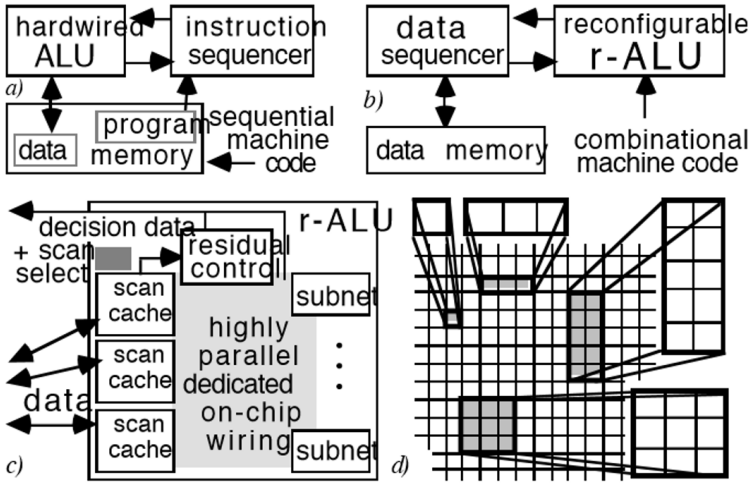
The Machine Paradigm of Xputers and its Application to Digital Signal Processing Acceleration

R. W. Hartenstein, A. G. Hirschbiel, M. Weber,
Universitaet Kaiserslautern, Postfach 3049; D - 6750 Kaiserslautern, F. R. G

Abstract. The paper gives an introduction to using xputers (a novel class of high performance processors - based on one of the most important machine concepts since von Neumann) for acceleration of digital signal processing. Its novel programming paradigm of data sequencing is illustrated by a FFT digital signal processing example.

Introduction. Extremely high throughput (up to several kiloMIPS) is needed at very low hardware cost for quite a number of real-time applications, such as computer vision, computer graphics, signal processing and others, when embedded in mass products. Such requirements cannot be met even by most advanced von-Neumann-type processors, nor by parallel or concurrent computer systems (because of excessively massive overhead and because most of the parallelism cannot be mapped onto such a hardware [HHW90]), nor by Application-specific Array Processors (ASAPs) exhibiting I/O overhead (scrambling and unscrambling of data streams), very high design cost. The main bottleneck in exploiting parallelism is the communication between processors or PEs, rather than the number of PEs. In ASAPs communication indigestion between PEs is avoided by locality: PEs are placed such, that only nearest neighbor communication is needed by dedicated on-chip wires being much faster than buses and avoiding any overhead [HaKo75].

But locality restricts ASAP application to systolizable algorithms only, so that many important DSP algorithms are not supported. **X**puters do not have such a restrictions, reaching the performance of ASAPs, also for non-systolizable parallel algorithms, where xputers are highly superior to (von Neumann) **com**puters. The superiority of xputers has several reasons which are summarized by fig. 5 and explained in detail elsewhere [Bil90, HHW90]. This paper first briefly summarizes the novel parallel machine paradigm and then shows by a FFT example the high flexibility and the easy use of this paradigm for implementing very high performance digital signal processing applications. Finally some experimental performance figures and hardware expense figures will be shown and discussed.



e)	(von Neumann) computers	xputers
the role of: data flow	derived (by pointers) from control flow	**primary activator** scheduled at compile time
the role of: control flow	**primary activator**	derived (only when needed): **(sparse) residual control**

Fig. 1: Basic structures of: a) von Neumann machine and b) xputer; c) r-ALU of the MoM xputer architecture, d) MoM cache adjust examples, e) primary activator.

Xputer Principles. Fig. 1 compares the basic structure of computers (fig. a) with the structure of **x**puters (b). The computer ALU is a very narrow bandwidth device: it can carry out only a single simple operation at a time. Xputers, however, use a PLD-based *r-ALU*, reconfigurable into one or more highly parallel very powerful combinational *compound operators* (subnets in fig. 1 c). PLDs are available from about 23 vendors of a booming billion US-dollar world market

Xputer operation [HHW90] (called data sequencing) is data-driven, which is fully deterministic (fig. 1 e) - unlike that of data flow machines (which operate indeterministically). Xputers avoid most of the well-kown von Neumann bottlenecks except one: word by word access to (data only) primary memory. This only bottleneck having been left over, however, has been greatly compensated [HHW90] because xputer hardware is much more compiler-friendly (than that of computers): very much more powerful optimization strategies can be mapped onto this hardware, optimized data maps yield very high hit rates with interleaved memory access, and/or significantly reduced number of memory access cycles with VLDW (very long data word) memory use (see later in this paper).

