Heriot-Watt University
Edinburgh

Optically Interconnected Dynamically Reconfigurable Computing

Abstract
Dynamically reconfigurable computing could potentially revolutionise the computer industry by providing custom hardware to the end user on demand. This report examines dynamically reconfigurable computing by first considering the evolution of its core component, the FPGA, and classifying various configurations. Limitations in FPGA bandwidths are highlighted and the potential of optics introduced as a solution. Three system adaptations are examined and conclusions drawn.

Name:	Keith J. Symington
Registration number:	9711710098481
Company/Address:	Physics Dept., Heriot-Watt University, Edinburgh, EH14 4AS
☎:	+44 (131) 451 3040
Fax.:	+44 (131) 451 3136
e-mail:	kjsymington@iee.org
Supervisor:	Dr. J. F. Snowdon
Date:	25th June 1999

1 Contents

2 Introduction

This literature search will examine the evolution of reconfigurable hardware to its current embodiment: specifically the field programmable gate array (FPGA). The FPGA will be examined in detail and along with its application in the field of reconfigurable computing. The addition of optical I/O opens up application areas in high bandwidth reconfigurable computing previously excluded by the low bandwidth I/O of high density FPGAs. Three case studies will then be examined and their application using dynamically reconfigurable optically interconnected FPGAs considered.

In an attempt to provide a consistent overview of configurable hardware the following terminology will be adopted [1]:

- **Configurable Hardware:** Hardware whose function can be configured one or two times. An example of such hardware is a standard circuit board produced by a manufacturer for a specific product line. Although the board supports any product versions from the low to high ends, its exact function is configured by adding the appropriate components and modifying sets of switches or jumpers.
- **Reconfigurable Hardware:** Reconfigurable hardware has the advantage that it can be reconfigured many times to suit the task on hand. Such systems usually contain components such as programmable logic devices (PLDs), which act as truth tables, Boolean equations or state machines, or programmable read-only memories (PROMs) which can be used as non-volatile memories for microprocessors or microcontrollers. Some forms of reconfigurable hardware are "in-system programmable" (ISP) and can be re-programmed while still resident on the circuit board.
- **Dynamically Reconfigurable Hardware:** The advent of static RAM (SRAM) based FPGAs opened up another chapter in reconfigurable hardware. The SRAM cells allowed the FPGA's logic to be configured by simply loading a bit pattern into the SRAM: thus the system could be quickly reprogrammed to perform any appropriate task. For example, on

start-up the FPGA could be configured to perform diagnostics on itself and its circuit board before dynamically reconfiguring to perform the main task(s) that it was designed for.

- **Virtual Hardware:** The problem with the majority of dynamically reconfigurable hardware is that in order to reconfigure the device, operation needs to be halted and the entire SRAM reloaded - also resulting in the irretrievable loss of any data in FPGA registers. This lead to the development of a new type of FPGA which supported dynamic reconfiguration of selected portions of internal logic with no disruption to the device's I/O or system level clocking and continued operation of portions of the device not undergoing reconfiguration. Of particular interest however, is that the contents of internal registers are not lost during reconfiguration. This allows one instantiation of a function to hand over data to a new instantiation of another function.

3 Device Evolution

There are many methods at a designers disposal for integrating discrete components onto a single VLSI application specific integrated circuit (ASIC) [2], each method having a specific capacity, performance, flexibility and unit cost (in both time and money). This section outlines the functions of various technologies with the aim of highlighting the new opportunities presented to designers through FPGAs.

3.1 Full Custom

A full custom system is where the entire VLSI circuit is carefully tailored to the designers requirements. This results in a high performance chip with optimal silicon usage and is sometimes the only method of implementation considered for certain applications; such as state-of-the-art microprocessors. Fabrication costs and initial set-up times for full custom solutions are also high since the flexibility afforded to the designer results in chip design being complex and time consuming.

3.2 Standard Cells

A standard cell design sacrifices the flexibility and performance of a full custom design to speed up the design process. This is done by using a specific set of design restrictions and standard cells enabling the use of software tools to automate the design process.

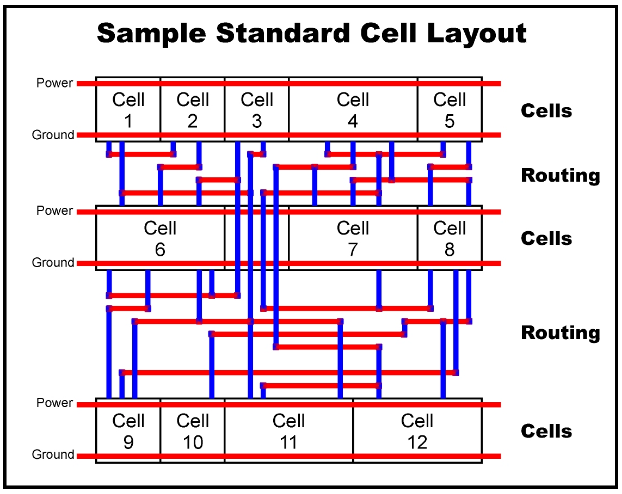

Figure 1

Sample routing of a series of standard cells.

A standard cell method predefines all gates as a series of cells with the same height and placement of power and ground lines (see figure 1). The chip is laid out in interleaved logic and routing rows, the latter being as large as is necessary to accommodate all routing. Routing between routing rows is accomplished using higher metal layers or by route through cells. Essentially, the designer need only specify functionality and software will create the layout. Fabrication of a standard cell is done by a silicon foundry, resulting in approximately the same fabrication times as full custom designs.

3.3 Mask-Programmable Gate Arrays (MPGA)

The problem with both full custom and standard cell designs is that they have to be fabricated from scratch. In an MPGA most of the fabrication is carried out beforehand. For example, an MPGA may consist of sets of transistors in specific locations which are partially interconnected to form the basis for mapping certain types of logic gate. Thus all any designer need do is interconnect the required elements to give the desired functionality.

This method reduces both cost and time of fabrication as a foundry can store large amounts of standard design partially fabricated chips. The designer still has a great deal of flexibility and is only limited by the amount of routing available on the chip. Unfortunately it is inevitable that there will be inefficiencies as some designs require more or less of a specific resource or of routing bandwidth than are available on chip.

3.4 Programmable Read Only Memory (PROM)

One of the first general purpose pre-fabricated chips to achieve widespread use, the PROM consists of an array of one time read only cells pre-programmed with a truth table function. The information on a PROM can be stored using one of three methods

- **Fuse/Antifuse:** Each bit is stored using a fuse/antifuse which can be blown by applying a high voltage. The condition of the fuse/antifuse determines what bit value is read.

- **EPROM (Erasable PROM):** Programmed by application of a high voltage but differs in the fact that it can be erased by exposure to UV light. This allows the PROM to be reprogrammed.
- **EEPROM (Electrically Erasable PROM):** Again, programmed using a high voltage but this time the chip can be erased by purely electrical means.

One other technology well worth a mention here (but not strictly classified as a PROM) is random access memory (RAM). Its characteristics are essentially identical to a PROM except that it is read/write and can be reprogrammed in system on the fly.

3.5 Programmable Logic Devices (PLD)

PLDs were specifically designed to implement logic circuits and typically consist of an array of AND gates followed by an array of OR gates (sum-of-products).

- **PAL (Programmable Array Logic):** The most common PLD with a programmable AND plane followed by a fixed OR plane.
- **PLA (Programmable Logic Array):** A more flexible version of the PAL which allows both AND and OR planes to be programmed.
- **GAL (Generic Array Logic):** A further enhancement of the PLD which can be configured to implement several different types of PAL with optional output inversion.
- **CPLD (Complex Programmable Logic Device):** CPLDs break up complex systems by effectively connecting several PLD elements together using a switching matrix.

PLDs can be implemented using fuse/antifuse, EPROM or EEPROM programming technologies, as described in section 3.4.

3.6 Field Programmable Gate Arrays (FPGA)

FPGAs are fully prefabricated devices designed to implement multi-layer circuits instead of the simple PLD sum-of-product terms. Unfortunately, this results in a less predictable propagation delay as the only limit to on chip circuit complexity is available resources. It should be noted that FPGAs are sometimes referred to as CPLD devices: to avoid ambiguity they will always be referred to here as FPGAs.

FPGAs are not only available in fuse/antifuse, EPROM and EEPROM technologies (section 3.4), but also in an SRAM/DRAM version [3]. SRAM/DRAM versions allow fast in-circuit reconfiguration of the chip but are disadvantaged by the amount of silicon required by RAM technology. This reconfiguration can either be partial or complete (device dependent) and need not even result in the disruption of logic blocks which are not being reconfigured.

4 The FPGA

The FPGA ([4] and [5]) was first introduced in 1985 by Xilinx and since then several other companies have released similar products. This chapter examines standard FPGA architectures and characteristics.

4.1 FPGA Classes

There are four main classifications of FPGA layout, each describing the positioning of both routing and logic blocks. Although interesting on a chip layout level, when designing a logic circuit sophisticated CAD tools are used which translate the design, making the underlying architecture invisible.

