9. Conclusions

Reconfigurable computing [HBK96] [MHA97] has been a promising research area for the recent years. Such systems have shown the potential to achieve high performance for a variety of applications. Up to now the research on reconfigurable computing has focused mainly on the accelerator architectures themselves. However, experiences from traditional computers have demonstrated, that the overall system performance depends also strongly on the peripheral parts. Especially the memory interface has turned out to be a bottleneck, which can extremely reduce the performance of the system. Further it has shown that the memory bottleneck is even more challenging for reconfigurable computers than for traditional computers. Since reconfigurable architectures are programmed before application execution the memory interface is only burdened by data accesses. Conventional cache implementations do not remedy, since computation data commonly does not show the same locality as instructions. Additionally reconfigurable architectures have a very high data throughput because of the improved computation speed.

The reconfigurable computing scene commonly tries to avoid this problem by using SRAM. SRAM is faster than DRAM and requires less controller hardware. However, SRAMs are bigger and more expensive than DRAMs. Therefore it not feasible to implement realistic applications and/or construct a successful commercial reconfigurable computer with SRAM. But as DRAMs are slower than SRAMs, additional concepts have to be developed to speed up data accesses.

In this thesis a solution to alleviate the memory communication problem for reconfigurable computing systems has been developed. It covers the memory communication speed, the reconfiguration load, and the overall design time.

Different methods have been developed to increase the memory bandwidth. Therefore the overall solution targets the memory communication problem on four levels:

- ❑ selection of well suited memory devices,
- ❑ the memory interface architecture,
- ❑ the data sequencing method, and
- ❑ compiler techniques.

The most direct way to enhance memory access is the utilization of faster memory devices. This approach aims to reduce the physical data access time. In this field, a large variety of devices, especially in the DRAM sector, are available. Often, those devices also offer enhanced data access for consecutive data words (burst mode), with fixed or variable length. Today’s most important memory types have been summarized in section 3.1 at page 25. The solution presented in this thesis is based on Multibank DRAM, which provides variable length burst access capabilities.

Apart from the utilization of fast memory devices, special memory architectures have evolved to increase the memory bandwidth of conventional microprocessors. The best known examples are caches [PH90] and interleaved memories with skewing schemes [BK71]. Since reconfigurable computers show a different memory access behavior, a memory architecture dedicated for reconfigurable computing has been developed. This architecture can be characterized by the following features:

- a two dimensional memory organization,
- parallel memory banks,
- a special row major mapping scheme to map the two dimensional memory to parallel memory banks, and
- a second memory hierarchy called smart interface, which implements the scan window overlap optimization.

The data sequencing method has also a big influence on the memory communication problem. Besides the memory interface speed, it also influences the reconfiguration load and the design time. It has turned out that the programming methods of conventional computers are not applicable on reconfigurable systems. While the sequencing of conventional computers is software based, reconfigurable computers demand a different concept. In this thesis a new data sequencing method has been developed which is based on the following concepts:

- The well defined Xputer paradigm (see e.g. [HHW89], [HHS92], or [AHR94]) has been used as underlying hardware architecture.
- A general generic model for data sequencing applications.
- A two level universal data sequencing method has been used. It involves handle position generation and scan window generation.
- Hierarchic handle position generation. The generic handle position generation allows to access large data sets in complex pattern with a few configuration bytes. This method efficiently keeps the reconfiguration load low. Further generic generation of handle positions can be performed quite fast. This led to the following idea:
- Stack based control of address generation. Here only one stepper is utilized to perform the handle position generation. Complex scan pattern are generated by changing the parameters. These parameters are stored in a stack architecture. This method saves many hardware resources, is very flexible, and is not controlled by instructions.
- Look-up table based scan window generation. The look-up table based scan window generation performs the low level accesses to the data needed in a computation step in an arbitrary sequence. This method enables hardware and software level access optimizations.

The involved application development methodology uses the following techniques to adapt applications for the utilization of the hardware architecture:

- loop-transformations,
- scheduling of data accesses, and
- modification of storage schemes.

An application example (see chapte r8) has demonstrated, that, based on the presented concepts, the number of memory cycles of a given application may be reduced by a factor better than an order of magnitude (see figure 8-13 at page 203). This optimization factor is technology independent, i.e. the number of memory cycles is reduced independently from speed of the utilized hardware platform.

The developed data sequencing method has been the basis for the implementation of a custom computing machine. The Map-oriented Machine with Parallel Data Access (MoM-PDA) has the following features:

- two parallel banks of MDRAM,
- a generic data sequencer, which implements all presented hardware and software level optimizations,
- multitasking capabilities,
- a PCI interface to be connected to a host computer, and
- computations are performed by the coarse-grained KressArray.

While the data sequencer of the MoM-PDA is a hardwired implementation, a second KressArray-based implementation has been developed. In contrast to the MoM-PDA, which uses multiple different devices, the KressArray implementation is the basis of a reconfigurable system on a chip: here the KressArray and memory banks are integrated into one device. This device may be configured simultaneously with an address generator and the computational datapath. To save valuable design space, a method to generate application specific data sequencers has been developed. These application specific data sequencers are still programmable to be adjusted to different data sets for applications of the same type. Multiple KressArray implementations (see section 6.5.2 at page 125and section 8.6 at page 207) of the data sequencer have demonstrated, that the data sequencer requires only a few (local) routing resources and is efficiently mappable to KressArrays with at least two nearest neighbor connections.

As a development framework for data sequencer applications the Xputer multimedia development framework (XMDS) has been developed (see appendix A). The XMDS has the following features:

- it is an Internet-based design software,
- supports the designer by multimedia features,
- provides an extensive toolset,
- enables remote prototyping,
- provides an IP-library, and
- integrates a generator for application specific data sequencers.

Future work may integrate an automated trade-off analysis between handle position generation time and scan window generation time as manually performed by the application analyzer (see appendix A.2 at page 229) for the application specific data sequencer generation (see appendix A.2 at page 229). This would allow to generate data sequencers, which are optimal in execution time and area requirement.