Supporting Bulk Synchronous Parallelism with a High Bandwidth Optical Interconnect

I. Gourlay, P.M. Dew, K. Djemame
Informatics Research Institute, University of Leeds, Leeds LS2 9JT
{iain, dew, karim}@comp.leeds.ac.uk

J.F. Snowdon, G. Russell
Physics Department, Heriot-Watt University, Riccarton, Edinburgh EH14 4AS

Abstract

The list of applications requiring high performance computing resources is constantly growing. The cost of inter-processor communication is critical in determining the performance of massively parallel computing systems for many of these applications. This paper considers the feasibility of a commodity processor-based system which uses a free-space optical interconnect. A novel architecture, based on this technology, is presented. Analytical and simulation results based on an implementation of BSP (Bulk Synchronous Parallelism) are presented, indicating that a significant performance enhancement, over architectures using conventional interconnect technology, is possible.

Keywords: Parallel, BSP, optical interconnect, sorting.

1. Introduction

A fundamental aspect of parallel computing is scalability and efficiency. Recent focus has been on computational clusters built from low cost commodity components, e.g. Cox et.al. [2]. These systems offer high performance at low cost and are becoming commonplace within the academic community. For such systems, a critical factor in determining overall performance is the speed with which inter-processor communication occurs. This is particularly true for high-bandwidth applications such as data mining and real-time graphics. However, physical limits on electrical interconnects, as indicated by Miller [15] are likely to limit the communication performance of systems based on such technology [20]. Consequently, this paper considers the use of commodity processors, communicating via an optoelectronic interconnect, to build a highly parallel machine. This offers the potential to design an architecture with high-speed inter-processor communication, scalable up to large numbers of processors. It is shown that a potential bottleneck in the interface between electronic and optical communication can be overcome by careful consideration of architectural and algorithmic design, ensuring that the optical bandwidth is utilised effectively such that a significant performance enhancement is obtained.

Specifically the paper addresses the feasibility of a computational cluster with a high bandwidth Free-Space Optical Interconnect (FSOI). The architecture is based around a commodity PC cluster (the term PC is used in the paper to signify a commodity processor), with communication occurring via the FSOI as explained in section 2. An additional smart-pixel based layer is added, with the purpose of interfacing between the PCs and the optical interconnect. This layer must be utilised in such a way as to overcome the problem presented by the bandwidth bottleneck in the links to the PCs. It is envisaged that the smart-pixel layer will have relatively simple computational functionality, e.g. to support combining and reordering of messages used in the BSP computational model (see section 3). By ensuring that each PC has a unique link to the

smart-pixel layer, it is expected that, since the cost of communication between the smart-pixel and PC layers does not increase with the number of processors, this architecture will be scalable with respect to communication cost. The proposed architecture is shown schematically in figure 1.

The optoelectronic architecture is characterised by a number of key parameters, also shown on figure 1. The processor speed, *sp* is the time taken for a basic computational operation (i.e. 1 flop). Communication between the PC layer and the smart-pixel layer is characterised by the latency and bandwidth parameters, L_{PC} and B_{PC}, where L_{PC} is the minimum time taken to send a message between the smart-pixel and PC layer and is assumed to be independent of direction. B_{PC} is the bandwidth available in communication between these layers. Specifically, B_{PC} is the number of bits per second that can be communicated by a link between a PC and the smart-pixel layer. The assumption is made that each PC has two links, so that two-way communication between the PC and smart-pixel layers can occur simultaneously and independently. Finally $L_{optical}$ and $B_{optical}$ are the latency and bandwidth associated with communication between a pair of smart-pixel arrays.

The paper is organised as follows. Section 2 describes the characterisation of the optoelectronic interconnect performance, i.e. communication between smart-pixel arrays. The physical implementation of the interconnect is discussed and the values of $B_{optical}$ (as a function of the number of processors, *p*) and $L_{optical}$ are estimated. Section 3 introduces the Bulk Synchronous Parallel (BSP) computational model (see Skillicorn et.al. [20]), which forms the basis of the analysis. This is followed in section 4 by a description of a method of implementing BSP on the optoelectronic architecture shown in figure 1. Methods of approximating cost model parameters in terms of physical parameters are also described. In particular, the bandwidth and latency figures used to characterise the system (see below) are combined into a single effective bandwidth and latency of the system architecture (B_{eff} and $L_{optical}$ respectively). Since there are a large number of parameters required to characterise the system, they are summarised in table 1.

Implementation of the BSP model on the optoelectronic architecture is discussed, in section 5, in the context of a typical computing problem: integer sorting. Analytic results are assessed by comparison with those generated by discrete-event simulation. Section 6 addresses this issue and summarises the main results and conclusions of the paper and discusses future work.

2. Characterisation of the optoelectronic network

This section models the bandwidth and latency ($B_{optical}$ and $L_{optical}$) of the optoelectronic interconnect for a Completely Connected Network (CCN), implemented via a FSOI system. This is constructed using modulator and detector arrays linked by an image relay lensing system as discussed by Dines et.al. [5] and Gourlay et.al. in [7]. Typical performance parameters are available from experiments performed at Heriot-Watt University for the FSOI technology considered here.

The communication between any processor pair is single hop, such that the signal remains in the optical domain from source to destination. Signal beams are added and removed from the optical interconnect by controlling the polarisation. At each node the polarisation of the beam determines whether the beam is diverted out of the interconnect to the Smart-Pixel Array (SPA) or continues. This forms a single bus-based system. For scalability, multiple buses can be utilised in parallel as is currently done electronically. For simplicity and without lost of generality only the single bus-based system is considered here.

