VoP, Real-Time, Linux and RTLinux

By
Vidyasagaran P

All Rights Reserved

MultiTech ⬤
Software Systems

95, 17th'B'Main Road
5th Block, Koramanagala
Bangalore – 560095, India
Tel: +91-80-2553 4471 / 76
Fax: +91-80-2553 6646
http://www.multitech.co.in

1   INTRODUCTION

Voice over Packet (VoP) is a hot technology and claims lot of advantages over the conventional "circuit switched" mechanisms for voice communications. Due to the digital nature of "packet switched" networks, the native analog voice signals are converted to digital packets for transmission. Such a conversion demands real-time performance and Linux as an operating platform lacks this property. A real-time version of Linux, RTLinux is capable of handling the situation. This article is a discussion on these issues.

2   VOP AND REAL-TIME

VoP is a technology where voice (usually received by a microphone) is converted from its native analog signal form to digital packets and transmitted over a packet switched network. At the receiving end, the received packets are converted back to analog signal and played out on a speaker (or similar equipment).

2.1   VOICE TO PACKET CONVERSION AND ITS REAL-TIME NATURE

The conversion of voice from analog to digital form generally follows the ITU-T, G.711 standard. Typically, this is done by a dedicated hardware or a digital signal processor (DSP).

The data generated by the above operation has a rate of 64 Kilobits per Second. However, there is a significant scope for reducing this rate by employing various compression (coding) techniques. ITU-T has defined standards like G.729 and G.723.1 for voice coding. The effective data rate can thus be reduced to less than 10 Kilobits per second. Such a drastic reduction in data rate is very helpful to transport voice over slower communication channels.

The voice coders (vocoders, in short) generate a stream of packets as they encode the samples. The encoded data is presented to the host processor at periodic intervals and the periodicity depends on the encoding algorithm chosen (refer to table 1). The encoded packets have to be read in by the host system at a rate not less than that at which they are generated. Failure to do so will result in packet overflow and loss in quality of reproduced speech at the receiving end. Only a system that can predictably respond to interrupts can handle this situation. Even if the microprocessor can respond to this demand, the host operating system may disable the interrupts for its convenience. (This is what happens with standard Linux)

Standard	Coding Method	Effective bit rate (kilobits/sec)	Inter-packet rate (milliseconds)
G.711	PCM	64	5
G.729A	CSA-CELP	8	10
G.723.1	ACELP	6.3	30

Table 1 – Data rates of typical Voice coders

2.2 PACKET TRANSMISSION

The packet generated in above section is transmitted over a packet switched network, which is typically an IP (Internet Protocol) network. There could be loss, delay, jitter (variable delay between successive packets) and out of sequence arrivals at the receiving end. The software has to take care of all such issues.

2.3 PACKET TO VOICE CONVERSION

The packets received by the receiver are converted back to Analog signal before playing out. Based on the encoding used for Voice to Packet conversion, the reverse process is applied here. This is a computation intensive process and again done with the help of DSPs. These decoders needs to be supplied with samples at the designated intervals or else they will under run, resulting in a bad quality of re-generated voice. It means that the software should be able to feed data to the decoder at the designated regular intervals. This interval is depends on the chosen vocoder algorithm (refer to table 1). Only an operating system that is capable of handling interrupts at a better rate than this required inter packet rate can handle the job.

3   LINUX – PLATFORM FOR VOP ?

Linux is an emerging platform (or already emerged?) for embedded computation and VoP is a typical case of an embedded application. The advantages of using the existing programming talents and the wide applications and tools are numerous.

Once it comes to the response times of the standard Linux, it becomes evident that Linux was never designed for this purpose. There are many places in the Linux Kernel where interrupts are blocked for long duration! Also Linux does not provide for prioritized interrupt handling. It doesn't matter much in the case of a general purpose OS, which Linux is.

It is possible that a network interface card is interrupting very often and we miss to read the samples presented by the vocoders. Table 2 shows how widely the interrupt response time of a standard Linux system varies under load. Obviously, we would have lost a large number of packets if we were to use Linux as the Operating platform for VoP applications.

	Plain Linux PC Interrupt Response Time (milliseconds)
Worst Interrupt response time during 10 iterations. Each iteration interrupted several times	852
	1,128
	1,068
	732
	588
	4,849,020
	3,852,312
	1,976,604
	6,048
	4,984,968
Worst interrupt response	**4,984,968**

Table 2 – Some relevant data on Linux Performance

Note: The target hardware used for measurement was - PC Platform – IBM PC compatible Intel Celeron CPU at 150 Mhz.

As the table (Table 2) indicates, Linux response times vary widely. Also the worst-case response is unacceptable for real-time applications. These observations clearly indicate that Linux is not suitable for real-time applications.

4   RTLINUX

RTLinux (Real-time Linux) is based on a work done at the University of New Mexico by Dr. Victor Yodaiken (CEO and co-founder of "FSM Labs" http://www.fsmlabs.com/). The idea behind this product is to use a dual-kernel-architecture and make a real-time Linux environment. A small real-time kernel takes over the complete control of the hardware and runs Linux Operating system itself as the lowest priority task. By carefully designing an application in such a way that only the essential real-time activity is performed within the real-time portion, we can exploit the best of both worlds.

In a typical RTLinux system, real-time portion of the application may collect the data. This ensures that the samples that are collected do not over flow. Such data collected are put into a real-time FIFO. This FIFO is accessible from a normal Linux application. The data read from the FIFO is processed by a normal Linux application.

In the VoP application, if the voice coder samples are handled in the real-time mode, they would not miss any samples.

4.1   RTLINUX PERFORMANCE

Table 3 indicates the typical performance of RTLinux. Please note the more deterministic nature of the interrupt response from RTLinux.

	Plain Linux PC	RTLinux PC	RTLinux 860	RTLinux MIPS
Interrupt Response	852	48	120	52
	1,128	44	136	148
	1,068	28	152	116
	732	56	200	152
	588	64	224	148
	4,849,020	40	232	120
	3,852,312	32	240	136
	1,976,604	36	216	168
	6,048	48	224	168
	4,984,968	56	200	116
Worst Interrupt Response	4,984,968	64	240	168

Table 3 – RTLinux Performance

Note: - all timings are in microseconds

The target hardware used for measurements were:
- PC Platform – IBM PC compatible Intel Celeron CPU at 150 Mhz
- MPC860 – proprietary hardware with MPC860 CPU running at 40Mhz
- MIPS – MIPS Atlas SBC with 4Kc CPU running at 80Mhz

As Table 3 indicates, it is clear that the interrupt response times of RTLinux is less and worst-case response is acceptable for many real-time applications. This makes RTLinux eligible for real-time applications.

5   <u>CONCLUSION</u>

The idea RTLinux can overcome the limitations of Linux for real-time applications. It is a serious contender in the Open Software arena for real-time applications.

6   <u>REFERENCES</u>

1. Voice and Data Internetworking – Gil Held – McGrawHill
2. Voice Over IP – Uyless Black – Prentice Hall Series in advanced communication technologies