Raw data: Interrupt latency measurements.

Poul-Henning Kamp [1]
The FreeBSD Project
Revised 1998-07-02

ABSTRACT

This paper presents some raw data on interrupt latency measurements on FreeBSD-current as of July 2nd 1998.

Objective

The objective of this experiment was to determine and characterize the interrupt latency in the FreeBSD kernel when under typical circumstances.

Setup

The experimental setup consist of the following parts:

Asus TX97 PC(AT) motherboard. This is a Pentium class motherboard of later vintage. Most of the standard peripherals are integrated onto the motherboard, for instance IDE/ATA/ATAPI controllers, floppy disk controller, async serial port and parallel printer ports.
`http://www.asus.com`

AMD K6/233 CPU A Pentium compatible CPU.
`http://www.amd.com`

ISOTemp OCXO-112-13 Ovenized Quartz Crystal Generator. This is a unit for which I do not have the actual specs. This quality of this unit is not critical for these measurements.
`http://www.isotemp.com`

Virtual Computer Corporation "HOT1" reprogramable computing device. This device (called a *xrpu*) is a PCI bus card which contains a FPGA into which userdesigned circuits can be programmed. For this experiment a 22 bit counter and a 22 bit latch were implemented. The counter runs from at frequency of 90.112 MHz which is derived from the OCXO signal using a PLL chip on the card.
`http://www.vcc.com`

Motorola UT Oncore GPS receiver evaluation kit. This is not really necessary, any source of a low frequency TTL squarewave would do, but I had this one wired up already. The high precision of the Pulse-Per-Second output makes it easier to detect spurious noise in the dataset.
`http://www.mot.com`

NCR 810 based SCSI controller, *Adaptec 5930 155Mbit/sec UTP5 ATM controller*, *NE2000 compatible 16bit ISA bus ethernet card*and *Quantum ProDrive 1800S SCSI diskdrives (2)* These devices complete the system.

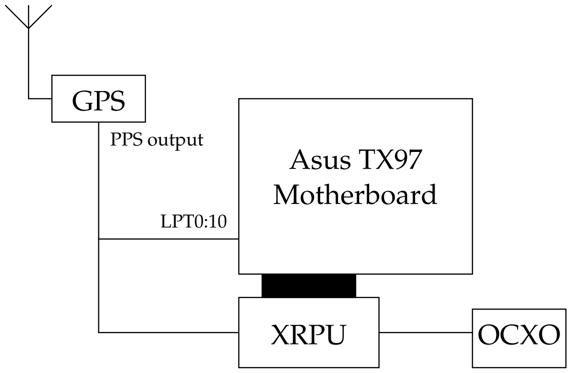

Fig. 1: The hardware setup.

On the software side the computer operated under *FreeBSD-current* as of 1998/07/02 06:00 UTC, with an improved driver for the xrpu unit.
`http://www.FreeBSD.org`

Theory

The system operates using a *timecounter* which is implemented in the *xrpu* device. This timecounter operates from a very stable frequency of 90.112 MHz derived from the OCXO. In the xrpu is also implemented a latch which will capture the reading of the timecounter at the rising edge of an external signal.

The Pulse-Per-Second output from the GPS receiver is connected to pin 10 on the parallel printer port on the motherboard and to the latch input on the xrpu unit.

The FreeBSD kernel is configured with the *lppps* device-driver, which will record and store a timestamp for each interrupt on the parallel printerport.

Since this timestamp is constructed from the timecounter in the xrpu, the timedifference between the recorded timestamp from the latch in the xrpu and the lppps driver represents the time it took from the rising edge of the pin in the parallel printer port until the FreeBSD kernel had entered the applicable interrupt service routine plus various systematic delays.

[1] This work was not sponsored by anybody. Poul-Henning Kamp was supported by his own daytime job. He would have loved to do this for some sponsors money instead.

During the run of this experiment a few different loads were put on the system, in order to get a more representative result.