The communication between the smart-pixel and optical highway layers is characterised by the bandwidth B_{OH} (the optical highway bandwidth), B_{SPA} (the off-smart-pixel array bandwidth) and the latencies L_{OH} and L_{SPA}. These are used to derive $B_{optical}$ and $L_{optical}$ (see table 1). The characterisation of communication in this architecture is shown in figure 2.

The following terminology will be used to describe the communication link between SPAs:

1. Physical link – A real 1 bit wide channel consisting of a modulator, detector and connecting lens system.

2. Logical link – The data link as see by the communications system. This consists of a number of physical links and has the aggregate bandwidth of them. $B_{optical}$ is the bandwidth of a logical link.

3. Group – The transceivers on a particular SPA forming physical links traversing the same number of optical relays (i.e. the same distance). The analysis assumes wrap-around links at the ends so each group will consist of the transceivers for two logical links, one to the left and one to the right. Let *Ti* be the number of transceivers belonging to the *ith* group, on the SPA operating at physical link bandwidth, B_i, between transceivers *i* links apart. (Note that for a CCN there are $X = P/2$ groups for even *p* and $X = (P-1)/2$ for odd *p*).

The following assumptions have been made in order to determine an expression for $B_{optical}$:

A1. The underlying network operates well within the physical limitations of its design to ensure that there is room for signalling, fault tolerance and other essential controls.

A2. The number of transceivers available on a SPA chip is greater than p, the number of processors, and B_{OH} *is not near saturation.*

A3. More than one physical link per data link may be exploited so that each logical data link has the same bandwidth and there are as many physical links as required to maintain the integrity of the logical bandwidth [14].

The first assumption ensures that it is possible to manufacture the network. The second implies $B_{optical}$ is limited only by B_{SPA} and from assumption *A3*, $B_{optical} = B_1T_1 = B_2T_2 = ... = B_XT_X$ which leads to the expression,

$$B_{SPA} = 2\sum_{i=1}^{X} B_iT_i \tag{1}$$

where *X* is defined above. The bandwidth of a physical link within an optical highway of more than one stage is limited by the dissipation of optical power between the stages (i.e. detector/receiver limited) as indicated by Layet et.al. [14]. Further, the operating speed is linear in power dissipation, (e.g. see Hecht [9]) and thus depends primarily on the size and topology of the network. This gives,

$$B_i = \xi^i f_0 \tag{2}$$

where ξ is the efficiency of the optics between transceivers 1 link apart and f_0 is the nominal communicating frequency of an unattenuated transceiver pair. Since the operating range of analogue electronics is limited compared with the modulators and detectors available [6,22], the value of f_0 is dependent on the design of the analogue driver electronics. Typically, available driver electronics operate in the 500Mhz to 1GHz range according to Forbes et.al. [6].

On noting that $2\sum_{i=1}^{X} T_i \le N$, where *N* is the maximum number of transceiver pairs on a SPA it follows from equations (1) and (2) that for assumption *A1* to be true,

$$B_{optical} \le \frac{1}{2} f_0 N \left[\sum_{i=1}^{X} \frac{1}{\xi^i} \right]^{-1} = \frac{1}{2} f_o N \left[\frac{\xi^X (1-\xi)}{1-\xi^X} \right] \tag{3}$$

The second parameter required to evaluate the optoelectronic interconnect is the latency, $L_{optical}$ which is the minimum time required to send a message between a SPA pair. Let,

$$L_{optical} = L_{SPA} + L_{OH} \tag{4}$$

where L_{OH} denotes the time of flight along the optical highway, which can be estimated as,

$$L_{OH} = \frac{pq}{2c} \tag{5}$$

Here q is the length of a single optical link and c is the speed of light.

L_{SPA} comprises both the message combining/routing (see section 3), and the latency of the electronic-optical conversions. The time required for any computation on the SPA layer has not been fully investigated. However, the following approximation of $L_{optical}$ is sufficient for this paper. As the chips are based on Field Programmable Gate Array (FPGA) technology it is assumed the message combining and routing is at cache speeds (i.e. 3-10 ns)[10,18]. The latency of the electronic-optical conversions has been investigated by Dambre et.al. in [4] and has been found to be of the order of 5-10 ns using state-of-the-art components according to the Technology Roadmap [20], which is commensurate with experimental results obtained at Heriot-Watt [22]. This gives an estimated L_{SPA} value of 8-20 ns, resulting in a worst-case estimate of,

$$L_{optical} \approx \frac{pq}{2c} + 20 \quad (ns) \tag{6}$$

Table 1 provides typical values of current, state-of-the-art components and is used throughout the analysis. Substituting these into Eq. (3) and (6) gives the critical equations, which are used in later sections in the paper, i.e.

$$B_{optical} \le 8192 \left\lceil \frac{0.05 \times 0.95^{X}}{1 - 0.95^{X}} \right\rceil \quad (Gbit/s) \tag{7}$$

and

$$L_{optical} \approx 1.67\,p + 20 \quad (ns) \tag{8}$$

This allows for up to 235 processors at 1Gbit/s, with optical latency of ≈ 412 ns, in a single bus based system.

The assumed geometry of the system (linear chain of SPAs) is primarily responsible for the apparent non-scalability to large numbers of processors. In particular, the limit on the number of processors is due to the

number of physical links required becoming greater than the number of channels available on the SPA, i.e. 16384 in this example (see table 2). It is possible to support more processors by reducing the logical bandwidth required so that fewer physical channels are required for each link. For example, reducing the required bandwidth to 500MHz allows a single bus to support over 260 processors. If a higher bandwidth and more processors are required, additional buses can be added. This could result in a more scalable system at the cost of a slight increase in latency and more hardware. In a larger system a 2D or 3D layout of SPAs would be deployed (possibly with multiple buses per dimension) to provide the scalability up to several thousand processors.

