Reconfigurable Processor Employing Optical Channels

Majd F. Sakr[a,c], Steven P. Levitan[a], C. Lee Giles[c], Donald M. Chiarulli[b]
[a]EE Department, University of Pittsburgh
Pittsburgh, PA, 15260 USA
[b]CS Department, University of Pittsburgh
Pittsburgh, PA, 15260 USA
[c]NEC Research Institute
Princeton, NJ, 08540 USA

ABSTRACT

We describe a reconfigurable computing architecture that exploits parallel optical channels to support fast reconfiguration and compare our architecture to configuration cache based designs.

Keywords: Reconfigurable Processor, Optical Channels, Run Time Reconfiguration, Configuration Cache, Photo Detector

Reconfigurable Processor Employing Optical Channels

Majd F. Sakr[a,c], Steven P. Levitan[a], C. Lee Giles[c], Donald M. Chiarulli[b]
[a]EE Department, University of Pittsburgh
Pittsburgh, PA, 15260 USA
[b]CS Department, University of Pittsburgh
Pittsburgh, PA, 15260 USA
[c]NEC Research Institute
Princeton, NJ, 08540 USA

1. INTRODUCTION

Reconfigurable computing architectures are gaining popularity as replacements for general purpose architectures in many high performance applications. Reconfigurable systems can take advantage of deep computational pipelines, perform concurrent execution and are inherently data flow in nature. Many applications can exploit these systems, such as genomic sequence scanning, Fast Fourier Transform, text searching, and computer vision. Current research efforts are applying reconfigurable computing to perform automatic target recognition, real-time image processing, and hardware implementation of neural networks. However, these architectures suffer from a trade off between slow reconfiguration times versus low logic gate density when used to support large computations[2]. This problem is due to the fact that configuration memory typically resides off-chip and reconfiguration is performed serially. Recent approaches[4] solve this problem by adding an on-chip configuration cache that provides faster reconfiguration at the cost of die area. That is, the area overhead of the configuration cache gives a low total logic gate density for the architecture. These disadvantages limit the performance, and therefore the applicability of current reconfigurable systems.

In this paper, a reconfigurable processor architecture is proposed that overcomes the limitations discussed above by using high bandwidth optical channels. The optical channels allow fast parallel loading of the reconfiguration control word as well as the migration of the configuration cache off-chip. The migration of configuration cache allows better utilization of the die area for reconfigurable processing elements. Further, it is possible to implement the optical detectors directly in silicon, which does not require significant alteration of the fabrication processes. These advantages make the optically reconfigurable architecture competitive for high performance applications.

2. PROPOSED ARCHITECTURE

Field Programmable Gate Array (FPGA) reconfigurable systems are implemented using arrays of reconfigurable processing elements such as the one shown in Figure1. Each element can realize the functionality of a logical operation under the control of a *configuration buffer*, which is capable of storing a single configuration for the processing element. The complexity of the reconfigurable element varies in different FPGA technologies. We use the term *gate equivalent* to represent such a generic processing element. Configuration buffers are also used to configure the interconnect between processing elements such that that groups of processing elements can realize more complex functionality which are called *microoperations*.

Reconfiguration of the processing elements is performed by reading the data for all the configuration buffers stored off-chip in a Read Only Memory (ROM) module. The reconfiguration is slow due to two factors. First, because the configuration data resides off-chip, reconfiguration of the processing elements requires long time delays to access the data from the ROM. Second, reconfiguration in these systems is performed serially. That is, after the control word is fetched from the ROM it is shifted serially to the configuration buffers of the reconfigurable processing elements. Due to long reconfiguration times, these systems are typically used as configure-once application-specific compute systems. Therefore, they do not fully exploit the flexibility offered by reconfigurable hardware.