The activity on the machine was:

samples 1-1100: various command-line work.

samples 1100-1500: idle.

samples 1500-2100: "cvs update" from a cvs repository mounted with NFS over the ATM link.

samples 2100-2500: light command-line work.

samples 2500-4400: "make world"

Results

Figure 2 shows a plot of the data samples gathered.

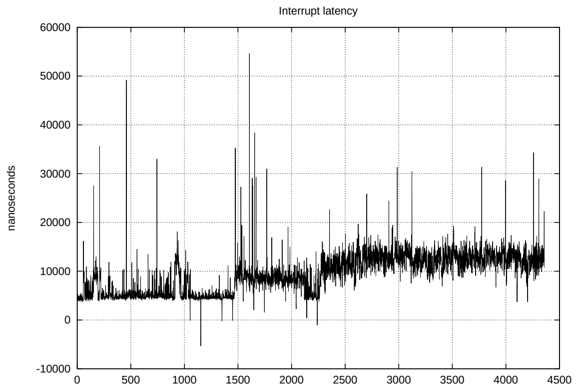

Fig. 2: The raw numbers

Figure 3 shows the same data plotted as a histogram.

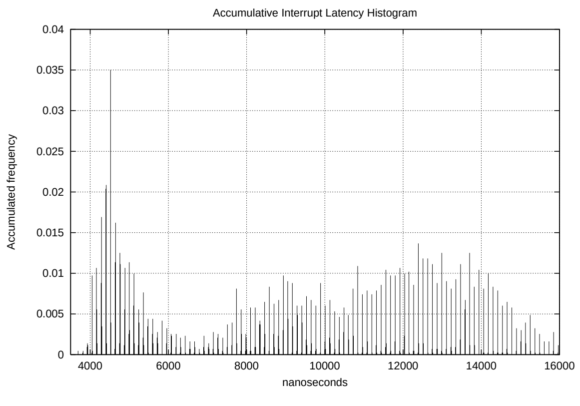

Fig. 3: Histogram

Finally Figure 4 shows the same data, this time plotted as an accumulative histogram.

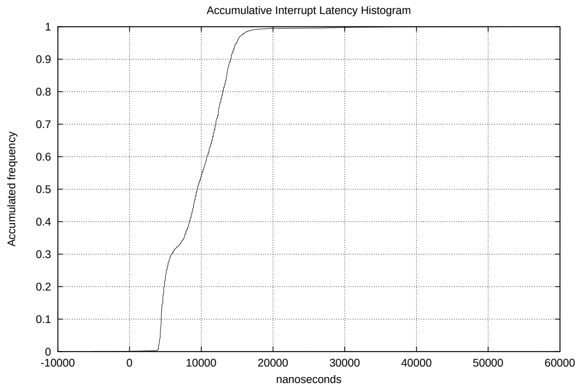

Fig. 4: Accumulative Histogram

Conclusion

It is hard to draw any conclusions from the data at present, which is why this paper has been labeled *raw data* in the title.

There is evidently some noise in the data, some of this may be caused my metastability problems in the design implemented in the xrpu.

The author is no VHDL wizard, and has probably broken all applicable rules about multiple clock domains in the design used. This would be a self-correcting condition which would simply result in one measurement being wrong. The exact magnitide and frequency of this noise-source has not yet been characterized, but other experiments indicate a frequency of less than one percent.

It is very important to realize that there is a potentially large systematic error on all the measurements due to a number of different effects and that it therefore would be wrong to conclude that four microseconds is the shortest time possible from stimulus to response for a FreeBSD kernel.

One of the effects not accounted for is the time it takes to read the xrpu timecounter (which the lppps0 driver does) compared to the zero delay of the latch in the xrpu device. This delay is probably less than 600 nanoseconds.

The parallel printerport is a legacy ISA device, and despite the fact that it is integrated into the chipset on the motherboard, it is still connected to the ISA bus. This is evident from the fencepost pattern in figure 3, which correspond to a ISA bus frequency of approx 8 MHz.

It is not know if there is any deglitching circuitry in the implementation of the parallel printerport, if there is that would probably cause some number of ISA clock cycles delay before the interrupt is delivered.