3. The Bulk Synchronous Parallel (BSP) model

The concept of separating communication from computation and sending large amounts of data at once seems well suited to the optoelectronic architecture, since data packets can be combined into much larger messages before being sent to a processor. Consequently, the BSP computational model is taken as the basis for considering the potential of this architecture. This section briefly describes the BSP model and the associated cost model.

3.1. The BSP model

The BSP model is based on a parallel computer, consisting of a set of processors (each with local memory), a communication network that delivers messages directly between processors and a mechanism for efficient synchronisation of all or any subset of the processors. A computation on a BSP computer consists of a series of supersteps, each of which involves three phases: Firstly, the processors perform a local computation, i.e. each (or a subset of) the processors perform a computation, using data that is stored in their local memory. This is followed by a communication phase, where each processor sends data to other processors, to be received at the beginning of the next superstep. Finally, a barrier synchronisation takes place; all processors are guaranteed to have received data sent in the previous phase at the end of synchronisation. A virtue of the BSP model is in the communication phase. Since each processor combines all its messages destined for the same processor into a single message contention can be relieved by re-ordering messages prior to sending them thus avoiding hotspots. Empirical results obtained by Hill et.al.[11] have shown that combining and re-ordering prior to sending provides significant performance

enhancements over sending data as it is produced. More importantly, by combining messages prior to sending, message start-up is paid only once in communication between a given processor pair. A BSP computation is shown schematically in figure 3.

3.2. The BSP cost model

A significant advantage of the BSP model is the simplicity of the associated cost model. Additionally, it has been shown to be accurate for a wide range of computations [11]. The cost of a superstep is the sum of three terms, describing local computation, communication and barrier synchronisation. The overall cost is normalised so that the cost of a basic computational operation is 1. Hence, if there are a maximum of w operations on any processor in the local computation phase, then the cost associated with this is simply w. The communication cost is described in terms of the communication throughput ratio, g; the maximum number of messages sent or received by any processor, h; and the maximum size (number of words) in a message, m. Here (noting that an h-relation refers to a communication pattern where each processor sends or receives a maximum of h messages), g is the cost of communicating a 1-relation under continuous traffic conditions. The total communication cost for the superstep is then mgh. Finally, a cost l is associated with barrier synchronisation. Hence, the total cost of a BSP superstep is of the form, $w + mgh + l$. For an algorithm consisting of S supersteps, the total cost can be written as

$$C = \sum_i w_i + mg \sum_i h_i + Sl \tag{9}$$

Hence, the problem of estimating the cost of a particular algorithm (as long as values for the number of supersteps and values of h and w for each superstep can be specified) reduces to determining values for g and l (normalised in terms of the time taken to perform a basic operation).

The following section discusses the implementation of BSP on the optoelectronic architecture.

4. Implementing the BSP model on the optoelectronic architecture

In this section, an approach to implementing BSP on the optoelectronic architecture is described, that exploits the high optical bandwidth. A method for estimating the BSP parameter g (which characterises inter-processor communication) is presented. Since a significant number of parameters are being used to characterise the system, it would be useful to combine these into smaller, manageable units to simplify the analysis. Consequently the following model is used:

The three-layered optoelectronic system architecture (figure 4) is viewed as equivalent to an architecture A, consisting of a set of processors connected by an interconnect, with an effective bandwidth B_{eff} available in communication between any processor pair, under continuous traffic conditions. Similarly, A is characterised by an effective latency L_{eff}, the minimum cost to be paid in any inter-processor communication.

Section 4.1 discusses the chosen BSP implementation. Section 4.2 presents analytic methods for assessing the performance of the system architecture in the context of the model indicated above. These form the basis for some comparisons with a conventional cluster, presented in section 4.3.

4.1. A data streaming based implementation of the BSP model

In order to utilise the high bandwidth offered by the optoelectronic architecture, it is critical to consider the following two issues. Firstly, due to the multi-layered nature of the architecture, careful consideration of the BSP implementation is required, to ensure that the system does not suffer from prohibitively high latency. Secondly, it is desirable to collect data in the smart-pixel layer and combine them into large messages in order to exploit the optical bandwidth.

Based on the above considerations the following implementation of BSP is proposed. A PC and its corresponding smart-pixel array (and buffer) are viewed as a single BSP processor. Hence, communication between the PC layer and the smart-pixel layer can occur during local computations, while inter-smart-pixel array communication can only occur at the end of a superstep.

Recall that it is assumed that there are two channels linking each PC to the smart-pixel layer, so that bi-directional communication can occur simultaneously. A superstep is carried out as follows:

1. Data communicated in the previous superstep resides in the smart-pixel layer. A fraction of the data is sent to the PCs. The remaining data being sent to the PCs during the course of the local computation. The data initially sent to the PCs and the size of the initial fraction of data sent, is chosen so that a PC never completes its computation on the received data before sufficient additional data has arrived from the smart-pixel layer to allow the PC to continue the computation. This property must hold until all the data necessary to complete the local computation resides in the PC layer.
2. As data is produced, it is sent to the smart-pixel layer, to be communicated at the end of the superstep This occurs concurrently with step 1 and is carried out in such a way as to ensure that the fraction of

data received by the smart-pixel layer upon completion of the local computation is as close as possible to 1.