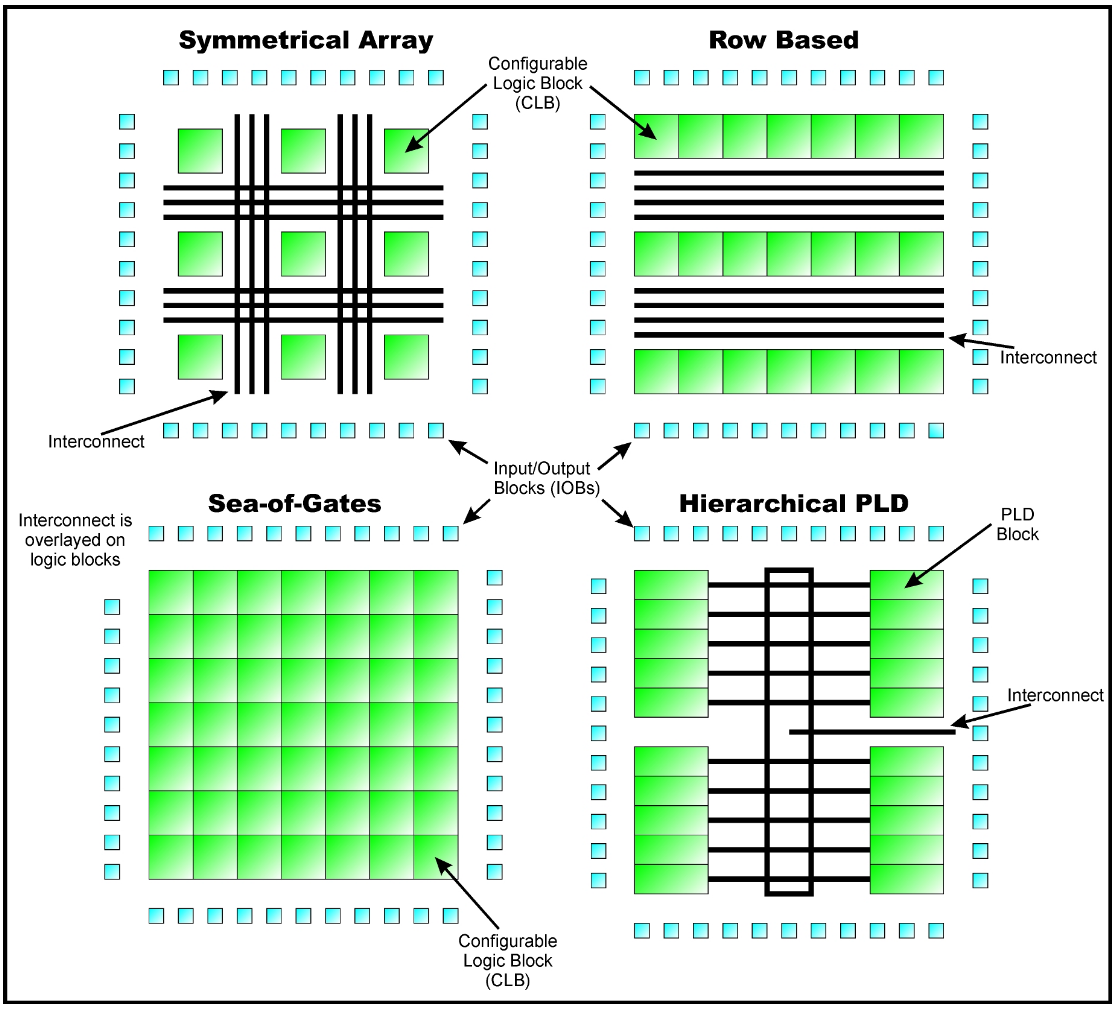
Symmetrical Array
Configurable Logic Block (CLB)
Row Based
Interconnect
Interconnect
Input/Output Blocks (IOBs)
Sea-of-Gates
Interconnect is overlayed on logic blocks
Hierarchical PLD
PLD Block
Interconnect
Configurable Logic Block (CLB)

Figure 2

This diagram shows the four major classes of commercially available FPGAs. IOBs provide the interface between package pins and internal signal lines.

Figure 2 illustrates the four main classes which are commercially available. The choice of architecture is really dependent on application, some being more suited to certain tasks than others.

4.2 The Configurable Logic Block (CLB)

Configurable logic blocks in an FPGA provide the functional elements for construction of a user's logic. The CLB's most important figure of merit is its functionality. Increased functionality allows more complex logic functions to be implemented in a single CLB. However, as the functionality of a single CLB increases so does its size, resulting in fewer CLBs on each FPGA.

CLBs can contain a number of building blocks arranged in a manufacturer specific manner to allow implementation of various logic functions. The standard building blocks are:

- **Look-Up Tables (LUT):** Look up tables take *M* Boolean inputs and give *N* outputs, where *N* is usually 1. The outputs given are chosen to represent some appropriate function. This report will adopt *M*-LUT-*N* to identify a LUT's properties e.g. a 3-LUT-1 is a three input LUT with one output. Such a look-up table could be used to implement a 3 input AND gate as shown in table 1. The number of output bits which need to be stored for any lookup table is a direct function of the inputs: 2^M. Thus a 4-LUT-1 requires 16 stored bits in contrast to the 3-LUT-1 which requires 8. The problem with lookup tables though is that larger ones are to a great degree under-utilised. It has been shown that, for most applications, the optimum LUT is a 4-LUT-1 [4].

Input *M*			Output *N*
A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Table 1

Sample 3-LUT-1 representing a 3 input AND gate.

- **Programmable Logic Array (PLA):** (See section 3.5) For the very reason that LUTs are inefficient with M>5, the area efficiency of PLAs was examined with M inputs, N outputs and K product terms. The optimal PLA was shown [6] to have values of approximately K=10, M=3 and N=12 giving a 4% space advantage over a 4-LUT-1 implementation.
- **D-Type Flip-Flops:** Extremely useful, although not necessary, when any kind of sequential logic needs to be performed. It has been shown experimentally [7] that without a flip-flop in a CLB the number of CLBs required to implement a function approximately doubles.

CLBs can contain other components such as logic gates and multiplexers, however there is no standard implementation methodology with FPGAs being highly manufacturer dependent. Figure 3 illustrates a sample 4-LUT-1 based CLB.

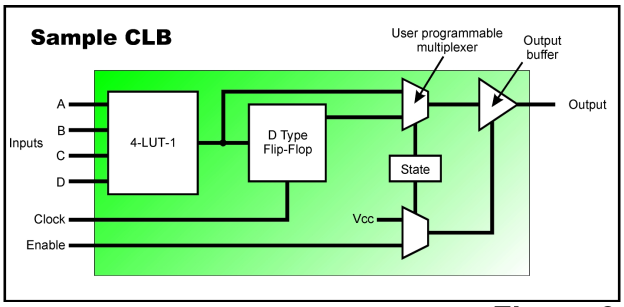

Figure 3

Sample CLB design taken from [7].

The problem that any manufacturer has when designing an FPGA is trading off CLB granularity against functionality. One possible way round this problem is to create a non-homogeneous array of logic blocks, however such an array will almost always have elements that are not suited to a given application. Regardless of CLB configuration, the efficiency of any FPGA is highly reliant on its CAD tools. If CAD tools do not map a logic circuit optimally onto the CLBs then any architectural advantage will be squandered.

4.3 FPGA Routing

FPGA routing deals essentially with the connection of CLBs together with other CLBs and IOBs to provide the necessary functionality. The balance between area given to routing and to CLBs is critical: too much to CLBs and it will not be possible to wire complex systems together, too little and routing will remain unused. On a standard FPGA, routing will normally occupy 70-90% of all the available area. Routing is expensive in terms of area and delay since the programmable switches take up a significant area and have appreciable resistance and capacitance.

There are two types of block used in routing an FPGA which are defined as connection blocks (CBs) and switch blocks (SBs). Figure 4 shows how a symmetrical CLB array architecture is constructed using these two additional blocks.

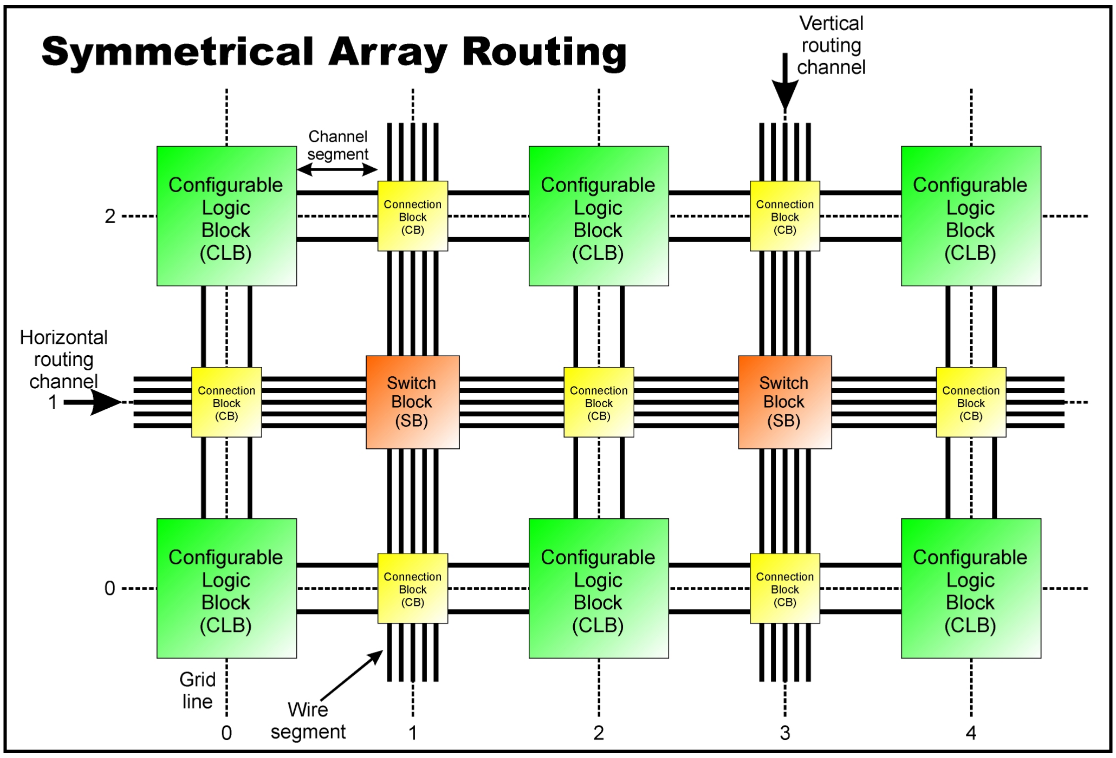

Figure 4

Connection blocks and switch blocks can be configured to route IO from CLB elements (figure 3) onto routing channels.

The internal connections possible in both CBs and SBs are again manufacturer dependent. A manufacturer must consider the amount of routing their architecture requires and optimise CB/SB connectivity to allow maximum interconnection flexibility whilst keeping redundancy to a minimum.

