Loading: ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##################################################################
         ##############################################
         3.9 MiB/s
done
Bytes transferred = 5435072 (52eec0 hex)
Speed: 1000, full duplex
Using dwmac.3100c000 device
TFTP from server 10.100.4.174; our IP address is 10.100.4.50
Filename 'sc589-mini.dtb'.
Load address: 0xc4000000
Loading: #
         2.8 MiB/s
done
Bytes transferred = 14580 (38f4 hex)
## Booting kernel from Legacy Image at c2000000 ...
   Image Name:   Linux-4.0.0-ADI-1.3.0
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    5435008 Bytes = 5.2 MiB
   Load Address: c2008000
   Entry Point:  c2008000
   Verifying Checksum ... OK
## Flattened Device Tree blob at c4000000
   Booting using the fdt blob at 0xc4000000
   Loading Kernel Image ... OK
   Loading Device Tree to cfe5c000, end cfe628f3 ... OK

Starting kernel ...
```

```
Booting Linux on physical CPU 0x0
Linux version 4.0.0-ADI-1.3.0 (test@madara.ad.analog.com) (gcc version 4.8.3 (Analog Devices Inc. ARM Tools
(6982e14d46b79b4d98b7172964d596c02615742a). Distributed as part of CrossCore8
CPU: ARMv7 Processor [410fc051] revision 1 (ARMv7), cr=10c53c7d
CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
Machine model: ADI sc589-mini
bootconsole [earlycon0] enabled
Memory policy: Data cache writeback
dump init clock rate
CGU0_PLL 450 MHz
CGU0_SYSCLK 225 MHz
CGU0_CCLK 450 MHz
CGU0_SYS0 112 MHz
CGU0_DCLK 450 MHz
CGU0_OCLK 150 MHz
CGU0_SYS0 112 MHz
Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 56896
Kernel command line: root=/dev/nfs rw nfsroot=10.100.4.174:/rootfs proto=tcp nfsvers=3 clkin_hz=(25000000)
earlyprintk=serial,uart0,57600 console=ttySC0,57600 mem=224M ip=10.100.4.50:1f
PID hash table entries: 1024 (order: 0, 4096 bytes)
Dentry cache hash table entries: 32768 (order: 5, 131072 bytes)
Inode-cache hash table entries: 16384 (order: 4, 65536 bytes)
Memory: 221848K/229376K available (3668K kernel code, 123K rwdata, 1356K rodata, 156K init, 85K bss, 7528K
reserved, 0K cma-reserved)
Virtual kernel memory layout:
    vector  : 0xffff0000 - 0xffff1000   (   4 kB)
    fixmap  : 0xffc00000 - 0xfff00000   (3072 kB)
    vmalloc : 0xce800000 - 0xff000000   ( 776 MB)
    lowmem  : 0xc0000000 - 0xce000000   ( 224 MB)
    modules : 0xbf000000 - 0xc0000000   (  16 MB)
      .text : 0xc0008000 - 0xc04f053c   (5026 kB)
      .init : 0xc04f1000 - 0xc0518000   ( 156 kB)
      .data : 0xc0518000 - 0xc0536e80   ( 124 kB)
       .bss : 0xc0536e80 - 0xc054c3f8   (  86 kB)
NR_IRQS:16 nr_irqs:16 16
GIC CPU mask not found - kernel will fail to boot.
GIC CPU mask not found - kernel will fail to boot.
sched_clock: 32 bits at 112MHz, resolution 8ns, wraps every 38177486839ns
Console: colour dummy device 80x30
Calibrating delay loop... 297.98 BogoMIPS (lpj=595968)
pid_max: default: 32768 minimum: 301
Mount-cache hash table entries: 1024 (order: 0, 4096 bytes)
Mountpoint-cache hash table entries: 1024 (order: 0, 4096 bytes)
CPU: Testing write buffer coherency: ok
Setting up static identity map for 0xc237d788 - 0xc237d7bc
devtmpfs: initialized
do_initcall_level level 0
do_initcall_level level 1
VFP support v0.3: implementor 41 architecture 2 part 30 variant 5 rev 1
pinctrl core: initialized pinctrl subsystem
NET: Registered protocol family 16
do_initcall_level level 2
DMA: preallocated 256 KiB pool for atomic coherent allocations
do_initcall_level level 3
L2C: device tree omits to specify unified cache
```

