

VIA C3 Nehemiah Hardware Random Number Generator

Linux Driver & Test Utility Usage Guide

Version 0.73, September 30, 2003
Copyright (C) 2003 VIA Technologies, INC.

1. Summary

The high-performance hardware-based random number generator (RNG) function is implemented in the “stepping 3” and later versions of the VIA C3 Nehemiah processor. The kernel starts to have a driver supporting the RNG from version 2.4.23-preX, and 2.6.0-testX. This document describes how to update and patch the kernel and then make use of the RNG. We also offer a test utility with source code for users’ immediate evaluation. The information and the utility in this document are provided “AS IS,” without guarantee of any kind.

2. File description

The package contains 7 files as described below.

sample/relnote	330	04-28-03	16:55	utility release note
sample/bin/rngtest	37,187	04-29-03	16:58	utility binary
sample/bin/help	921	04-28-03	16:57	help file
sample/src/rngtest.c	10,027	04-29-03	15:22	utility source code
patch_2.4_kernel	2,732	09-29-03	09:43	patch file for 2.4 kernel
patch_2.6_kernel	3,247	09-29-03	09:43	patch file for 2.6 kernel
Readme.doc				this file

Users are advised to directly download the kernel source package version 2.4.23-pre1, 2.6.0-test1 or later from <http://www.kernel.org>.

3. Update kernel

Note: The RNG function is implemented in the “**stepping 3**” and later versions of the VIA C3 Nehemiah processor. Users can check the “**CPUID**” in “**system information**” when booting up or check “**stepping**” in `/proc/cpuinfo` file. The “**CPUID**” must be “**069x**”, and the “**x**” must be equal or greater than 3.

The following procedures should work on most Linux distributions, though we tested only on Red Hat Linux 9.0. Both 2.4.22-pre1 and 2.6.0-test1 or above kernels have RNG function implemented. We only use kernel 2.6.0-test5 as an example, and users can use any compatible kernel that has RNG function implemented.

(1) Install the kernel source

Run the following command to decompress the kernel source code.

```
# tar xjf linux-2.6.0-test5.tar.bz2
```

(2) Configure the kernel

Change the current directory to “**linux-2.6.0-test5**”.

```
#cd linux-2.6.0-test5
```

We also provide patch files for speeding up the generative rate of RNG

```
#patch -p0 < patch_2.6_kernel  
patching file drivers/char/hw_random.c
```

Then run the following command to configure the kernel.

```
# make menuconfig (xconfig or config)
```

There are some situations that may need your special attention.

- (a) Enable “**VIA C3-2 (Nehemiah)**” under “**Processor type and features**”. And if your platform is not a dual-processor system or above, disable “**symmetric multi-processing support**”.
- (b) Enable “**Intel/AMD/VIA HW Random Number Generator support**” under “**Character device**”, by selecting the built-in or module mode.

In case you see the following message, this is because the system built-in module tool cannot load the newly added module in 2.5 and 2.6 kernel series.

```
#modprobe hw_random  
modprobe : QM_MODULES: Function not implemented  
modprobe : Can' t locate module hw_random
```

Download and install “**module-init-tools-x.y.z.tar.gz**” (x.y.z the version) from URL: <http://www.kernel.org/pub/linux/kernel/people/rusty/modules/>.

(3) Rebuild the kernel

Run the following command to rebuild the kernel.

```
# make dep clean bzImage modules modules_install
```

Next, copy the newly built kernel to **/boot/**.

```
#cp arch/i386/boot/bzImage /boot/vmlinuz-test
```

If using the **GRUB** boot loader, add the following two lines to the **/boot/grub/menu.1st** file. Note you may need to modify the “**hda1**” according to your actual system settings.

```

title linux-test
kernel /boot/vmlinuz-test ro root=/dev/hda1

```

On the other hand, if using the **LILO** boot loader, add the following four lines to the **/etc/lilo.conf** file. Note you may need to modify the “**hda1**” according to your actual system settings.

```

image=/boot/vmlinuz-test
Label=linux-test
read-only
root=/dev/hda1

```

Run “**lilo**” and let the newly added boot configuration take into effect. On the screen you should be able to see a message like below.

```

Added linux *
Added linux-test

```

Finally, reboot the system and test the new kernel.

4. Verify success of driver enhancement

Reboot the system and choose the newly added “**linux-test**” label to boot. If **/dev/hwrandom** does not exist, run the following command to create one.

```
#mknod -m 644 /dev/hwrandom c 10 183
```

Next, run the following commands to confirm whether the RNG driver has been loaded into kernel. If not, verify if you have re-built the kernel correctly or if you have the right CPU model.

```

#dmesg | grep “hw_random”
hw_random hardware driver 1.0.0 loaded

```

We also offer a test utility for users’ immediate evaluation. For example, run the following command to generate 1MB of random data with silent mode.

```
#./rngtest -b 1048576 -s
```

The following tables show how long each RNG driver spend to generate 1MB of random data in different linux distributions and kernels.

	Run 1	Run 2	Run 3	AVG
Original RNG driver (1byte per instruction)	3 min 27 sec	3 min 27 sec	3 min 28 sec	3 min 27 sec
Original RNG driver (4 bytes per instruction)	53 sec	52 sec	53 sec	53 sec
Patched RNG driver	<= 1 sec	<= 1 sec	<= 1 sec	<= 1 sec

Note: Red Hat Linux 9.0 and kernel 2.6.0-test5 were used.

Original RNG driver (1byte per instruction)	4 min 6 sec	4 min 6 sec	4 min 6 sec	4 min 6 sec
Original RNG driver (4 bytes per instruction)	1 min 2 sec	1 min 2 sec	1 min 1 sec	1 min 2 sec
Patched RNG driver	<= 1 sec	<= 1 sec	<= 1 sec	<= 1 sec

Note: Red Hat Linux 9.0 and kernel 2.4.23-pre4 were used.

In the test utility, we add a new option, FIPS 140-1 test (*).

```

#./rngtest -f
FIPS 140-1 Test Result
### Monobit Test (Passed if between 9654 and 10346)
    PASS! There are 10131 bit 1 and 9869 bit 0!
### Poker Test (Passed if between 1.03 and 57.4)
    PASS! Poker value = 16.998400
### Runs Test
    Ones  Result | Zeros  Result
1: 2451  PASS | 2479  PASS (Required Interval 2267-2733)
2: 1213  PASS | 1277  PASS (Required Interval 1079-1421)
3:  662  PASS |  614  PASS (Required Interval 502-748)
4:  319  PASS |  291  PASS (Required Interval 223-402)
5:  146  PASS |  159  PASS (Required Interval 90-223)
6+:  179  PASS |  150  PASS (Required Interval 90-223)
    Test Result: PASS!
### Long Run Test
    PASS! No a run of length 34 or more of either 0s or 1s.

```

For more information of FIPS 140-1 test, refer to section 4.11.1 in the site

<http://www.itl.nist.gov/fipspubs/fip140-1.htm>.

5. Test configuration

The following configuration was used for test.

Motherboard	EPIA-M10000 (CLE266+VT8235)	
CPU	VIA C3 Nehemiah CPU 1 GHz (133x7.5)	
System Memory	128MB DDR RAM	
HDD	Quantum LM20500AT 20GB HDD	
OS	Red Hat Linux 9.0	
Kernel	2.4.22 plus 2.4.23-pre4 patch	2.6.0-test5