A sample connection block (CB) is shown in figure 5. A configurable routing switch is

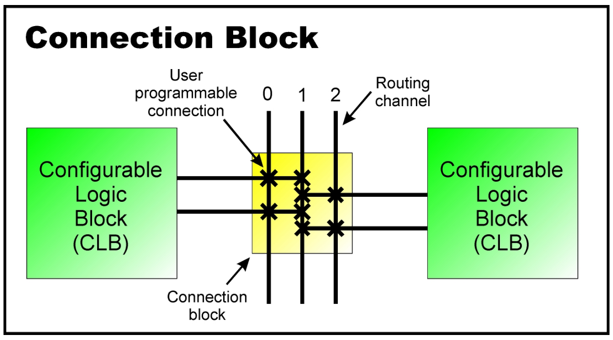
Connection Block
User programmable connection
0
1
2
Routing channel
Configurable Logic Block (CLB)
Configurable Logic Block (CLB)
Connection block

Figure 5

Sample connection block where X represents a possible connection to a routing indicated by an X, the CB itself using a particular interconnection topology.

Flexibility is a very important issue here as too few routing switches could result in it being impossible to interconnect two CLB pins.

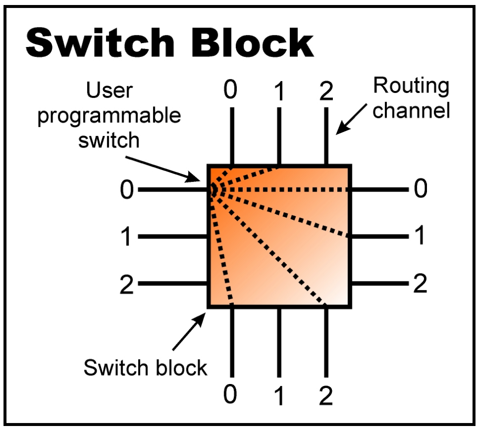

Figure 6

Sample switch block with all possible switch configurations from one input shown.

Figure 6 shows a sample switch block (SB). The flexibility of a switch block is defined as the number of wire segments any incoming wire segment can be connected to: in the example show this is *F*=6. When designing a switch block topology the designer must be very careful to chose a topology that does not prohibit two CLB pins from being connected together.

Routing in an FPGA is normally classified into one of three categories which define its primary use. These are:

- **General purpose interconnect:** For connections that span one or more local CLBs. Implemented using routing channels, connection blocks and switching blocks as described above. Unfortunately, each switch or connection block through which a signal must pass has an associated *RC* delay – this can be a serious problem at higher frequencies.
- **Direct interconnect:** Provides a direct connection with one or all of a CLBs neighbours to either the right, left, top or bottom.
- **Long lines:** Used to route connections that need to span several CLBs with low skew that would otherwise traverse several routing switches.

One final but very important consideration in FPGA routing is I/O blocks (IOBs), without which an FPGA would be useless. Therefore an IOB must be chosen such that it supports various output modes, TTL or CMOS I/O and provides enough drive current for any attached device to function, but not so much that the FPGA's power consumption is greatly increased. Unfortunately, I/O is a weak link in FPGAs and has recently improved disproportionately to chip density.

4.4 Commercially Available FPGAs

To illustrate currently available technology, table 2 contains information on state-of-the-art FPGAs.

Device	Architecture	SRAM (bits)	IOBs	Ref.
Xilinx XC6264	Sea-of-Gates, FPGA.	16,384	512	[8]
Dynachip 6055	Symmetrical array, FPGA.	51,200	320	[9]
Altera Apex EP20K1000E	Symmetrical array, PLD.	540,672	780	[10]
QuickLogic QL4090	PLD.	25,344	316	[11]
Lucent Orca OR3T165	Sea-of-Gates, FPGA.	134,144	512	[12]

Table 2

State-of-the-art FPGA technology as of June 1999.

5 Dynamically Reconfigurable Computing

Dynamically reconfigurable field programmable gate arrays (FPGAs) combine, in principle, the speed of a dedicated hardware solution with the flexibility of software ([13], [14] and [15]). Such FPGAs can be reconfigured by an external controller to implement any desired set of Boolean operations [16]. This is the concept behind dynamically reconfigurable computing: combine the flexibility of software with the speed of hardware [17]. At present there is growing interest in the technology: two companies, Triscend [18] and Starbridge Systems [19], have both recently released computer systems which take advantage of dynamically reconfigurable computing. Although the field is still emergent [20], this section examines ([21] and [22]) the criterion which are used to classify a reconfigurable computer system.

5.1 The Aims of Reconfiguration

There can be two reasons for using dynamic reconfiguration:

- **Speed Improvement:** Increase performance through a custom architecture applied to a specific problem.
- **Fault Tolerance:** To improve the manufacturing process – design alterations or problems are easily overcome.

5.2 Granularity

Classifies the size and complexity of the smallest block in any reconfigurable device:

- **Fine Grained:** Consists of small and simple logic blocks which are configured to perform a more complex operation.
- **Medium Grained:** Consists of complex logic blocks which can each perform a significant part of any calculation. Non-standard calculations are performed by reconfiguration of the interconnection network.

- **Coarse Grained:** Consists of a set of execution units each with their own set of instructions. Each unit is simpler than a microprocessor but integrated tightly enough to give high communication speeds ([23] and [24]).

5.3 Integration

Specifies how a reconfigurable system is coupled to its host.

- **Dynamic Systems:** Bio-inspired systems which are not controlled by an external device. The idea is that such systems evolve themselves.
- **Static, Closely Coupled:** Reconfigurable units are closely coupled as execution units on a host processor's datapath.
- **Static, Loosely Coupled:** The reconfigurable units are situated on a separate board from the host. This is generally detrimental to any speedup as data must be transferred to and from the daughterboard.

5.4 Interconnection Network Reconfigurability

Specifically, the reconfigurability of an external interconnection network between reconfigurable units.

- **Reconfigurable External Network:** The concept of reconfiguration can be extended over several reconfigurable circuits effectively providing a large reconfigurable unit. However, the penalties for going off-chip are high.
- **Fixed External Network:** Fixed connections exist between reconfigurable circuits effectively limiting flexibility in order to maximise speed and minimise cost.

6 Optoelectronic Interconnects

Dynamically reconfigurable (FPGAs) are normally reconfigured by an external controller to implement a desired circuit. Reconfiguration can be performed with speeds at or near to real time, depending on the extent of reconfiguration and on the time taken to download configuration data onto the FPGA. As with all VLSI systems, the increasing density and speed of silicon circuits frequently transfers the performance bottleneck of a system to its communications – FPGAs are no exception. To make dynamically reconfigurable computing at all viable, new FPGA configurations must be downloaded at a rate which puts the component out of action for the shortest possible time period. Optoelectronic interconnects are widely considered as a potential solution to the problem and are already being deployed (at a crude level) in commercial systems. Aside from the potential of optics for raw data throughput (unattainable in conventional systems), what is perhaps more exciting is the enabling of new architectural concepts by a combination of this throughput with the potential to reconfigure at high speed.

This section considers the advantages of optoelectronic interconnects and the associated benefits they could bring, not just to reconfigurable computing but to VLSI interconnection as a whole.

6.1 The Aim of Optoelectronics

The difference between electronics and optics is essentially the difference between the electron and the photon (figure 7). Since electrons carry mass and charge they interact very strongly with each other making them ideally suited to switching. Photons, on the other hand, do not carry mass or charge and are therefore non-interacting in free-space. This makes them ideally suited as long

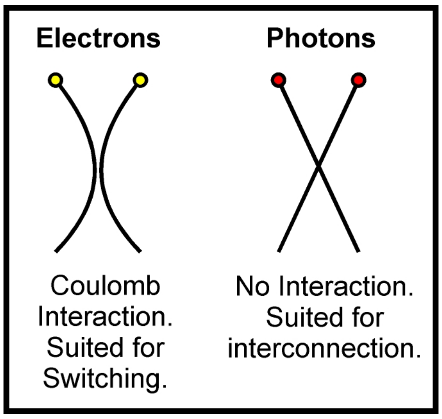
Electrons
Photons
Coulomb Interaction. Suited for Switching.
No Interaction. Suited for interconnection.

Figure 7

Photons are non-interacting in free space.

distance data carriers for dense signals (the telecommunications industry has largely abandoned electrical interconnection in favour of fibre optics.) Optoelectronics attempts to make the best of both of these worlds, however there will be a point below which conversion into the optical domain is wasteful both in terms of transmission rates and money.

The free space optical systems considered here are constructed by flip-chipping optical detectors and emitters (or modulators) onto silicon circuitry (see section 7.3) so the whole ensemble can be viewed as conventional silicon equipped with a large number of "optical pins" normal to the chip surface.

The rest of this chapter assesses the advantages and disadvantages of optical interconnection.

6.2 Propagation Speeds

In an attempt to compare and contrast, we can examine the propagation speed of an optical signal against that of an electronic one in a media of refractive index *n*. Equation 1 describes the velocity at which any signal propagates where *c* is the speed of light.

$$v = \frac{c}{n}$$ **Equation 1**

In the electronic case *n*≈1.2-1.5 whereas a fibre optic core has *n*≈1.4-1.6 proving that, if anything, optical signals will propagate slightly slower. The only difference is when free-space optics are used, where *n*≈1.0, but this is still not a highly significant difference when the length of any practical system is considered. The advantages of optics lie elsewhere.

6.3 Impedance Matching

Any high-speed electrical transmission line has impedance matching criteria where a resistive load is used to match the real impedance of the transmission line. Regardless of how a line is terminated, if correctly impedance matched no reflections will occur and all power will be absorbed within the terminator: giving rise to thermal concerns. At the transmitter, an appropriate drive current must be supplied resulting in further thermal concerns, reducing physical packing density. To serve as an illustration, let us consider a 64-bit wide bus of 50Ω terminated transmission lines with a 5V swing operating at 50% duty cycle. Equation 2 calculates the amount of power dissipated in the terminating resistors and equation 3 the required drive current.

$$64 \times \frac{1}{2} \times \frac{V^2}{R} = 16W$$ **Equation 2**

$$64 \times \frac{V}{R} = 6.4A$$ **Equation 3**

This is quite a considerable figure to add to most power supply requirements.

For both free-space and wave guided optics, the impedance matching criteria are considered to be met when the full signal is absorbed without reflection [25]. This can be done by fabricating a section of the transmission media to be exactly one quarter of a wavelength thick (or an integer multiple plus one quarter) and to have an impedance between that of free space and the detector. Such optical impedance matching devices are called anti-reflective (AR) coatings.