3. After completion of the local computation, the process of passing data (to be communicated to other processors) to the smart-pixel layer is completed, i.e. data still residing on the PCs is sent to the smart-pixel layer. Inter-BSP processor communication occurs, with data being passed from sending to receiving smart-pixel arrays, via the optical interconnect. Any necessary message combining and re-ordering occurs in the smart-pixel layer.
4. Barrier synchronisation occurs.

In order to help to quantify the effectiveness of steps 1 and 2 above, it is useful to introduce two parameters, *r* and *s*:

Definitions

For a given superstep, *r* is a lower bound on the fraction of data remaining in the smart-pixel layer associated with any PC after the initial communication between the smart-pixel and PC layers at the beginning of step 1 above.

For a given superstep, *s* is a lower bound on the fraction of data that resides in the smart-pixel layer for any PC, upon completion of the local computation at the end of step 2 above.

Note that the buffer storage capacity is assumed to be sufficiently large so that it does not overflow. The significance of the parameters *r* and *s* will become clearer in section 4.2.

4.2. Approximating the BSP communication cost: L_{eff} and B_{eff}

This section addresses the issue of estimating the parameter *g* (in terms of the effective bandwidth and latency) for the implementation of BSP on the optoelectronic architecture and expressions for L_{eff} and B_{eff} are obtained. However, before addressing the optoelectronic architecture, it is useful to consider a conventional cluster. This allows useful generalisations regarding the characterisation of systems implementing BSP. In particular, it clarifies the reasoning behind the introduction of the parameters L_{eff} and B_{eff}, in the analysis that follows.

Currently, one of the leaders in interconnect technology is Myrinet [24]. A cluster based on this technology can be viewed simplistically as a set of PCs, connected to a switch with a maximum bandwidth of

≈ 1.3Gbit/s. This is shown in figure 5. Two bandwidth parameters can be used to characterise this system, B_{clust} and B_p. Here, B_{clust} is the aggregate bandwidth supported by the switch, i.e. 2 Gbit/s in a given direction, while B_p is the bandwidth available in communication between any pair of PCs under continuous traffic conditions. For example, in a CCN $(p-1)$ PCs may be communicating data to the same PC, so that $B_p \approx 2\times10^9/(p-1)$bit/s. Thus, for a cluster consisting of a large number of processors, more than one switch is required.

In addition to bandwidth, the communication cost is influenced by the latency, L_p of the system. It is assumed here that this includes the cost associated with any message combining and re-ordering required in implementing BSP, in addition to all other sources of latency. Note that it is not practical to obtain an accurate figure for L_p analytically, since it is not simply the communication latency, rather it is the communication latency when implementing BSP. Nevertheless, using typical latency figures for current systems [24], L_p is expected to be $O(\mu s)$.

It is possible to obtain an analytic expression, allowing the BSP parameter g to be estimated in terms of the parameters described above. Recall that g is the cost of communicating a 1-relation under continuous traffic conditions. Notice that this assumes that it costs the same to send h m-word messages from a processor as it does to send one message with mh words. This is reasonably accurate for large message sizes, but for small messages start-up costs can dominate [19]. This can be incorporated into the model, so that the cost of sending an h-relation of m-sized (i.e. m words) messages is $mg(m)h$. Here (as indicated by Skillicorn et.al. in [19]), the communication throughput is given by,

$$g(m)=\left(\frac{n_{1/2}}{m}+1\right)g_\infty \qquad (10)$$

In the above expression, g_∞ is the asymptotic (optimal) communication throughput ratio, reached in the limit of large messages, and $n_{1/2}$ is the message size that would produce half the optimal throughput ratio. The cost, $T_{1-relation}(m)$, of sending a 1-relation of m-sized messages under continuous traffic conditions is given by,

$$T_{1-relation}(m) = L_p + \frac{m}{B_p} \tag{11}$$

Note that, in Eq. (11) the bandwidth must be expressed in terms of words/second (words/flop when normalised, as required in BSP). The same applies in the equations that follow. From (10) and (11),

$$(n_{1/2} + m)g_\infty = L_p + \frac{m}{B_p} \tag{12}$$

which leads to the expressions,

$$n_{1/2} = B_p L_p \tag{13}$$

and

$$g_\infty = \frac{1}{B_p} \tag{14}$$

Clearly the analysis is more complicated for the optoelectronic architecture, since it is a multi-layered system, with data streaming being used to manage communication between the layers of the architecture. However, equivalent expressions can be obtained, in terms of the effective bandwidth and latency (B_{eff} and L_{eff}).

In order to determine B_{eff} and L_{eff} , the above approach of estimating the cost of communicating a 1-relation (under continuous traffic conditions) can be taken. Subsequently, due to the use of data streaming and the consequent dependence of g on r and s, the parameter g now becomes a variable dependent on both software and hardware. Although this is a deviation from the usual BSP approach, where the cost parameters are determined by the hardware, it is necessary here in order to account for the effects of data streaming.

The cost of delivering a 1-relation (with message size m) is given by the sum of the costs associated with the following actions:

1. Communicating $(1-s)m$ words of data from the PC layer to the smart-pixel layer.
2. Preparing and sending a message of size m between smart-pixel arrays, via the optical interconnect.
3. Preparing and sending $(1-r)m$ words of data from the smart-pixel layer to a PC.

