

Homogeneous multiprocessing for the masses

Paul Stravers

Philips Research

Processor architectures have reached a point where it is getting increasingly hard to improve their performance without resorting to complex and exotic measures. Polack observed in 2000 that Intel processors had been “on the wrong side of a square law” for almost a decade. Embedded processors for consumer and telecommunication chips are now confronted with the same rule of diminishing returns. To further improve their performance, the processors are getting disproportionately bigger and consume much more energy per operation than previous generations.

Traditionally, embedded systems-on-chip (SoC) have been designed as heterogeneous multiprocessors, where most processors are not programmable and a single control processor synchronizes all communication. Obvious advantages of such systems include low cost and low power consumption. In high volume products this outweighs disadvantages like a low degree of design reuse, little software reuse, and long product lead times. Despite all the hard work and good intentions it has proved difficult to establish a *platform* around heterogeneous SoC architectures.

With the rise of non-recurrent engineering costs and an increasingly global and competitive semiconductor market, the need for a successful SoC platform is felt stronger than ever in the industry. Next to cost, the availability of qualified engineers is often even a bigger problem. Given that it is not unusual to spend several hundreds of man years on software development for a single product, it is easy to see that even a multinational company can only have a very limited number of products in development at any point in time.

The solution we propose is to move away from heterogeneous SoC and instead embrace homogeneous embedded multiprocessors. In this talk we discuss embedded multiprocessor architectures and how they relate to programming models. We contrast heterogeneous to homogeneous architectures, and we show how the traditional efficiency gap between the two is narrowing. We also discuss issues related to hardware and software reuse, and the quest for composable systems to speed up the often lengthy process of embedded system integration.