```
L2C-310 dynamic clock gating enabled, standby mode enabled
L2C-310 cache controller enabled, 8 ways, 256 kB
L2C-310: CACHE_ID 0x410000c9, AUX_CTRL 0x06040000
sc58x_init: registering device resources
sec init...
enabled
hw-breakpoint: Failed to enable monitor mode on CPU 0.
ADI DMA2 Controller
do_initcall_level level 4
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
i2c-bfin-twi 31001400.twi: Blackfin on-chip I2C TWI Contoller, regs_base@f4001400
i2c-bfin-twi 31001500.twi: Blackfin on-chip I2C TWI Contoller, regs_base@f4001500
i2c-bfin-twi 31001600.twi: Blackfin on-chip I2C TWI Contoller, regs_base@f4001600
pps_core: LinuxPPS API ver. 1 registered
pps_core: Software ver. 5.3.6 - Copyright 2005-2007 Rodolfo Giometti <giometti@linux.it>
PTP clock support registered
do_initcall_level level 5
Switched to clocksource cs_gptimer
NET: Registered protocol family 2
TCP established hash table entries: 2048 (order: 1, 8192 bytes)
TCP bind hash table entries: 2048 (order: 1, 8192 bytes)
TCP: Hash tables configured (established 2048 bind 2048)
TCP: reno registered
UDP hash table entries: 256 (order: 0, 4096 bytes)
UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
NET: Registered protocol family 1
RPC: Registered named UNIX socket transport module.
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
RPC: Registered tcp NFSv4.1 backchannel transport module.
do_initcall_level level 6
hw perfevents: enabled with armv7_cortex_a5 PMU driver, 3 counters available
futex hash table entries: 256 (order: -1, 3072 bytes)
jffs2: version 2.2. (NAND) ©2001-2006 Red Hat, Inc.
io scheduler noop registered (default)
ADI serial driver
adi-uart4.0: ttySC0 at MMIO 0x31003000 (irq = 20, base_baud = 7031250) is a ADI-UART4
console [ttySC0] enabled
console [ttySC0] enabled
bootconsole [earlycon0] disabled
bootconsole [earlycon0] disabled
loop: module loaded
adi-spi3 31042000.spi: registered ADI SPI controller spi0
adi-spi3 31043000.spi: registered ADI SPI controller spi1
m25p80 spi2.38: n25q512ax3 (65536 Kbytes)
3 ofpart partitions found on MTD device spi2.38
Creating 3 MTD partitions on "spi2.38":
0x000000000000-0x000000080000 : "uboot (spi)"
0x000000080000-0x000000600000 : "kernel (spi)"
0x000000600000-0x000004000000 : "root file system (spi)"
adi-spi3 31044000.spi: registered ADI SPI controller spi2
libphy: Fixed MDIO Bus: probed
```