Message preparation time (including combining and re-ordering) can be incorporated into the optical latency, while latencies associated with communication between PCs and the smart-pixel layer are assumed to be independent of direction (i.e. PC to smart-pixel array and smart-pixel array to PC communication have the same latency). Similarly, assume B_{PC} is independent of direction. The cost of communicating a 1-relation can then be approximated by the following expression.

$$C_{1-relation} = \left(L_{optical} + \left(\sigma(r)+\sigma(s)\right)L_{PC}\right) + m\left(\frac{(2-r-s)}{B_{PC}} + \frac{1}{B_{optical}}\right) \quad \sigma(x) = \begin{cases} 0 \text{ if } x = 1 \\ 1 \text{ otherwise} \end{cases} \tag{15}$$

Here, the function $\sigma(x)$ expresses the fact that the cost L_{PC} is not paid unless data is passed between the PC and smart-pixel layers while no local computation is being performed on the PCs. Writing this (see Eq. (11)) as,

$$C_{1-relation} = L_{eff} + \frac{m}{B_{eff}} \tag{16}$$

leads to the expressions given in Eq. (17) and (18).

$$B_{eff} = \frac{1}{\frac{(2-s-r)}{B_{PC}} + \frac{1}{B_{optical}}} \tag{17}$$

Note that the effective bandwidth depends on *s* and *r*, and consequently varies from superstep to superstep in the course of a BSP algorithm. Consequently, *g* is algorithm dependent and varies from superstep to superstep within a given algorithm.

$$L_{eff} = \frac{L_{PC}\left(\sigma(r)+\sigma(s)\right)}{h} + L_{optical} \tag{18}$$

In Eq. (18), *h* is included because the cost associated with PC latency is paid only twice, at most, in communicating an *h*-relation (i.e. the term in $mg(m)h$ associated with PC latency is independent of *h*).

In analogy with Eq. (13) and (14),

$$n_{1/2} = B_{eff} L_{eff} \tag{19}$$

and

$$g_{\infty} = \frac{1}{B_{eff}} \tag{20}$$

Hence g can be expressed in terms of an effective bandwidth and latency, which reflect the effectiveness of data streaming for a given application. Critical in this respect (noting that it makes sense to concentrate on large message communication) is Eq. (20), which gives an estimate for the effective bandwidth and consequently g_∞ for the optoelectronic architecture. Upper and lower bounds on g_∞ can be obtained, which are not dependent on the application by considering the extreme cases (no data streaming and perfect data streaming) as discussed in section 4.3.

In addition to g, the barrier synchronisation parameter, l, also involves inter-processor communication. However, barrier synchronisation is not likely to dominate the cost of a well-designed BSP algorithm. Consequently, any performance enhancement as a result of faster barrier synchronisation is ignored and the value of l is assumed to be the same for the optoelectronic architecture as for a conventional cluster.

There follows a discussion and interpretation of the results obtained in this section, based on realistic parameter figures and comparison with a conventional cluster.

4.3. Interpretation of results

This section discusses the interpretation of the results obtained in section 4.2 based on realistic parameter figures and comparison with a conventional cluster. It may be useful in this respect to refer to table 1 in the introduction, which summarises the parameters used to characterise the optoelectronic system architecture. Initial indications of the expected performance and the relevance of data streaming can be obtained by comparing the parameters B_{eff} and L_{eff} with B_p and L_p respectively. Upper and lower bounds can be obtained for B_{eff} by considering the extreme cases, $r = s = 1$ (perfect data streaming) and $r = s = 0$ (no data streaming). This leads to,

$$\frac{1}{\left(\frac{2}{B_{PC}} + \frac{1}{B_{optical}}\right)} \leq B_{eff} \leq B_{optical} \tag{21}$$

Suppose the optoelectronic architecture is based around a 64-processor cluster. In this case, using (for example) $B_{PC} \approx 1\text{Gbit/s}$ and from Eq. (7), $B_{optical} \approx 98.4\text{Gbit/s}$. This leads to

$$500\text{Mbit/s} \leq B_{eff} \leq 98.4\text{Gbit/s} \tag{22}$$

Clearly this is a significant range, reflecting the importance of data streaming when $B_{optical} >> B_{PC}$. It can be concluded from these estimates that a significant performance enhancement can only be ensured if data streaming is effective. For a larger cluster of, for example, 200 processors a considerable performance enhancement may be obtained even without the use of data streaming, since while B_p decreases as p increases, B_{PC} is constant (each PC has a unique link to the smart-pixel layer).

The relationship between L_{eff} and L_p is strongly dependent on h. For large h, $L_{eff} \approx L_{optical}$ and in this case it is expected that the optoelectronic architecture will be low latency compared to a conventional cluster. Typical latency values for current systems are $O(\mu s)$ [24]. However if h=1, for example, then L_{eff} is approximately $2L_{PC}$. Note that L_p is just the sum of L_{PC} and the cost associated with message re-ordering and combining. Hence, in this case, the latency of the optoelectronic architecture is likely to exceed (or at least be comparable to) the latency in a conventional cluster. However, for $m >> B_{eff} L_{eff}$, the effects of latency can be neglected as bandwidth then dominates over latency in the communication cost. It is in this regime that the high bandwidth of the optical interconnect can be fully utilised.

5. Algorithmic case study of the optoelectronic architecture

The practicality of designing algorithms with effective data streaming is assessed in this section, by considering the system performance in a typical application: parallel integer sorting. The algorithm used is Parallel Sorting by Regular Sampling (PSRS), chosen because it is asymptotically optimal and is an appropriate algorithm for communicating large messages to utilise the optical bandwidth. Section 5.1 describes the algorithm and its implementation on the optoelectronic architecture, while section 5.2 considers performance issues, such as the conditions under which data streaming can be implemented effectively. Section 5.3 then builds on this by considering 64 and 128 processor examples, using the preceding analysis to determine the effectiveness of data streaming. Section 5.4 then presents results obtained using discrete-event simulation of the optoelectronic architecture and uses these results to assess the validity of the analysis.