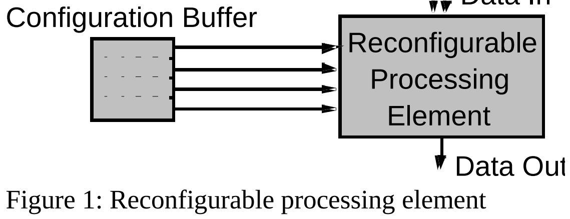

Figure 1: Reconfigurable processing element

To fully take advantage of the reconfigurable hardware and perform different computations in real time, we must perform "on the fly" reconfiguration or run time reconfiguration. Current architectures overcome the slow reconfiguration time by adding on-chip configuration cache to the reconfigurable processing module. The on-chip cache can alleviate the problem of slow off-chip memory access to fetch the needed configuration data. Hence, the reconfiguration time is reduced at the cost of die area. Current designs that implement this approach estimate the overhead of on-chip configuration cache to approach 50% of the total die area[4]. This overhead is very high and hence limits the capability of such systems to represent complex functionality in hardware.

In order to provide maximum utilization of the die area for computation, we propose a reconfigurable processing system that has the configuration cache off-chip and high-speed parallel optical channels for loading configurations. An array of optical detectors is added to the die of the reconfigurable processing unit. Each detector can receive configuration data for a large group of processing elements since each optical channel offers a high bandwidth connection to memory. The configuration data is transmitted optically in a 2D fashion to each of the on-chip detectors achieving very fast parallel reconfiguration of the processing elements. This architecture is illustrated in Figure 2. This architecture overcomes the limitations of current reconfigurable systems, it enjoys fast run time reconfiguration as well as full utilization of the processing resources.

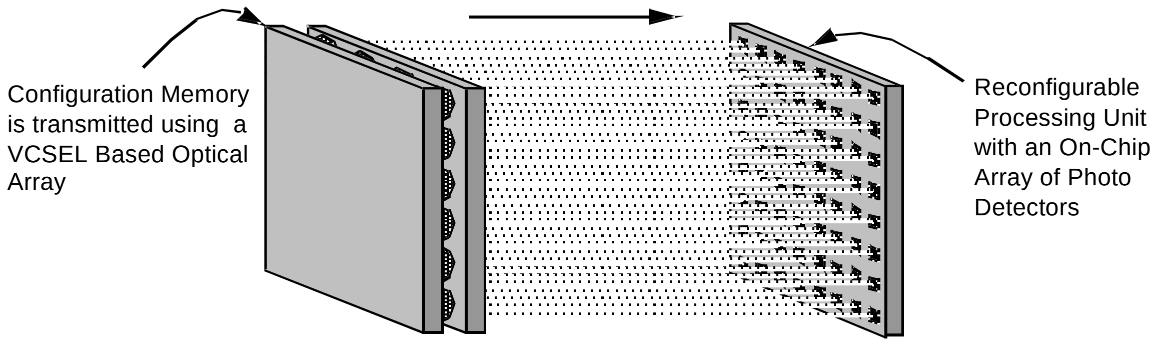
Configuration Memory is transmitted using a VCSEL Based Optical Array
Reconfigurable Processing Unit with an On-Chip Array of Photo Detectors

Figure 2: Reconfigurable Processor with off-chip configuration memory read using a 2D array of optical detectors

3. PERFORMANCE ANALYSIS

In this section we develop a performance measure to compare architectures that employ on-chip configuration cache and a single channel to off-chip configuration memory versus architectures that do not utilize any on-chip cache but use multiple optical channels to load off-chip configurations. We focus our model to take into effect the performance parameters that will affect the total execution time and specifically the configuration time for the architectures mentioned. In other words, we focus on studying the effectiveness of both architectures at lowering the run time reconfiguration time. A more detailed analysis, in which a general reconfigurable processor performance model was developed, has been presented in previous work[3].

1. Performance Modeling

We define the performance of a processor as the time a processor needs to complete the execution of an application. In general, the execution time can be defined as the total time needed per configuration of the hardware times the number of configurations required to execute the application. The time needed per hardware configuration consists of three parameters. First, is the time to load the configuration data that defines the functionality represented in the logic, T_C. Second, is the time to read/write the data needed to perform the computation, T_D. Third, is the time needed to execute the operations defined on the available data T_E. In this summary we focus only on the effects of T_C and T_E on total execution time. Therefore, for this study we define total execution time as: $T = (T_C + T_E) \times C$.

The number of configurations needed, C, is the total number of microoperations, M, that define the application divided by the average number of microoperations that can be represented in a single hardware configuration, M_{config}. This is a function of both the processing area available on the chip and the structure of the application being executed, as explained below.