```
stmmaceth 3100c000.ethernet: no reset control found
stmmac - user ID: 0x10, Synopsys ID: 0x37
 Ring mode enabled
 DMA HW capability register supported
 Enhanced/Alternate descriptors
        Enabled extended descriptors
 RX Checksum Offload Engine supported (type 2)
 TX Checksum insertion supported
 Wake-Up On Lan supported
 Enable RX Mitigation via HW Watchdog Timer
libphy: stmmac: probed
eth0: PHY ID 2000a231 at 0 IRQ POLL (stmmac-0:00) active
usbcore: registered new interface driver usb-storage
musb-hdrc musb-hdrc.1.auto: MUSB HDRC host driver
musb-hdrc musb-hdrc.1.auto: new USB bus registered, assigned bus number 1
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 1 port detected
musb-hdrc musb-hdrc.3.auto: MUSB HDRC host driver
musb-hdrc musb-hdrc.3.auto: new USB bus registered, assigned bus number 2
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 1 port detected
mousedev: PS/2 mouse device common for all mice
i2c /dev entries driver
Driver 'mmcblk' needs updating - please use bus_type methods
Synopsys Designware Multimedia Card Interface Driver
dwmmc_adi mmc.0: num-slots property not found, assuming 1 slot is available
dwmmc_adi mmc.0: IDMAC supports 32-bit address mode.
dwmmc_adi mmc.0: Using internal DMA controller.
dwmmc_adi mmc.0: Version ID is 270a
dwmmc_adi mmc.0: DW MMC controller at irq 91, 32 bit host data width, 1024 deep fifo
dwmmc_adi mmc.0: No vmmc regulator found
dwmmc_adi mmc.0: No vqmmc regulator found
dwmmc_adi mmc.0: 1 slots initialized
Blackfin hardware CRC crypto driver
bfin-hmac-crc 31001200.crc: initialized
bfin-hmac-crc 31001300.crc: initialized
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
TCP: cubic registered
NET: Registered protocol family 17
do_initcall_level level 7
ThumbEE CPU extension supported.
console [netcon0] enabled
netconsole: network logging started
IP-Config: Gateway not on directly connected network
stmmaceth 3100c000.ethernet eth0: Link is Up - 1Gbps/Full - flow control rx/tx
VFS: Mounted root (nfs filesystem) on device 0:12.
devtmpfs: mounted
Freeing unused kernel memory: 156K (c04f1000 - c0518000)
External imprecise Data abort at addr=0xb6fb6880, fsr=0x1c06 ignored.
Starting logging: OK
Starting mdev...
Starting watchdog...
Initializing random number generator... random: dd urandom read with 34 bits of entropy available
done.
```

```
Starting system message bus: done
Starting network...
/bin/sh: run-parts: not found
Starting sshd: /var/empty must be owned by root and not group or world-writable.
OK
Starting inetd: OK

Welcome to Buildroot
buildroot login: root
Password:


                          _____________________________________
        a8888b.           / Welcome to the buildroot distribution \
       d888888b.         /       _     _                           \
       8P"YP"Y88        /       | |   |_|            __  __ (TM)   |
       8|o||o|88  _____/        | |    _ ____  _   _ \ \/ /        |
       8'    .88       \        | |   | |  _ \| | | | \  /         |
       8`._.' Y8.       \       | |__ | | | | | |_| | /  \         |
      d/      `8b.       \      \____||_|_| |_|\____|/_/\_\        |
     dP   .    Y8b.       \   For embedded processors including    |
    d8:'  "  `::88b        \    the Analog Devices ADSP-SC5xx      /
   d8"         'Y88b        \_____________________________________/
  :8P    '      :888
   8a.   :     _a88P         For further information, check out:
 ._/"Yaa_:   .| 88P|
 \    YP"    `| 8P  `.
 /     \.___.d|    .'           - http://buildroot.org/
 `--..__)8888P`._.'  jgs/a:f    - http://www.analog.com/

Have a lot of fun...
# random: nonblocking pool is initialized
```

> "VFS: Mounted root (nfs filesystem) on device 0:12.
> devtmpfs: mounted"
> Means the Linux kernel mounts the filesystem using the NFS tool

6 NFS Rootfs Test

NFS rootfs can test via modifying the NFS shared directory in the host PC, and the same changes will be shown in the dorresponding directory of the target boards.

- **In the host PC**

```
test@madara:~$ cd /rootfs/root/
test@madara:/rootfs/root$ ls
test@madara:/rootfs/root$ date >date.log
test@madara:/rootfs/root$ ls
date.log
test@madara:/rootfs/root$ cat date.log
Mon Oct 15 17:12:27 CST 2018
test@madara:/rootfs/root$
```

We can get the **date.log** file in directory /roofs/root

- **In the target board**

```
# cd /root/
# ls
date.log
# cat date.log
Mon Oct 15 17:12:27 CST 2018
```

Simultaneously, the time in **date.log** on target board matched it in the host PC.