5.1. Parallel sorting by regular sampling (PSRS)

In addressing the problem of parallel sorting of initially randomly ordered integers into ascending/descending order, PSRS (described by Tiskin in [21]) was chosen for two main reasons. Firstly, if the set of elements to be sorted is large, then the algorithm involves the communication of large messages between processors. Secondly, the coarse-grain nature of the local computations indicates that it may be possible to utilise data streaming between the PC and smart-pixel layers effectively.

Suppose there are n integer elements to be sorted and they are initially distributed across the processors, with each processor being assigned a sub-array of size n/p. The BSP algorithm proceeds as follows:

SUPERSTEP 1: Each processor sorts its sub-array using an efficient sequential sorting algorithm. Each processor then selects $(p+1)$ regularly spaced *primary samples* from the sorted sub-array, including the first and last elements. This splits each sub-array into p primary blocks, each of size n/p^2. Every processor then broadcasts its primary samples to all other processors, using a direct broadcast. Note that using the two-phase broadcast would not be advantageous, since all processors are broadcasting elements.

SUPERSTEP 2: Each processor performs the identical computation of sorting the $p(p+1)$ primary samples using an efficient sequential sorting algorithm and then selecting $(p+1)$ regularly spaced *secondary samples* from these, including the first and last elements. These split the n elements into p secondary blocks, each of size $O(n/p)$. The next stage is to collect elements that belong to a given secondary sample to one processor. Hence (labelling the processors $P_0,...,P_{p-1}$ and the secondary blocks $S_0,...,S_{p-1}$), processor P_i receives all elements belonging to secondary sample S_i. In order to achieve this, each processor identifies, for each primary block, which secondary block(s) it overlaps with and sends the entire block to that (those) processors at the end of the superstep.

SUPERSTEP 3: Each processor merges the received primary blocks, discarding elements that do not belong to its assigned secondary block.

Before considering the optoelectronic implementation, it is appropriate to consider the BSP complexity of this algorithm. Note that, at the beginning of the third superstep, no processor receives more than $3p$ primary blocks. To clarify this, note that a primary block can be one of the following three types:

a) A primary block is an inner block if all its elements belong to the secondary block.

b) A primary block is an outer block if none of its elements belong to the secondary block.

c) A primary block is a boundary block if some but not all of its elements belong to the secondary block.

Since there are only $(p+1)$ primary samples in a secondary block, there cannot be more than p inner blocks. In addition, since a boundary block must contain one or both secondary block boundaries, there cannot be more than two boundary blocks per sub-array (i.e. $2p$ boundary blocks in total). Consequently $3p$ is an upper bound on the number of primary blocks containing elements belonging to a given secondary block.

The total local computation cost is then given by,

$$C_{comp} \approx d\left(\frac{n}{p}\log\left(\frac{n}{p}\right)+ p(p+1)\log(p(p+1))\right)+\frac{3an}{2p}\log(3p) \tag{23}$$

where a is a constant (a is approximately 2 multiplied by the number of basic operations involved in a compare/store). The constant d is discussed in the following subsection.

The communication cost is dominated by the sending of primary blocks (to be merged) at the end of the second superstep and since there are three supersteps, there is also a barrier synchronisation cost of $3l$. Hence the total cost is,

$$C \approx d\left(\frac{n}{p}\log\left(\frac{n}{p}\right)+ p(p+1)\log(p(p+1))\right)+\frac{3an}{2p}\log(3p)+ g\left(\frac{n}{p^2}\right)\frac{3nW}{p}+3l \tag{24}$$

Here, W is the word size (number of bits) used to represent an integer. Clearly, if the optical bandwidth is to be utilised in the optoelectronic implementation, the effectiveness of data streaming is important. In order that data streaming has the best possible impact on performance, it is necessary to ensure that

a) All or most of the (sorted) sub-array elements reside in the smart-pixel layer, prior to sending primary blocks at the end of the second superstep.

b) The value of r in the final superstep must be close to 1, i.e. only a small fraction of the primary blocks are sent to the PC layer to begin merging.

Regarding point a), the sub-arrays should be sent to the smart-pixel layer, during the sorting in the first superstep, since this is the most costly local computation prior to the communication of primary blocks. In order to achieve this, it is necessary to use an efficient sequential algorithm that sorts sections of a sub-

array at a time, so that sorted sections of the sub-arrays can be sent to the smart-pixel layer while the remaining data is sorted. This can be achieved using quicksort, described by Kumar et. al. in [13]. The sub-array can be sorted such that, for some integer $y << x$, the smallest 2^y elements are sorted first, by always choosing a pivot from the smallest set of elements (where the sets are identified by the pivots) until the set of elements is of size $\approx 2^y$. Pivots are then chosen from the other subsets of the sub-array until the entire sub-array is split into sets containing $\approx 2^y$ elements. These are then sent to the smart-pixel layer while the next smallest 2^y elements are sorted, and so on. The integer *y* must be chosen to satisfy two conditions:

1. Sorting 2^y elements must take at least as long (on average) as sending 2^y elements from the PC layer to the smart-pixel layer.
2. The value of *y* must be small enough that sending the final 2^y elements from the PC layer to the smart-pixel layer.

In attempting to satisfy point b) above, the following approach is taken (on each BSP processor) in the third superstep:

Suppose there are *w* blocks to be merged, where $w \leq 3p$. Only $k << w$ of the primary blocks are sent to the PC. While these are merged, another *k* blocks are sent to the PC. The newly received blocks are then merged while another *k* blocks are sent to the PC, and so on until all the blocks have been received. The merging of blocks into a single secondary block is then completed. This imposes the requirement that the time taken to merge *k* blocks is at least equal to the time taken to send *k* blocks from the smart-pixel layer to the PC layer (see section 5.2). Note that data streaming may still be useful, even if the conditions described above are not satisfied. However, for purposes of analysis, these conditions give a good indication of the value (and limitations) of this approach.