The total processing area or die capacity of the processor can be represented as the total number of *gate equivalents* available on-chip, G. However, this die capacity is not fully used as processing area, since some of the die area is used as configuration cache for some architectures and as photo detectors for other architectures. Hence, the total available processing area $G_{proc} = G - G_{cache}$, where G_{cache} is the die area needed for configuration cache measured in gate equivalent area. For architectures employing optical channels $G_{proc} = G - G_{optic}$, where G_{optic} is the area required to implement the photo detectors measured in gate equivalents.

We define M_{proc} as the maximum number of microoperations that can be realized simultaneously in a single hardware configuration $M_{proc} = G_{proc} / G_m$, where G_m is the number of gate equivalents needed to realize a single microoperation. However, in general, the actual number of microoperations that can be realized simultaneously in a single hardware configuration is a function of the parallelism in the application, the partitioning efficiency, and the average amount of hardware reuse that takes place from one configuration to the next. Therefore, $M_{config} = r \times M_{proc}$, where r captures these effects. Hence, $C = M / M_{config}$. The number of gate equivalents required to represent M_{config} microoperations in a single hardware configuration is $G_{config} = M_{config} \times G_m$. We define B_g as the number of configuration bits needed to configure a single gate equivalent. Hence, to configure G_{config} a total of $B_g \times G_{config}$ configuration bits are needed.

T_C is the time it takes to load the configuration data that is required to realize the M_{config} microoperations used in a single hardware configuration. For the architecture that employs on-chip configuration cache, when reconfiguration is required, the needed configuration bits could either reside in the on-chip cache or off-chip in configuration memory. The probability that a needed configuration can be found in the cache, or the cache hit-rate, is denoted as P_{hit} and the miss rate as P_{miss}. Hence, the time required to reconfigure the processor $T_C = P_{hit} \times T_{onchip} + P_{miss} \times T_{offchip}$, where T_{onchip} is the time to access a single configuration from on-chip configuration cache and $T_{offchip}$ is the time to access a single configuration from off-chip configuration memory. We assume that the access to on-chip cache to retrieve the bits that constitute a single configuration is performed in parallel. Hence, T_{onchip} is defined as the time to access a single configuration bit from on-chip cache. Furthermore, since this architecture uses a single port to memory, $T_{offchip} = (B_g \times G_{config}) / S_{elec}$, where S_{elec} is the bandwidth of the electronic serial channel to memory.

For the second architecture that employs optical channels, all configuration bits must be loaded from off-chip memory, $T_C = T_{offchip}$. In this architecture, N optical channels connect the reconfigurable processor to configuration memory. Therefore, $T_{offchip} = (B_g \times G_{config}) / (S_{optic} \times N)$. With N channels accessing configuration memory in parallel, $T_{offchip}$ is the time of the channel that needs to retrieve the most bits from off-chip memory. In this model, we assume the worst case number of bits that are needed per optical channel.

2. Performance Comparison

In order to compare these two reconfigurable processors we use the execution time function (T) defined above. Since we assume that processors are fabricated using the same technology, we can assume that T_E is the same across processors and hence can be treated as a constant. We compare the performance of two processor architectures executing a single application consisting of 10,000,000 microoperations (M = 10,000,000). Both processors have the same die area, which is G = 100,000 logic gate equivalents. We compare the effect of adding more communication channels to configuration memory on the total execution time to the effect of adding more configuration cache on the total execution time.

The first architecture employs on-chip configuration cache, G_{cache}, with an on-chip access time (T_{onchip}) of 2ns and a single electrical channel to off-chip configuration memory with a bandwidth (S_{elec}) of 50MHz. The cache hit rate (P_{hit}) for the first architecture is modeled versus cache size using the well known *parachor curve*[1] as follows:

$$P_{hit} = \frac{1}{1 + \frac{G(1-L)}{G_{cache}}} \qquad 0 < L < 1$$

where L, represents the locality of the accesses, an L value close to one indicates high locality. For this study we use L = 0.98 and 0.99.