Xputers have a special philosophy of using multiple register files called *data scan caches* or *window scan caches*. A scan cache is a hardwired window to the memory space, the format of which is electrically adjustable at run time. Fig. 1 d shows a few scan cache format examples from the MoM (Map-oriented Machine [HHW87]) having a 2-dimensional memory organization. A scan cache is very tightly connected to the r-ALU. This fully parallel bidirectional interconnect uses hundreds or thousands of fully dedicated direct (mostly on-chip) wires (no bus! - see above) between the r-ALU and each bit of each word held by the cache.

The data sequencer is a hardwired address generator featuring scan patterns making the location of a scan cache travel through data memory space (fig. 1). Examples of such scans are [HHW87, HHW89]: single step, data jump, linear scan, video scan shuffle, butterfly, trellis etc. For data dependent scans, such as e.g. in curve following [HHW87] or Lee routing [HHW90] and other applications, decision data generated by the rALU may influence data sequencer operation (see fig. 1 c).

Constant-geometry FFT. The constant-geometry version of the FFT algorithm is a suitable and sufficiently simple example to demonstrate the efficiency of the xputer paradigm. Fig. 2 a shows the data flow graph of a 16-point FFT example looking like an iterative overlay of spiders (an operator node is the body of a spider). Fig. b shows a MoM xputer data map and the ease of deriving it from the flow graph. Fig 2 c shows, how the cache configuration is derived by just picking contiguous locations of the feet of a particular spider. Caches communicate via a r-ALU subnet carrying out the compound operations "a + w.b" and "a - w.b" (fig. 2 d). Fig. 2 b indicates the initial locations of the 3 caches from which the following two-level nested scan pattern is starting.

At each cycle the r-ALU subnet reads from cache1 and writes into cache2 and cache3. For linkage to next scan after this one a tagged control word TCW (fig. 2 b) will be decoded by an extra r-ALU subnet (fig. 2 g). Since being derived from data sequencing we call this residual control (compare fig. 1 e), and, since only 1 of 144 memory accesses is used for control, we also call this sparse control.

VLDW (Very Long Data Word) Version. Using a VLW memory (Variable Length Word Memory) more word length and thus more performance can be traded off for address space [VL90]. This is a way to speed up a given xputer implementation of a parallel algorithm. To illustrate this with the above FFT example instead of one spider we take 4 adjacent spiders (fig. 2 f) to derive a r-ALU configuration being 4 times more powerful (fig. 3 b). A VLDW containing 6 variables has been introduced for memory and caches to reduce the number of memory access cycles (fig. 3 b). Fig. 3 a indicates initial and final cache locations of a new scan pattern. In terms of memory semi cycles a speed-up by a factor of 5 has been obtained, which demonstrates the flexible performance of xputers.

Acceleration factors achieved by xputers. Our main goal is it, not to measure the performance of the latest technology processor design, but to prove and to measure the superiority of the fundamental principles of xputers over von Neumann principles in a technology-independent way. That's why - instead of the highly technology-driven measure of MIPS - we prefer to use the notion of *acceleration factor* to measure the relative benefit of xputer use over the use of a computer of comparative technology.

For performance evaluation we have implemented a number of other algorithms. This experimental approach compares the processing time needed by a VAX 11/750 to that needed by a first generation MoM [HHW87] using technology of the early and mid' 80s (TTL- based breadboards and 3 micron nMOS ICs). Fig. 4 shows some of these acceleration factors. Factors of 100 or substantially more seem to be typical for DSP applications. For other applications acceleration factors up to > 2000 have been measured. Fig. 5 summarizes some of the reasons of this superiority of xputers having been published elsewhere [Bil90, HHW90].