The major advantage though is that an optical receiver operates in a non-dissipative manner [26]. Any photons incident on a photodetector are converted into charge based on the device's quantum efficiency β, with β~1 feasible, resulting in little excess heat. Given that typical photodetectors generate ~0.4 to 0.5A/W of incident light, a 1MΩ loaded photodetector will generate the current shown in equation 4.

$$V = IR \Rightarrow I = \frac{5}{1E^6} = 5\mu A$$ **Equation 4**

As little as 10μW of incident light could generate 5V. Of course, the response of a photodetector is highly dependent on its configuration: for example, the frequency response is *RC* dependent and as the load resistance increases its frequency response decreases. The problem with heat dissipation in optical interconnection does not lie in the reception but rather emission. One of the most promising technologies, the vertical cavity surface emitting laser

(VCSEL), still does not have the necessary power conversion efficiencies to eliminate excess heat: β~0.1 (10%) in commercially available devices (can be β>0.5 in lab situations). According to the MEL-ARI Optoelectronics Roadmap [27], a standard 64 element VCSEL array transmitter operating at 50% duty cycle with a 4mA operating current at 2.4V would thus produce the excess heat as shown in equation 5:

$$64 \times \frac{1}{2} \times V \times I \times (1-\beta) = 0.28W$$ **Equation 5**

Each VCSEL in this illustration produces 1mW of optical output power, two orders of magnitude greater than that required for detection. Nevertheless, this is still a lot less dissipated power than standard transmission lines generate even using a low efficiency VCSEL – the difference is that excess thermal energy must be dealt with at the transmission end.

6.4 Interconnection Bandwidth Limitations

As the length and/or density of electrical interconnection increases they begin to suffer from increased wire resistance, residual wire capacitance from fringing fields and inter-wire cross-talk. The maximum bandwidth limit of any electrical system, regardless of interconnection scheme, can be derived from physical principles and is described by equation 6.

$$B_{\max} \approx B_0 \frac{A}{L^2}$$ **Equation 6**

Here B_{max} is the maximum bandwidth, B_0 is a constant of proportionality, A is the interconnect's cross-sectional area and L is the interconnect length. B_0 has been independently estimated [28] to be approximately 5×10^{14} for electrical systems where A/L^2 is the aspect ratio. Thus the maximum bandwidth for a 10cm off-chip electrical connection is around 150GHz.

When optics are used this limit simply does not apply. In free space, optics propagate by definition without a guiding medium and with an attenuation significantly smaller than that seen in electronics. The parallelism available is estimated to be >100,000channels/cm^{-1}. Admittedly, the connections from data source to VCSEL or detector to data destination still need to be electrical: but when this distance is 10 to 100 microns in a flip-chip bonded system (as opposed to a few centimetres in an electrical system) the limitation is, in practice, irrelevant. The number of optical pins that can be driven depends primarily on thermal and real estate considerations. Currently we can drive 4,096 channels from 1cm^2 and see no real obstacle to reaching >10,000 channels. Rates of 200MBs^{-1} (CMOS limited) have been demonstrated giving a bandwidth of approximately 20Tbs^{-1}. However, individual devices may routinely be driven at 10Gbs^{-1} so the relay could essentially handle 1,000Tbs^{-1} if we were not CMOS limited (the theoretical limit is actually much higher).

Optical channels can be driven at speeds much greater than that of CMOS and take considerably less power to drive than pads and wire bonds. What should be remembered though is that optical interconnection of a chip is an enhancement and does not preclude conventional electrical connection as well.

6.5 Non-Local Interconnection

In 1971, Stone developed a sorting algorithm for parallel computing systems which has, to this day, not been surpassed as far as a minimal rate of growth of computational steps is concerned. The algorithm is based on work done by Batcher in 1968 called the bitonic merge-sort, Stone adapting Batcher's work for a shuffle exchange network ([29] and [30]). The interconnection methodology is generally known as "Stone's perfect shuffle". The major disadvantage is that to implement this architecture on a large scale the amount of electrical interconnection layout becomes prohibitive as can be seen in figure 8(a). It is this very interconnection problem that limits any

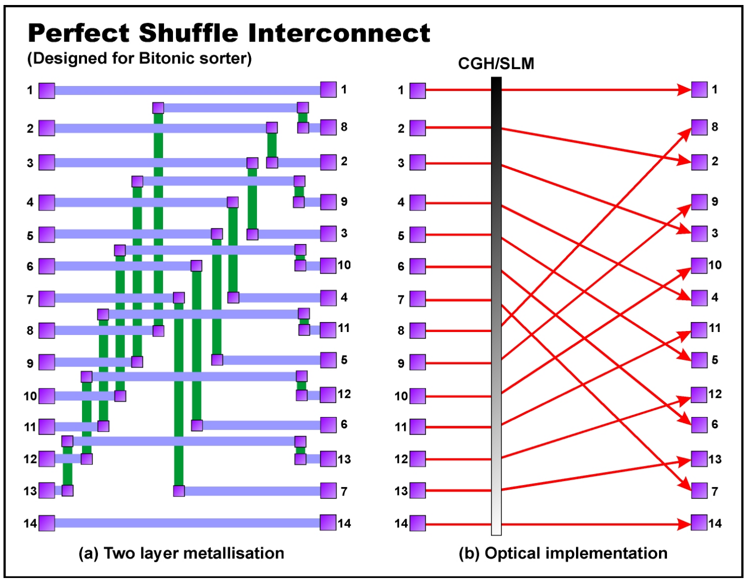
Perfect Shuffle Interconnect
(Designed for Bitonic sorter)
CGH/SLM
1 2 3 4 5 6 7 8 9 10 11 12 13 14
1 8 2 9 3 10 4 11 5 12 6 13 7 14
(a) Two layer metallisation
(b) Optical implementation

Figure 8

Perfect shuffle interconnect implemented in (a) electronics and (b) optics.

implementation of highly interconnected concurrent electronic systems and where optical interconnection comes into play. Figure 8(b) shows how a computer generated hologram (CGM) or spatial light modulator (SLM), which are both optical elements that can deflect an input beam to a pre-selected target, could be used to rout optical data channels appropriately through free space. The non-interacting nature of free space optical channels means that they can pass through each other to form any desired interconnection topology without cross-talk. Interconnects such as the perfect shuffle and the hypercube thus become relatively simple to implement and since the interconnects are more direct and interleaved, the amount of routing and its complexity is reduced with skew becoming less of a problem.

6.6 Optical Alignment

The drawback with any optical system is alignment: all components must have a particular tolerance which is directly related to cost. These tolerances need to compensate for a variety of effects such as thermal expansion/contraction, long term “creep” and environmental conditions such as mechanical vibration from cooling fans etc. One way round these problems is to use adaptive optics (AO) [31] which perform measurement and correction of focusing and positional error in real time. The commercial viability of such techniques is easily seen by looking at a CD player [32], generally regarded as a disposable piece of machinery, which maintains focus and position of a light spot in real time on a rapidly rotating optical disk.

6.7 Conclusions

Free space optical interconnects appear to offer tremendous advantages over electronics: they are attenuated far less than electrical signals, offer transmission lengths of the order of meters without significant driving powers and their naturally parallel nature implies high parallelism around any implemented machine: e.g. to and from memory and/or peripherals.

However, there are still many engineering issues to be confronted before such interconnects can be routinely deployed.

7 Optically Interconnected DR-FPGAs

To offset the limited I/O bandwidth of FPGAs, optical I/O has been proposed as a means of both FPGA configuration and data I/O. It should be noted that optical I/O is not currently commercially available for FPGAs and this section is written from a theoretical/research point of view.

7.1 Dynamically Reconfigurable Systems

So far we have examined the advantages of optics, but these are general advantages and not specific to dynamically reconfigurable systems. There are two additional reasons why optical interconnects are specifically of interest to DR-FPGAs:

- **Bandwidth:** FPGAs are routed dynamically and each I/O channel must therefore traverse both switching and routing blocks, each with an associated *RC* delay, to reach an I/O block. This can result in serious bandwidth limitations. Optical I/O can relieve associated bottlenecks.
- **Reconfiguration:** To make dynamically reconfigurable computing at all viable, new FPGA configurations must be downloaded at a rate which puts the component out of action for the shortest possible time period. If highly parallel optical I/O was used to download new configurations, reconfiguration times could be minimised thus making best use of all available processing time.

7.2 CLB Optical I/O Mapping

The systems proposed here consist of three basic stages. The first stage we will consider as the *input* stage which will consist of a detector that is capable of receiving a digital optical input be it from free space or waveguide. The second stage is the *processing* stage and consists of a dynamically reconfigurable FPGA system which could be anything from a single CLB (figure 9) to an array of CLBs (figure 10). The final stage is the *output* stage and consists of an optical emitter or modulator (VCSEL [33], SEED [34]) with digital output to either free space or a waveguide. This combination of stages will be referred to as an *element*.

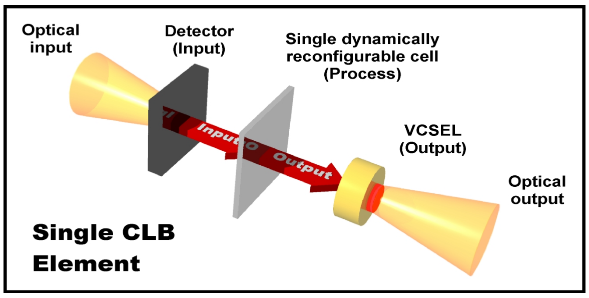

Figure 9

Element with one dynamically reconfigurable CLB.

To make full use of parallelism, these *elements* are arrayed in 2D: i.e. one detector array, one FPGA and one VCSEL array. The processing stages (FPGA) of each *element* are capable of communicating with one-another electronically, or with any other local electronics for that matter, and can be considered as a standard dynamically reconfigurable FPGA with an extra optical input and output available to a specific CLB or set of CLBs.

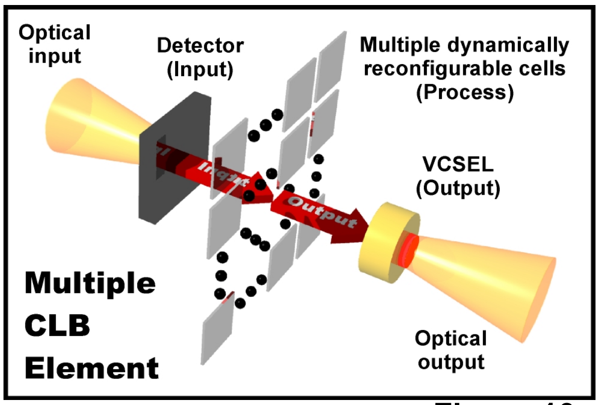

Figure 10

Element with multiple dynamically reconfigurable CLBs (gate array).