The second architecture is an optically reconfigurable processor that employs *N* optical channels to configuration memory but does not use any on-chip configuration cache. The overhead of each optical channel is 100 gate equivalents, therefore $G_{optic} = 100 \times N$. The bandwidth (S_{optic}) of each optical channel is 200MHz. We vary *N*, the number of optical channels used, and record the effect on total execution time. Other parameters shared by both architectures are $B_g = 5$ configuration bits per gate equivalent; $G_m = 100$ gate equivalents to represent a single operation, $r = 0.6$ the percentage of operations that are represented in a single configuration, and $T_E = 0.1$ms.

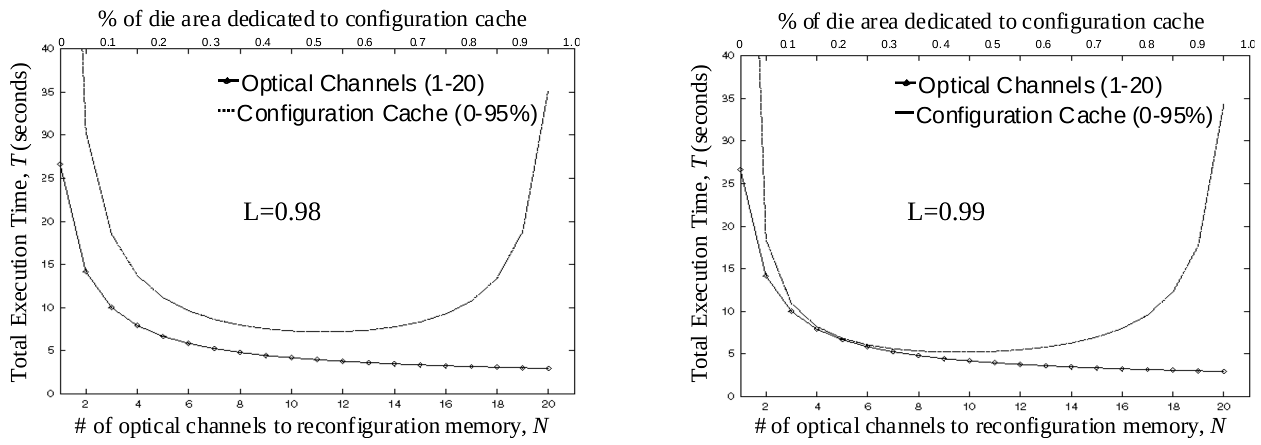

Figure 3: Total execution time vs. number of optical channels & percentage of die area dedicated to configuration cache

In Figure 3, we show the results of the performance comparison for two values of locality. For the cache based architecture, we show the effect of increasing the die area used as configuration cache, G_{cache}, on the execution time, *T*. The configuration time decreases as the percent of the die area that is used as configuration cache increases. However, the number of configurations grows as the die area used for configuration cache increases. As illustrated by the total execution time shown in the graph, the number of configurations required for cache sizes greater than 60% dominates the advantage of the increased cache.

Also, in Figure 3, for the optically reconfigurable architecture, we plot the total execution time, *T*, versus the number of channels, *N*, to configuration memory. The number of optical channels is varied from 1 to 20. The more channels used to access configuration memory, the shorter the execution time.

The analysis demonstrates that even with a few optical channels a significant improvement in performance can be achieved in comparison to systems using part of the die area for configuration cache. Further, when the number of channels is increased to 16-20, still a relatively small number, a substantial advantage over corresponding cache based systems can be achieved even for applications that exhibit very high degrees of locality.

4. ACKNOWLEDGEMENTS

S. P. Levitan and D. M. Chiarulli would like to acknowledge support from AFOSR Grant F-49620-93-1-0023 for work done at the University of Pittsburgh.

5. REFERENCES

1. H.M. Deitel, *Operating Systems*, Addison Wesley, 1990.
2. J. Rose and D. Hill, "Architectural and Physical Design Challenges for One-Million Gate FPGAs and Beyond," *in FPGA `97, ACM Symposium on FPGAs*, Feb 1997, pp. 129-132.
3. M.F. Sakr, S.P. Levitan, C.L. Giles, D.M. Chiarulli, Reconfigurable Processor Architectures Exploiting High Bandwidth Optical I/O," *Computer Science Technical Report CS-TR-98-2*, University of Pittsburgh, 1997.
4. H. Schmit, "Incremental Reconfiguration for Pipelined Applications," *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines*, 1997.