Using slow hardware for fast signal processing

M.G. van der Horst

Computers become obsolete very fast; a computer that was brand new 5 years ago is considered to be an antique today. This is caused by a trend called Moore's law that, states that the processing and memory capacity of computers double roughly every one and a half years. This means that a 5 year old computer performs almost 1/4 as well as a brand new one.

While this rate of progress is amazing, there is another technology that is making even more progress: communication technology. The speed at which we are now able to communicate data is increasing at an astonishing rate. Just look at the advancements in optical discs drives for computers. The first CD-rom drive was introduced in the early eighties, but the speed at which it can be read has not developed at a rate anywhere near Moore's law. However, the developments in recent years have accelerated; the increase in transfer speed of the DVD, which was introduced at the end of 1996, is now following Moore's law quite nicely. Even faster, the speed of computer networks is increasing by a factor of 10 at each step. A lot of companies have not even finished upgrading their 10 Mbps (Megabits per second) networks to 100 Mbps, but networks of 1,000 Mbps are already available and being used. In fact 10,000 Mbps networks are currently in the works. And fiber optics will make even greater network speeds available. In short, if the current trend in communication technology continues our ability to communicate data will eventually exceed our capacity to process it.

All these advances in communication technology rely on signal processing to work. Scratches on a DVD, a noisy line or atmospheric disturbances, are all examples of conditions that may cause distortions in the signal received by our computer or cellphone. These devices contain hardware which uses signal processing algorithms to correct and filter these distortions. The design of this hardware is the focus of our research: since the data rate increases at a greater rate than the hardware's capacity we want to be able to design the hardware in such a way that we can handle almost any difference in communication rate and processing rate.

Furthermore, with the results of this research we will be able to use slower (and therefore cheaper) processing hardware to handle our current signal processing needs. In effect, besides solving a future problem, it will allow us to construct cheaper signal processing hardware.

So the question we want to answer is: How can we design a scalable, efficient implementation from the specification of any signal processing algorithm? The combination of hardware and software, called the implementation, has to be scalable, because we want to be able to handle any difference between the processing hardware and the communication rate. And, even though the slower

hardware is cheaper, we still want efficient implementations: every bit of hardware costs money and therefore we want to use it to the maximum extend possible.

We intend to achieve our goal by looking at hardware opertating in parallel. The idea is quite simple: if a single (slow) processor cannot handle all the work we will have to use more of them. But while it looks simple, it has the same problems as putting more people on a job. If you put two people on a job then they can generally finish it in half the time. But as anybody knows this does not scale well: if you put one hundred people on the same job the chances are that it will take more than one hundredth of the time it would take one person. This can be cause by many factors: the communication between the people will cause overhead, as well as the fact that they have to wait for others to finish before they can start their own part of the job. There is also the possibility that the work just cannot be divided over 100 people and that some of them will have nothing to do. And any manager knows that hiring employees that do nothing or spend most of their time waiting for others is a waste of money. These same problems occur when trying to divide a signal processing algorithm over multiple processors, but with careful design these problems can be minimized or even avoided.

There are several possible outcomes of this research and each has it advantages. Perhaps we are not able to find a design method that applies to all signal processing algorithms. In this case the research will produce good design methods for large classes of signal processing algorithms. Which is useful to designers that need to implement an algorithm in such a class. Another possibility is that we find a method applicable to any algorithm, but that there exist specialized design methods for some algorithms which give more efficient solutions. In this case the designers will have a general method that always works, which is very useful. And in cases were extra efficiency is needed they can use the specific design methods. But when the research is completely successful we will have a design method that always works and produces good implementations. With this method we will be able to construct cheaper signal processing hardware, and the difference in the development rate between communication and processing technology will not become a problem.