Any CLB has the potential to be reprogrammed by its optical input stream if configuration information is interlaced using a predefined protocol, or to reprogram another CLB in another system by interlacing the same configuration information onto its optical output stream.

The systems described here will be referred to from now on as Optical FPGAs (OFPGAs).

7.3 Flip-Chip Bonding

Unfortunately there are a few problems with constructing a practical I/O system: although it is possible to implement a detector (input) on silicon it is not possible to implement VCSELs (output) due to an insufficient bandgap energy difference in silicon. In general the optoelectronic interface is

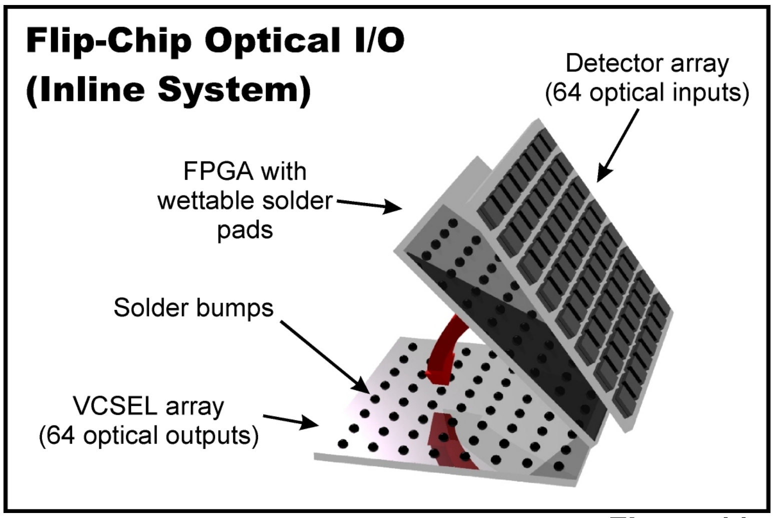
Flip-Chip Optical I/O (Inline System)
Detector array (64 optical inputs)
FPGA with wettable solder pads
Solder bumps
VCSEL array (64 optical outputs)

Figure 11

Flip-chip bonding an FPGA with optical input and output.

constructed of hybrid processing chip technologies, which employ GaAs optical chips hosting detectors and emitters, flip-chipped [35] on top of the Si FPGA. The combination of input, processing and output elements is generally known as a smart pixel. To connect both substrates they must be placed on top of one another and heated to reflow the solder bumps (which are normally 15-30µm in diameter) thus creating a connection. Such high temperatures can be detrimental to the chips and create a strained connection when attached due to thermal expansion coefficient mismatch. Figure 11 shows a three layer chip which makes visualisation of circuits simpler (may soon become unnecessary as newer technologies are capable of fabricating MSM photodetectors onto the same substrate as the VCSEL array).

7.4 Optical I/O Architectures

This section considers various architectural issues which may arise when constructing a dynamically reconfigurable system that uses optical I/O.

7.4.1 On-Board Optical IO

Refers to a single PCB with both optical input and output elements as shown in figure 12. Board level I/O has the advantage that standard parts (as far as FPGA and associated circuitry are concerned) can be used and heat dissipation can be easily handled. The major disadvantage with such devices is their bandwidth. We now have the problem of transmission lines between FPGA and optical input/output stages which will throttle any increased bandwidth potential that optical I/O may bring.

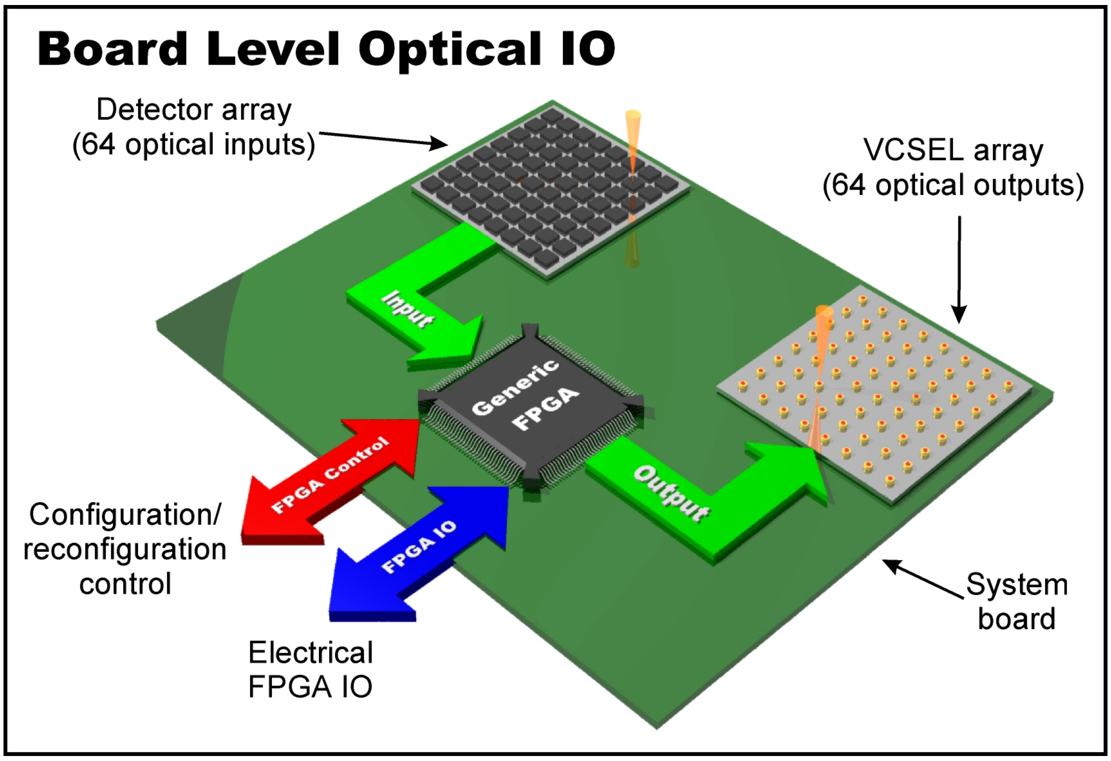
Board Level Optical IO
Detector array (64 optical inputs)
VCSEL array (64 optical outputs)
Input
Generic FPGA
Output
FPGA Control
FPGA IO
Configuration/ reconfiguration control
Electrical FPGA IO
System board

Figure 12

A board level system could incorporate standard components.

7.4.2 Flip-Chip Optical I/O

Using flip-chip bonding, it is possible to sandwich an FPGA chip to create a three layer smart pixel (figure 13). This report will refer to such a configuration as an *inline* system from now on. If the FPGA is dynamically reconfigurable then the inline system could be used for applications such as a telecommunications router ([36] and [37]) where each element receives an input data stream and configures itself to electronically send the stream to a target output element. Similarly, an inline DSP-like

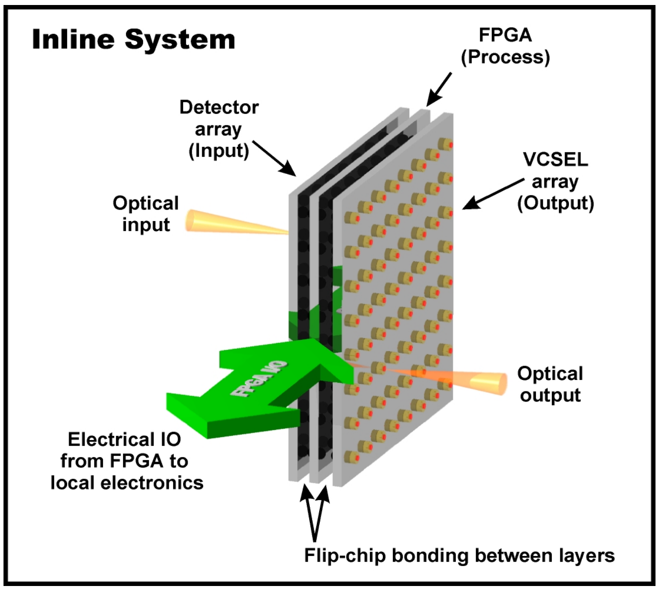

Figure 13

Inline system with FPGA sandwiched between optical input and output elements.

system [38] could reconfigure its hardware to perform signal processing on an incoming data stream such as a parallel optical bus.

The real advantage of a system using flip-chip bonding is that there are no transmission lines, therefore its potential bandwidth is consequently higher.

7.4.3 Integrated Optical Input

There have been various studies on integrating optical reconfiguration input onto Si FPGAs with a high degree of success ([39], [40], [41] and [42]). However, fabricating VCSELs on the same chip is obviously a problem since Si's bandgap is insufficient to support lasing. This still has potential however as configuration download times to the chip can be minimised even if the chip lacks optical output.

7.4.4 On-Chip Long Lines

Long lines on an FPGA are used for signal distribution on chip over a longer distance. The speed and bandwidth of optics is such that going off chip (and consequently saving routing resources) may prove a viable alternative to using long lines.

7.5 Conclusions

One final point is how do we map optical I/O to an OFPGA architecture: should the input/output be treated as a one-way IOB or as an integral part of a CLB to which information can be latched? Regardless of implementation, the constraining factor in any practical future system using high frequency optical transmitters is heat dissipation.

8 Optoelectronic DR Systems

The possible role of FPGAs within three optical systems is examined here. Of the three systems described, only the second, the neural network, has actually been constructed.