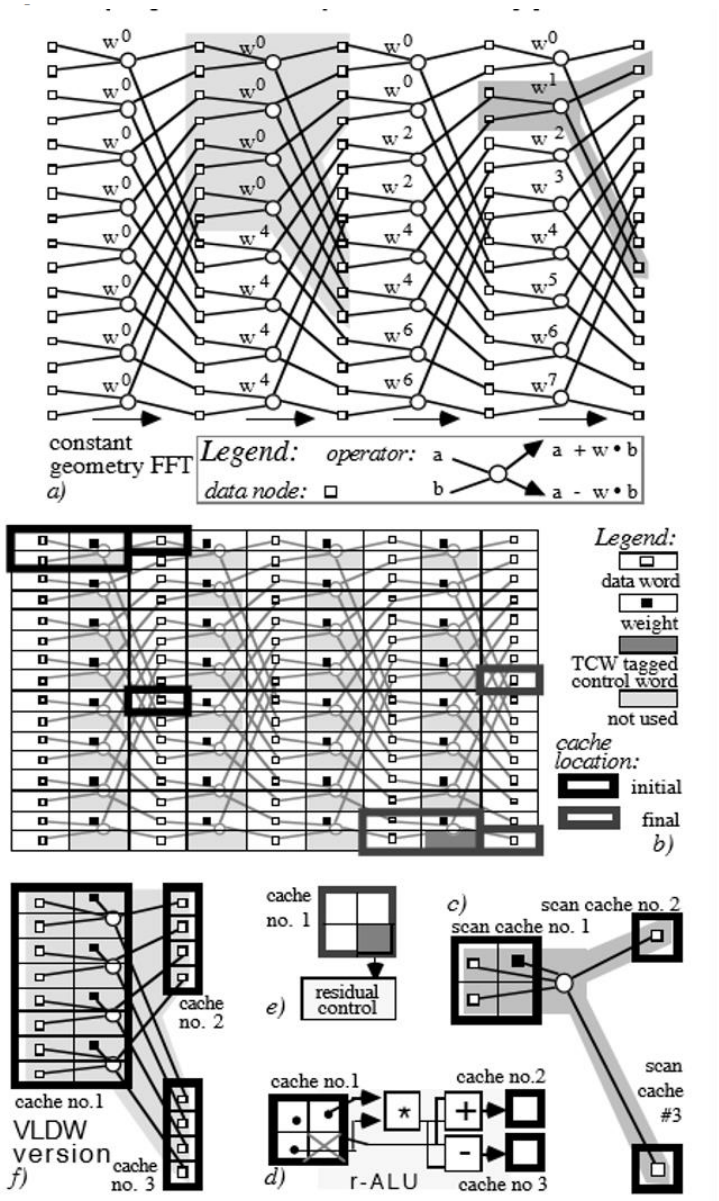


Fig. 2. A 16-point constant-geometry FFT on the MoM: a) signal flow graph, b) data map c) deriving cache config. from b, d) r-ALU subnet , e) deriving a VLDW version

Technology Issues. From Plessey also "normal" gate arrays are available being compatible to available PLDs. That's why tested and debugged xputer machine code can be used directly - without extra design effort - to freeze an implementation into an ASIC - since because of their universality [HHW89-90] xputers may also run in stand-alone mode. Fig. 4 also shows the number of PLDs or fraction of a PLD needed for the r-ALU. Since the (non-PLD) data sequencer is very simple (<10,000 transistors)) often also the standard xputer parts would often conveniently fit onto the r-ALU chip, so that single chip DSP solutions are possible which could replace very expensive conventional kiloMIPS hardware

```
scandef LINEARDOWNSCAN                parbegin (* synchronously*)
cache1 [0..1, 0..1] from (0,0) step (0,2) to (0,14);
cache2 [0, 0] from (2,0) step (0,1) to (2,7),
cache3 [0, 0] from (2,8) step (0,1) to (2,15);   parend      endscandef;

scandef CGFFT_SCAN
scan LINEARDOWNSCAN from (0,0) step (2,0) to (2,0) endscandef;
scan  CGFFT_SCAN   from FFT_DATAMAP (0,0);     (* execute *)
```

Figure 2 g.