5.2. Performance issues

In the implementation of PSRS described above, data streaming is used both in sending sub-arrays to the smart-pixel layer during local (sequential) sorting and in sending data to the PC layer during merging of primary blocks at the end of the algorithm. The practicality and usefulness is considered below.

As indicated in section 5.1, the communication cost is dominated by the cost of sending primary blocks (to be merged) at the end of the second superstep. For large *n*, this is characterised by the effective bandwidth associated with sending this data.

Theorem 1

The effective bandwidth in communicating the primary blocks, in order to merge them into secondary blocks, is given by

$$\frac{1}{B_{eff}} \cong \frac{\left(\frac{k'}{3p} + \frac{p2^{y'}}{3n}\right)}{B_{PC}} + \frac{1}{B_{optical}} \tag{25}$$

Proof outline: Consider first, the streaming of data from the PC layer to the smart-pixel layer during the local sorting in the first superstep. The cost of sorting 2^y elements is $d2^y \log(2^y)$ where *d* is a constant, while the time taken to send 2^y elements to the smart-pixel layer is $(L_{PC} + 2^y / B_{PC})$ if the bandwidth, B_{PC} is expressed in terms of words rather than bits (where it is assumed that each element requires one word of data). Hence the requirement that sorting 2^y elements takes at least as long as sending 2^y elements to the smart-pixel layer becomes,

$$d2^y y \geq \left(L_{PC} + \frac{W2^y}{B_{PC}}\right) \tag{26}$$

As before, *W* is the size of a word (i.e. number of bits) required to encode a single data element. This imposes a lower bound on *y* and consequently on the number of data elements that are initially sorted and sent to the smart-pixel layer. The result of this approach (assuming Eq. (26) is satisfied) is that, upon completion of the sequential sort in the first superstep, a fraction $(1 - 2^{y-x})$ of the sub-array elements already reside in the smart-pixel layer. More elements can be sent prior to the end of superstep 2, so that s_2, the value of *s* at the end of the second superstep, satisfies

$$(1 - 2^{y-x}) \leq s_2 \leq 1 \tag{27}$$

Clearly if $2^x >> 2^y$ then $s_2 \approx 1$. This requires, $\frac{n}{p} >> 2^y$.

In merging primary blocks in the final superstep, data streaming is used in sending data to the PC layer during merging. The requirement that the time taken to merge *k* blocks is at least equal to the time taken to send *k* blocks from the smart-pixel layer to the PC layer can be expressed as

$$\frac{ank}{p^2 2}\log(k) \geq \frac{Wkn}{p^2 B_{PC}} + L_{PC} \tag{28}$$

In order to achieve the desired $r \approx 1$, the condition, $\frac{k}{3p} << 1$, must be satisfied.

Note that Eq. (26) and (28) indicate the optimal (minimum) values for *y* (hence s_2) and *k* (hence r_3) respectively. Letting these values be y' and k' leads to theorem 1.

It is appropriate at this point to consider the implications of the critical equations that determine the practicality and effectiveness of this approach. This is addressed in the following sub-section, by considering examples with realistic parameter values, in order to assess the feasibility of data streaming and the likely performance enhancement obtained for this algorithm.

5.3. Examples and discussion

There follows an analysis, based on the results presented above for realistic examples. Two cases are considered here: 64 processors and 128 processors. Two parameters are of particular interest: the communication cost, obtained using Eq. (18) and Theorem 1, and the ratio C_{comp}/C_{comm}. This ratio is critical, since a substantial performance enhancement in communication is of little use if the computation cost dominates.

Typical current day PCs perform 1-3 Gflop/s. If the processor speed is 1Gflop/s and B_{PC} =1Gbit/s then $B_{PC} = 1$bit/flop. $L_{PC} = 10\mu s$ (current systems can have smaller latencies than this [24]) results in a normalised value of $L_{PC} = 10^4$ flops. It is assumed that *a* (the number of operations required to compare two elements and store one in a designated memory location multiplied by two) is 4. The constant *d* is 1.4 multiplied by the number of steps required to compare and exchange two data elements (assumed here to be 2). Hence $d = 2.8$. In each case, approximately the minimum possible values of *k* and *y* have been chosen, such that Eq. (26) and (28) can be satisfied. In those cases where Eq. (28) cannot be satisfied, the restriction

$\frac{k}{3p} << 1$ is dropped. If this does not suffice then data streaming is not used in receiving blocks in the final superstep.

Based on this approach values for the effective bandwidth can be obtained as a function of *n*. Note that these depend on word size, since the values of *y* and *k* (see section 5.2) depend on *W*. Figure 6 shows B_{eff} as a function of *n* for 32-bit words for both $p = 64$ and $p = 128$. Note that for large *n*, the effective bandwidth is close to 1bit/flop (corresponding to 1Gbit/s) in both cases. Noting that *this is the bandwidth available between any processor-pair under continuous traffic conditions*, this is a substantial improvement over a conventional cluster. For example, taking the (2+2) Gbit/s switch discussed in section 4.2, the effective bandwidth available if 64 processors were supported, is 0.031bits/flop. Even with more than one switch the effective bandwidth that can be dedicated to communication between a given processor pair is much less than that available in the case of the optoelectronic architecture. Consequently, a substantial performance enhancement can be anticipated as long as the communication cost is a substantial proportion of the overall cost. This issue is addressed in the following sub-section.