8.1 The Optoelectronic Sorting Demonstrator

A sorting module [43] was chosen as a demonstrator following suggestions from the computing community that this kind of task would be "interesting" to them. The architecture of the demonstrator utilises optoelectronics as described above and exploits a non-local interconnect, in this case the perfect shuffle. (This sorter is the direct descendent of the optical CLIP experiments [44]). Figure 14 shows a schematic of the sorting demonstrator. The data to be sorted are entered sequentially into the processing loop through electrical

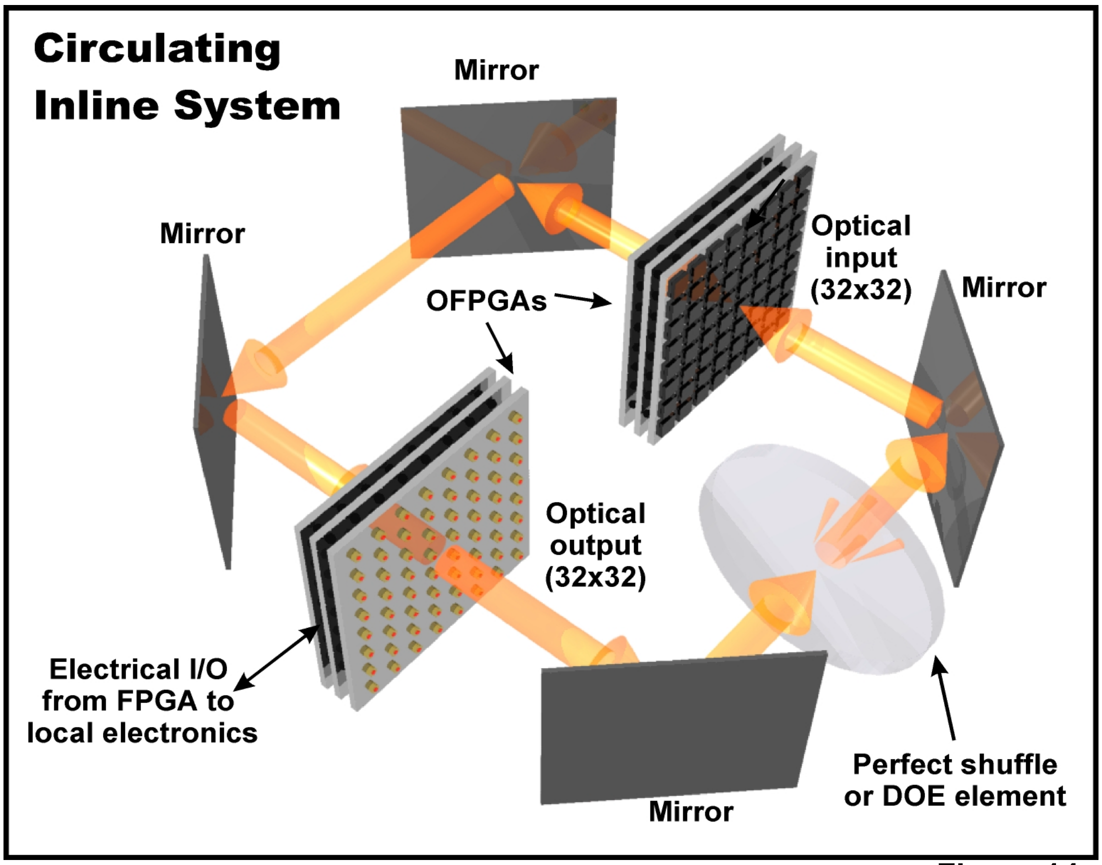
Circulating Inline System
Mirror
Mirror
OFPGAs
Optical input (32x32)
Mirror
Optical output (32x32)
Electrical I/O from FPGA to local electronics
Mirror
Perfect shuffle or DOE element

Figure 14

Optoelectronic sorting demonstrator modified to include inline components. NB. Components not to scale – 8x8 components shown for clarity.

I/O as shown. Sixteen bit planes of 32x32 bits (the number of optical communication channels) may be entered in this version. At run time, the 2D perfect shuffle is performed by a lens operation during each cycle of the machine and all computations are performed in parallel by the FPGA. The total number of cycles scales as $(\log N)^2$ for a Batcher's bitonic sort of N data points. At the conclusion of the computation, the sorted set of data can be sequentially downloaded to the electronic domain.

Each inline array can convert electrical or optical inputs into electrical or optical outputs. Four modes of operation are therefore possible. During the processing, the array receives optical data streams, stores or processes them electronically and outputs them optically. The array can also function as an I/O device by using the electronic-input/optical-output and optical-input/electronic-output modes of operation respectively.

A simpler system in which one of the inline devices is replaced with RAM (and a suitable optical interface) gives the ability to connect memory in parallel to the FPGA and thus potentially reconfigure in a single cycle across the whole array. Calculations within the array could be used to supply the next lot of stored reconfiguration data – also in parallel. Alternatively the optical pathways could be used to supply an inordinately high throughput to a previously configured chip. In this case there is no requirement for the circuit to be looped. Other authors have concluded that advantages may be gained with a single memory to FPGA optoelectronic interface [40]

We have shown a sorting system built using two inline devices; however it is the uses of the FPGAs reconfigurable aspects that are of greatest interest here. A first level of flexibility could be introduced in that the CLBs could be reconfigured to optimise execution time for different array sizes [45]. As a further level, the iterative perfect shuffle shown here forms (with suitable node switching) an omega class network capable of arbitrarily permuting data. Bearing this in mind, the set up could be used to implement any algorithm or multistage switching function given the right logic configuration. Indeed it is known that many algorithms map exceedingly efficiently onto this topology (the FFT being the classic example) and performance results at this level of parallelism would be of some interest to the engineering community.

Therefore the combination of the FPGA and the high bandwidth optical interconnect gives us a general purpose fine grained massively parallel processor. The sorting demonstrator system described here has an off chip data rate of 200 Gbs^{-1} which is what will be required by the semiconductor industry (according to the SIA roadmap) in 2007.

8.2 The Neural Network Packet Switch Controller

The second example addressed here is the neural network switch controller. An experimental version of this which showed a performance commensurate with state of the art all electronic switches has actually been built [46] (albeit with discrete components) and a novel version using smart pixels and FPGAs is under construction.

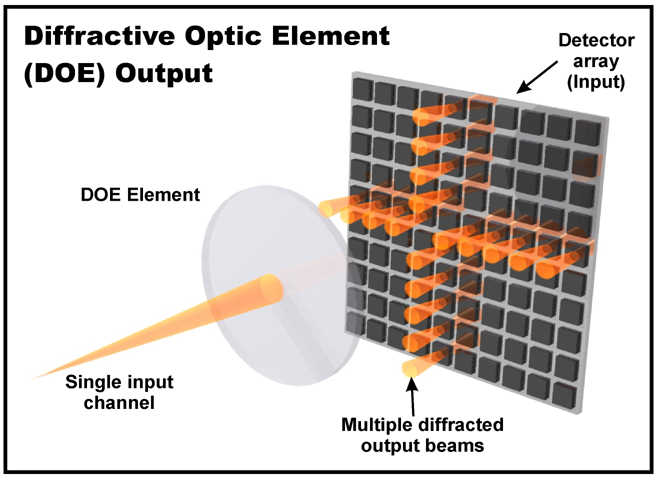
Diffractive Optic Element (DOE) Output
Detector array (Input)
DOE Element
Single input channel
Multiple diffracted output beams

Figure 15

Illustration of a single input beam imaged onto multiple detectors using a DOE.

Although the physical layout of the neural switch controller is somewhat different to the sorting module described above, from an architectural perspective the neural network may be formed from the module merely by replacing the optics that are used to generate the perfect shuffle with a diffractive optical element (DOE) [47] such as that shown in figure 15. The effect of the element is to fan-out in a space invariant manner every input channel into the pattern shown. This may readily be seen to effectively amount to an analogue sum over rows and columns of the outputs of the previous inline device. The system, as it stands at present, has a fixed set of weights and is designed to perform this single task. The inclusion of the FPGA stages will allow, in the first instance, programmed weights to be considered which would allow the network to be configured for a wider variety of tasks such as the travelling salesman problem or other optimisation problem. As in the above example however, the most exciting possibility is being able to reconfigure these weights in near or at real time so that fully adaptive, supervised and unsupervised learning schemes may be implemented.

So the combination gives us a neural network with fully real time adaptive weights with an interconnection density that could never be achieved electronically.

8.3 The Generic Multiprocessor Harness

The concept of optical highways [48] is of perhaps the most interest in that a general purpose multiprocessor solution can be envisaged. Figure 16 schematically shows a highway used to connect nodes in an arbitrary topology, with each node having access to >1,000 channels. The interconnect is point to point and hard wired, with several thousand channels being passed from each node via a smart pixel interface into a free space optical relay system which can hold (under reasonable assumptions regarding aberrations) several hundred thousand channels. Polarising optics are used to determine the position of every individual interconnection link thus defining the computational topology.

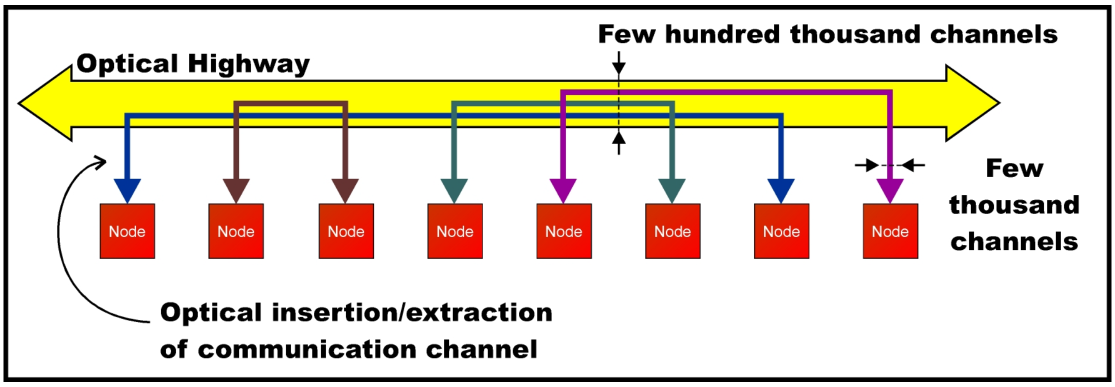
Few hundred thousand channels
Optical Highway
Node
Node
Node
Node
Node
Node
Node
Node
Few thousand channels
Optical insertion/extraction of communication channel

Figure 16

Optical highway for connection of multiple processing nodes

The data used in this design is extrapolated directly from the components used in the sorting demonstrator (number of optical channels off chip is 2,500, clock speed 250MHz, data width is 64 bits) which shows the highway to support bisection bandwidths in excess of 10,000 Tbs^{-1} connecting up to 1 million nodes (depending on topology) [48]. Each node might consist of a microprocessor, memory and cache, and one or more OFPGAs to handle communications.