Conclusions. The paper has paper given an introduction to xputer use to accelerate also DSP applications. Dramatically high acceleration factors (up to >2000) have been achieved experimentally. Factors around 100 - 300 seem to be typical in DSP. Efficiency and comprehensibility of the programming paradigm, and flexibility of the deterministically data-driven xputer machine paradigm has been illustrated by a FFT example. It has been shown that the compile-time data scheduling principles avoid overhead by achieving, that the right data are at the right time at the right place. It has been shown that xputers are very cheap and useful accelerators. Xputers combine the flexibility, generality and low cost of microcomputers with the speed advantages of specialized hardware solutions.

Acknowledgements. Early MoM concepts stem from the E.I.S. project, funded by BMFT and Siemens-AG, Munich, coordinated by GMD,

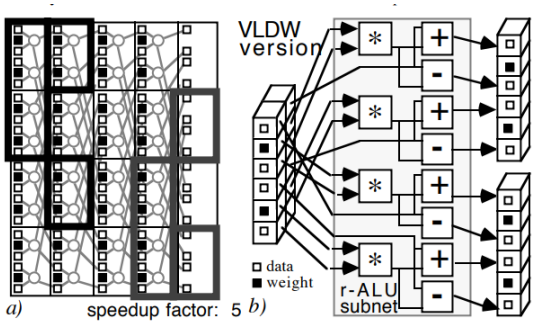


Fig. 3. VLDW version of fig. 3: a) data map, b) r-ALU/cache configuration

Schloß Birlinghoven. We appreciate valuable ideas from Klaus Singer at eltec GmbH at Mainz. We also would like to acknowledge the various kinds of support from GMD, especially by Elfriede Abel, Herwig Heckl, Gustl Kaesser, Theo Vierhaus and Klaus Woelcken.

Literature

[Bil90] R. Hartenstein, A. Hirschbiel, M. Weber: *Using Xputers as Inexpensive Universal Accelerators for Digital Signal Processing;* Proc. Bilkent International Conference on New Trends in Communication, Control and Signal Processing, Ankara, Turkey, July 1990; North Holland, Amsterdam 1990

[HHW87] R. Hartenstein, A. Hirschbiel, M. Weber: MOM - Map Oriented Ma- chine, in: (Ambler et al.:) Hardware Accelerators (Proc. Int'l Workshop on Hardware Accelerators, Oxford UK, 1987), Adam Hilger, Bristol 1988

[HHW89] R. Hartenstein, A. Hirschbiel, M. Weber: MOM - a partly custom-designed architecture compared to standard hardware; Proc. IEEE Comp Euro '89, Hamburg, F.R.G., IEEE Press Washington DC, 1989

[HHW90] R. Hartenstein, A. Hirschbiel, M. Weber: Xputers: an Open Family of non-von Neumann Architectures; Proc. GI/ITG Conference on the Architecture of Computing Systems, Munich F.R.G., March 1990; VDE-Verlag, Berlin, 1990;

[VL90] R. Hartenstein et al.: An Experimental Variable Length Data Word Memory for the MoM; report, Universitaet Kaiserslautern, 1990 [1979]

algorithm example	acceleration factor	number of PLD chips for r-ALU	
		LCA XC 3200 [1]	ERA 60400 [2]
design rule check	> 2000	8	4
electr. rules check	> 300	1	< 1
Lee routing	> 160	< 1	< 1
shuffle sort	80	<< 1	<< 1
2-dim filtering	300	< 1	<< 1
FFT	> 300	3	2
vect/Matrix mult.	150	1	< 1

[1]) available from Xilinx [2]) announced by Plessey

Fig. 4. Xputer acceleration factors and hardware expense figures

	(von Neumann) computers	xputers
instructions	hardwired + simple multiplexed: only 1 at a time	tailored: config'ed at compile time very powerful compound functors
machine code	sequential	combinational
sequencer	instruction sequencer	data sequencer
parallelism at	process or (VLIW:) instruction level	gate level and/or data path level
compilation techniques	traditional	partly derived from VLSI CAD
data format	dependant of instruction format	independent - highly flexible adaptable VLDW is feasible
interleaved data memory	low hit rate	very high hit rate
'cache' use	probabilistic - low hit rate	deterministic - very high hit rate
addressing overhead	high to very high	extremely low
other overhead	very high	very low

Fig. 5. Performance-relevant differences between computers and xputers