The role of OFPGAs in this architecture is to optimise the communications and processing in real time during execution of a range of algorithms. The bandwidth of the communications in this system is sufficiently high to implement a flat memory model, but superior performance on particular algorithms may be obtained by changing topology in the interconnect harness. In essence, the destination of any data channel output into the optical domain is determined by the spatial location of the emitter output upon the OFPGA. Thus by re-routing signals within the OFPGAs, a particular global topology may be established. The machine could of course be configured arbitrarily into several differently connected regions if this was desirable. In addition, the functionality of the interface may be changed, for example, OFPGAs allow us to configure the interface as a router (necessary if we wished to utilise a hypercube in the optical domain) or as a simple switch of high throughput (necessary if we wished to use a large crossbar in the optical domain). For nodes of sufficient complexity, the maximum throughput of the system may often be attained by changing the width of data words as well as the topology so as to keep all communication channels busy. OFPGAs could support variable width multiplexing as well as routing at the interfaces.

For the generic interconnect the combination of FPGAs with high bandwidth optoelectronics enables an intelligent communications interface to be constructed which allows maximisation of the ultra high throughput available. In turn this enables real time optimisation and load balancing of the whole machine over a range of computational models.

9 Conclusion

Dynamic reconfiguration is becoming ever more popular as a means of high speed processing and as an increasing amount of companies begin to release such systems the issues of bandwidth will become ever more prominent for the reasons described above. The author believes that the only way to overcome associated bandwidth limitations is by utilising optical interconnection, be it through free-space or waveguides: without optical interconnection, dynamically reconfigurable computing will hit a premature performance ceiling.

10 Glossary

AO	Adaptive Optics
AR	Anti Reflective
ASIC	Application Specific Integrated Circuit
CAD	Computer Aided Design
CB	Connection Block
CD	Compact Disk
CLB	Configurable Logic Block
CMOS	Complementary Metal-Oxide Semiconductor
CPLD	Complex Programmable Logic Device
DR	Dynamically Reconfigurable
DRAM	Dynamic Random Access Memory
EPROM	Erasable Programmable Read Only Memory
EEPROM	Electrically Erasable Programmable Read Only Memory
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
GAL	Generic Array Logic
I/O	Input/Output
IOB	Input/Output Block
ISP	In-System Programmable
LUT	Look-Up Table
MPGA	Mask Programmable Gate Arrays
OFPGA	Optical Field Programmable Gate Array
PLD	Programmable Logic Device
PAL	Programmable Array Logic
PCB	Printed Circuit Board
PLA	Programmable Logic Array
PROM	Programmable Read Only Memory
RAM	Random Access Memory
ROM	Read Only Memory
SB	Switching Block
SEED	Self Electro-optic Effect Device
SRAM	Static Random Access Memory
TTL	Transistor-Transistor Logic
UV	Ultra-Violet
VCSEL	Vertical Cavity Surface Emitting Laser
VLSI	Very Large Scale Integration

11 References

This reference list includes some comments on certain entries. The idea is to help assess the relevance of any source before it is looked up. Some comments require an understanding of the systems described in this report.

[1] C. Maxfield, *"Logic that mutates while-u-wait"*, EDN, ISSN 0012-7515, volume 41, number 23, 7th November 1996, pp. 137-40, 142.

The advent of SRAM-based FPGAs presented a new capability to the electronics community: dynamically reconfigurable hardware, or designs that you can reconfigure on the fly. As devices, FPGAs can be difficult to characterise, because each FPGA vendor fields a proprietary architecture. However, a generic architecture is given to illustrate the sophistication of FPGAs as compared to traditional PLDs.

[2] S. Hauck, *"Multi-FPGA Systems"*, PhD Thesis, University of Washington, 1995.

Chapter 2 provides a detailed and precise overview of current technologies.

[3] M. Motomura, Y. Aimoto, A. Shibayama, Y. Yabe and M. Yamashina, *"An embedded DRAM-FPGA chip with instantaneous logic reconfiguration",* Proceedings of IEEE Symposium on FPGAs for Custom Computing Machines, ISBN 0 8186 8900 5, IEEE Computer Society, 1998, pp. 264-6.

Reconfigurable computing is attracting wide attention as a novel general purpose computing paradigm for accelerating compute intensive and/or data-parallel applications, such as compression, encryption, searching, sorting, and image processing. A key enabling technology for a reconfigurable computer is in-system logic reconfiguration of SRAM-based FPGAs, through which its hardware architecture is dynamically customised for a specific task on demand. Quicker a reconfiguration is, more frequent the reconfigurations can become: i.e., a reconfigurable computer can adapt to applications which have more dynamic behaviour. A whole-chip reconfiguration in conventional FPGAs, however, takes at least 100 mu s. With this long latency, a reconfigurable computer is adaptable only to static applications, substantially losing the general-purposeness of the original concept. Integrating a DRAM with an FPGA can become an ideal solution to this problem. The on-chip DRAM can store hundreds of configuration programs, and the logic reconfiguration can get extremely faster by context-switching among the programs utilising huge bandwidth internal to the DRAM core. Being driven by this observation, we have conducted prototype design of an embedded DRAM-FPGA chip.

[4] S. D. Brown, R. J. Francis, J. Rose and Z. G. Vranesic, *"Field-Programmable Gate Arrays"*, Kluwer Academic Publishers, USA, 1993.

Very informative book on FPGAs, if slightly dated.

[5] J. V. Oldfield and R. C. Dorf, *"Field Programmable Arrays: Reconfigurable Logic for Rapid Prototyping and Implementation of Digital Systems"*, John Wiley and Sons, 1995.

[6] J. Kouloheris and A. El Gamma, *"FPGA Area vs. Cell Granularity - Lookup Tables and PLA Cells"*, First ACM Workshop on Field

Programmable Gate Arrays, FPGA '92, Berkeley, CA, February 1992, pp. 9-14.

[7] J. S. Rose, R. J. Francis, D. Lewis and P. Chow, *"Architecture of Programmable Gate Arrays: The Effect of Logic Block Functionality of Area Efficiency"*, IEEE Journal of Solid State Circuits, volume 25, number 5, October 1990, pp. 1217-1225.

[8] Xilinx Corp., "XC6200 Field Programmable Gate Array", April 24th 1997.

[9] Dynachip, *"DY6000 Family FAST Field Programmable Gate Array"*, Datasheet, Revision 1.1, http://www.dyna.com.

[10] Altera, *"Apex 20K Programmable Logic Device Family"*, Datasheet, Version 2, http://www.altera.com, May 1999.

[11] QuickLogic, *"QL4090 9,000 Useable PLD gate QuickRAM ESP"*, Datasheet, http://www.quicklogic.com, 27th August 1998.

[12] Lucent Technologies, *"ORCA Series 3 Field Programmable Gate Array"*, Datasheet, Revision 1, http://www.lucent.com, September 1998.

[13] J. M. P. Cardoso and M. P. Véstias, *"Architectures and Compilers to Support Reconfigurable Computing"*, http://www.acm.org/crossroads/xrds5-3/rcconcept.html, 8th April 1999.

[14] I. Page, *"Reconfigurable processor architectures"*, Microprocessors & Microsystems, ISSN 0141-9331, volume 20, number 3, May 1996, pp. 185-96.

No particular application is well-supported by a conventional microprocessor which has a pre-determined set of functional units. This is particularly true in highly dynamic areas, such as multimedia, communications and other embedded systems. We suggest that additional silicon is used to provide hardware which can be dynamically configured to support any application. By combining a conventional microprocessor and FPGA reconfigurable logic on one chip, commodity pricing is maintained and yet the same part can effectively support a wide range of applications. A novel FPGA architecture is outlined which is particularly suitable for this style of implementation.

[15] Virtual Computer Corporation (VCC), *"Field Programmable Gate Arrays - An Enabling Technology"*, http://www.vcc.com/fpga.html.

[16] J. E. Vuillemin, P. Bertin, D. Roncin, M. Shand, H. H. Touati, P. Boucard, *"Programmable active memories: reconfigurable systems come of age"*, IEEE Transactions on Very Large Scale Integration

(VLSI) Systems, ISSN 1063-8210, volume 4, number 1, March 1996, pp. 56-69.

Programmable active memories (PAM) are a novel form of universal reconfigurable hardware coprocessor. Based on field-programmable gate array (FPGA) technology, a PAM is a virtual machine, controlled by a standard microprocessor, which can be dynamically and indefinitely reconfigured into a large number of application-specific circuits. PAM`s offer a new mixture of hardware performance and software versatility. We review the important architectural features of PAM`s, through the example of DECPeRLe-1, an experimental device built in 1992. PAM programming is presented, in contrast to classical gate-array and full custom circuit design. Our emphasis is on large, code-generated synchronous systems descriptions; no compromise is made with regard to the performance of the target circuits. We exhibit a dozen applications where PAM technology proves superior, both in performance and cost, to every other existing technology, including supercomputers, massively parallel machines, and conventional custom hardware. The fields covered include computer arithmetic, cryptography, error correction, image analysis, stereo vision, video compression, sound synthesis, neural networks, high-energy physics, thermodynamics, biology and astronomy. At comparable cost, the computing power virtually available in a PAM exceeds that of conventional processors by a factor 10 to 1000, depending on the specific application, in 1992. A technology shrink increases the performance gap between conventional processors and PAM`s. By Noyce`s law, we predict by how much the performance gap will widen with time.

[17] D. Conner, *"Reconfigurable logic. Hardware speed with software flexibility"*, Australian Electronics Engineering, ISSN 0004-9042, volume 29, number 8, August 1996, pp. 40, 42, 44.

If you view reconfigurable FPGAs only as static logic devices that accommodate design changes during prototyping and as occasional field upgrades, you are missing the latest frontier in digital design. The ability of infinitely reprogrammable FPGAs to dynamically redefine hardware provides performance improvements you can't afford to ignore. Reconfigurable logic is not so much a new product as a new wag of thinking. Although SRAM-based FPGAs have been available for years, new devices are offering greater performance. Depending on the device used, you can reprogram an SRAM-based FPGA from several times a second to more than a thousand times a second. Despite the device's flexibility, most designers use the reprogrammable capability as a development aid to debug and refine designs, and occasionally for field upgrades. In the context of this article, reconfigurable logic refers to infinitely-reprogrammable logic devices.

[18] Triscend, *"Triscend E5 Configurable Processor Family"*, Datasheet, http://www.triscend.com, 9th November 1998.

[19] Starbridge Systems Inc., http://www.starbridgesystems.com/Pages/about.html.

General information and examination of the company and their ideas for dynamically reconfigurable supercomputers.

[20] M. Butts, *"Future directions of dynamically reprogrammable systems"*, Proceedings of the IEEE 1995 Custom Integrated Circuits Conference, ISBN 0 7803 2584 2, IEEE, 1995, pp. 487-94.

FPGAs have made it possible to build dynamically reprogrammable systems. This paper reviews the current state of FPGA technology, and discusses the two main classes of dynamically reprogrammable systems: logic emulators and reconfigurable computers. Current practice and future directions for each are explored.

[21] S. Kumar, L. Pires, D. Pandalai, M. Vojta, J. Golusky, S. Wadi and H. Spaanenburg, *"Benchmarking Technology for Configurable Computing System"*, Proceedings of IEEE Symposium on FPGAs for Custom Computing Machines, ISBN 0 8186 8900 5, IEEE Computer Society, 1998.

[22] B. Radunović and V. Milutinović, *"A Survey of Reconfigurable Computing Architectures"*, Field-Programmable Logic and

Applications, Proceedings of FPL'98, ISBN 3 540 64948 4, Springer-Verlag, 1998, pp. 376-385.

[23] B. Fawcett, *"FPGAs as reconfigurable processing elements"*, IEEE Circuits & Devices Magazine, ISSN 8755-3996, volume 12, number 2, March 1996, pp. 8-10.

In most applications, FPGAs are used to implement "glue logic", providing the advantages of high integration levels without the expense and risk of custom ASIC development. However, as SRAM-based FPGA devices have increased in capability, their use as in-system-configurable computing elements is receiving considerable attention. Indeed, reconfigurable FPGA technology holds the potential for reshaping the future of computing by providing the capability to dynamically alter a computer's hardware resources to optimally service immediate computational needs. Computing circuits built from SRAM-based FPGAs can meet the true goal of parallel processing-executing algorithms in circuitry with the inherent parallelism of hardware, while avoiding the instruction fetch and load/store bottlenecks of traditional von Neumann architectures. There are many computationally-intensive algorithms that can benefit from being partially or wholly implemented in hardware. Typically, these algorithms are too specialised to justify the expense of manufacturing custom IC devices.

[24] K. Nagami, K. Oguri, T. Shiozawa, H. Ito and R. Konishi, *"Plastic Cell Architecture: Towards Reconfigurable Computing for General Purpose"*, Proceedings of IEEE Symposium on FPGAs for Custom Computing Machines, ISBN 0 8186 8900 5, IEEE Computer Society, 1998.

[25] J. A. B. Dines, *"Optoelectronic Computing: Interconnects, Architectures and a Systems Demonstrator"*, PhD Thesis, Heriot-Watt University, Riccarton, Edinburgh, EH14 4AS, UK, 1998.

[26] D. A. B. Miller, *“Optics for Low-Energy Communication Inside Digital Processors – Quantum Detectors, Sources and Modulators as Efficient Impedance Converters”*, Optics Letters, Vol. 14, pp. 146–148, 1989.

[27] P. Malinverni and A. Frochel (Editors), *“Technology Road Map: Optical Interconnects for Integrated Circuits”*, European Commission, ESPRIT programme – MEL-ARI OPTO, http://www.cordis.lu/esprit/src/melop-rm.htm, 1998.

[28] D. A. B. Miller, H. M. Ozaktas, *“Limit to the Bit-Rate Capacity of Electrical Interconnects from the Aspect Ratio of the System Architecture”*, Journal Of Parallel and Distributed Computing, Volume 41, No. 1, pp. 42–52, 1997.

[29] H. S. Stone, *“Parallel Processing with the Perfect Shuffle”*, IEEE Transactions on Computers, Volume C-20, No. 2, February 1971

[30] M. J. Quinn, *"Parallel Computing Theory and Practice"*, 1994 McGraw-Hill.

[31] J. Gourlay, T. Y. Yang, M. Ishikawa and A. C. Walker, *"Low-order Adaptive Optics for Free-Space Optoelectronic Interconnects"*, Submission to Applied Optics Special Issue on Optics in Computing, 1999.

[32] E. W. Williams, *"The CD-ROM and Optical Disk Recording Systems"*, Oxford University Press, 1994.

[33] C. J. Chang-Hasnain, M. and J. P. Harbison, L. T. Florez and C.Lin, *"Monolithic Multiple Wavelength Surface Emitting Lasers"*, Journal of Lightwave Technology 9 (12), pp. 1665-1672, 1991.

[34] D. A. B. Miller, D. S. Chemla, T. C. Damen, T. H. Wood, C. A. Burrus, A. C. Gossard and W. Wiegmann, *"Novel Hybrid Optically Bistable Switch: The Quantum-Well Self-Electro-Optic Effect Device"*, Optics Letters QE-21, 13-15, 1984.

[35] M. Makiuchi, H. Hamaguchi, T. Kumai, O. Aoki, Y. Oikawa and O. Wada, "GaInAs pin Photodiode/GaAs Preamplifier Photoreceiver for Gigabit-rate Communications Systems using Flip-Chip Bonding Techniques", Electronics Letters, volume 24, number 16, pp. 995-996, August 4th, 1988.

[36] A. Touhafi, W. F. Brissinck and E. F. Dirkx, *"Simulation of ATM switches using dynamically reconfigurable FPGA's"*, Field-Programmable Logic and Applications, Proceedings of FPL'98, ISBN 3 540 64948 4, Springer-Verlag, 1998, pp. 461-5.

We discuss a new approach for efficient simulation of queuing systems by using the configurable computing paradigm. One of the classical approaches to optimising the simulation of queuing systems is by implementing coprocessors for the hard parts of the simulation algorithm using ASIC or FPGA. The approach that we are proposing is different from the previous one in the sense that we are not using FPGA to optimise the simulation algorithm but we implement an equivalent system to the one that has to be simulated in the FPGA. As a case study we have made a simulator for ATM switching fabrics. Our research shows that dynamically reconfigurable FPGA such as the 6200 RPU of XILINX can be efficiently used for the simulation of queuing systems.

[37] S. AlKasabi and S. Hariri, *"A dynamically reconfigurable switch for high-speed networks"*, Conference Proceedings of the 1995 IEEE Fourteenth Annual International Phoenix Conference on Computers and Communications, ISBN 0 7803 2492 7, IEEE, 1995, pp. 508-14.

We present a design for a dynamically reconfigurable switch for high-speed networks. The proposed switch will be used to build high-performance parallel/distributed systems. Switch reconfigurability is exploited to implement different interconnection topologies and support different application requirements. We discuss an FPGA implementation of such a switch where a new configuration can be attained by reprogramming its routing component. The design supports both circuit and packet switching communications with wormhole routing. Our design employs virtual channel flow control to improve throughput and provide additional degree of freedom in routing packets to their destinations. We also show how to use the proposed switch design to build a high-speed multi-link ring network and how to embed other topologies, such as Hypercube, onto that ring network. The same approach can be adopted to embed other topologies.

[38] P. Popovic, *"Reconfigurable FPGA-based co-processor accelerates DSP"*, Electronic Product Design, volume 18, number 2, February 1997, pp. 23-4, 26.

The availability of large SRAM-based FPGAs is fuelling development of computing architectures using dynamically reconfigured hardware. The author describes one of the first devices to reach the market, and gives a graphics application example.

[39] T. H. Szymanski, M. Saint-Laurent, V. Tyan and A. Au, *"A Field Programmable Gate Array with Optical I/O"*, Technical Digest OSA Optics in Computing, Aspen, Colorado, ISBN 1557525846, page 149, April 1999.

[40] L. Selavo, S. P. Levitan and D. M. Chiarulli, *"An Optically Reconfigurable Programmable Gate Array"*, Technical Digest OSA Optics in Computing, Aspen, Colorado, ISBN 1557525846, page 146, April 1999.

[41] J. Mumbru, D. Psaltis, G. Zhou, X. An and F. Mok, *"Optically Programmable Gate Array (OPGA)"*, Technical Digest OSA Optics in Computing, Aspen, Colorado, ISBN 1557525846, page 153, April 1999.

[42] M. F. Sakr, S. P. Levitan, C. L. Giles and D. M. Chiarulli, *"Reconfigurable Processor Architectures Exploiting High Bandwidth Optical Channels"*, Proceedings of IEEE Symposium on FPGAs for Custom Computing Machines, ISBN 0 8186 8900 5, IEEE Computer Society, 1998.

[43] J. Gourlay, T. Yang, J. A. B. Dines, J. F. Snowdon and A. C. Walker, *“Development of Free-Space Digital Optics in Computing”*, Computer, 31, 2, pp. 38- 44, 1998.

[44] B. S. Wherrett, J. F. Snowdon, S. Bowman and A. Kashko, *“Digital Optical Circuits for 2-D data Processing”*, SPIE Proceedings on Optical Computing, vol. 1806, pp. 333-346 1993.

[45] S. G. Akl, *“Parallel Sorting Algorithms”*, Academic Press, 1985.

[46] R. P. Webb, A. J. Waddie, K. J. Symington, M. R. Taghizadeh and J. F. Snowdon*, "An Optoelectronic Neural Network Scheduler for Packet Switches"*, Submitted to Applied Optics, May 1999.

[47] M. R. Taghizadeh and J. Turunen, *"Synthetic Diffractive Elements for Optical Interconnection"*, Optical Computing and Processing, 2 (4), pp. 221-242, 1992.

[48] J. A. B. Dines, J. F. Snowdon, M. P. Y. Desmulliez, D. B. Barsky, A. V. Shafarenko and C. R. Jesshope, *"Optical interconnectivity in a scalable data-parallel system, Journal of Parallel and Distributed Computing"*, 41, 120-130, 1997.