5.4. Simulation of the optoelectronic architecture

The objectives of the simulation experiments are twofold: 1) an observation of the behaviour of the system in terms of computation and communication phases, and 2) a comparison between the analytical and simulation results. The simulation objectives are achieved by considering both a 64-processor and a 128-processor machine and varying the size of a word (16, 32 bits) and the number of elements to sort $[10^6,...,10^7]$. It was found that for *n* significantly less than 10^6, agreement between simulation and analysis was poor due to the analytic approximations used. This does not present a problem, since for small *n*, the cost is dominated by the computation and it is not possible to exploit the optical bandwidth. Hence this regime is of little interest in the context of the work presented here.

The effect that varying these parameters has on computation and communication times are studied and compared to the results obtained analytically.

The simulation tool used in obtaining these results, PARSEC (PARallel Simulation Environment for Complex systems) [1], an extension of MAISIE, is a C-based discrete-event simulation language developed at UCLA Parallel Computing Laboratory. This tool adopts the process interaction approach to discrete-

event simulation. An object (also referred to as physical process) or set of objects in the physical system is represented by a logical process. Interactions among physical processes (events) are modelled by timestamped messages exchanged among the corresponding logical processes. The programs developed for simulating the optoelectronic architecture are executed using the traditional sequential simulation protocol (Global Event List).

The simulator randomly generates the elements to be sorted and performs the three supersteps in the algorithm described above. The simulation has significant storage requirements, with the consequence that results are only presented for up to 10^7 data elements to be sorted. Nevertheless, this is sufficient to make meaningful comparisons with the analytic results in order to validate them.

Figure 7 shows the communication cost and the ratio C_{comp}/C_{comm} for both the analytic and simulation cases, for 16-bit words, while figure 8 shows the same graphs for 32-bit words. Figures 9 and 10 show these graphs for 128 processors.

Several points are worth making regarding the extent to which the analytical and simulation results are expected to agree. Note that the analytic cost expressions are only approximate. Firstly, the cost of quicksort and merging used in the algorithm are probabilistic, hence the cost presented is only an average cost. Since the run-time is limited by the last processor to finish in each superstep, the average run-time for the quicksort and merging computations may be expected to be a little slower than assumed in the analysis. However, in approximating the merging cost, the worst-case was assumed. In view of the above points, it is expected that

a) The results simulation and computation results will agree fairly well.
b) The general trends, regarding computation and communication costs (and the ratio between them) as n increases will be the same in both cases.

Both these expectations are satisfied.

In all cases, the agreement between simulation and analysis for both the communication cost and C_{comp}/C_{comm} is excellent. Notice that the slight discrepancy between the analytic and simulation results for the computation cost is more pronounced when $p = 128$ than when $p = 64$, particularly for small n. This is due to the fact that the approximations used in estimating the costs of merging and sorting are less accurate for $p = 128$ than for $p = 64$. Specifically, the approximation used to estimate the cost of quicksort is more

accurate when the number of elements to be sorted is large. Specifically, the quicksort is used to sort n/p elements, hence the accuracy for a given n improves with decreasing p. A similar argument applies to merging, where the blocks to be merged are of size n/p^2.

Taking the results presented here, it can be concluded that this approach to implementing the parallel sorting ensures a performance enhancement as long is n is large, given that the communication cost constitutes a substantial proportion of the overall cost. For large n, the ratio C_{comp}/C_{comm} is approximately 2 for the 16-bit case and 1 for the 32-bit case.

6. Conclusions and future work

In this paper a novel parallel system architecture, based on a computational cluster, which makes use of a high bandwidth free-space optical interconnect has been presented and analysed. It has been shown, using an analytical approach that the optical bandwidth can be exploited to significantly improve inter-processor communication performance, taking parallel sorting as a case study. A middle layer, consisting of buffers and smart-pixel arrays with simple computational functionality, is used to manage the bandwidth mismatch between the optical interconnect and the PCs. Large messages are collected in the smart-pixel layer prior to inter-processor communication, using data streaming between the PC and smart-pixel layers, allowing the PC bandwidth bottleneck to be circumvented. By communicating data as a small number of large messages rather than a large number of small messages, the significance of latency is reduced and the optical bandwidth can be utilised. In particular, the PC to smart-pixel layer communication bottleneck can be partially overcome by use of data streaming and the effect of this bottleneck is also reduced by the inherent scalability of this system. This scalability is provided by the fact that the cost of communicating between the smart-pixel and PC layers is independent of the number of processors. Although the optical bandwidth drops off rapidly with p, according to the analysis presented in section 2, this is not a fundamental problem since the value of $B_{optical}$ can be made substantially larger by adding additional buses in 1 dimension or using a 2D or 3D layout. Work is currently underway to consider this case in detail.

More detailed simulation results, covering a wider range of cases are desirable, while it is also worthwhile considering ways of making more use of the computational functionality of the smart-pixel layer and investigating whether this can give a significant performance enhancement compared to more conventional

parallel computing platforms. Work is currently underway to address these issues by considering divide and conquer applications, where the smart-pixel layer is used to enhance the implementation of dynamic load balancing algorithms. On this basis, the optoelectronic architecture is being compared directly to alternative parallel architectures (a cluster, a network of clusters, etc.). Further simulations of the optoelectronic architecture are being carried out in this context.

Acknowledgements: This work was funded by the Engineering and Physical Sciences Research Council (EPSRC). The authors would like to thank Keith Symington for useful discussions relating to this research.

References

1. R. Bagrodia, PARSEC: PARallel Simulation Environment for Complex systems, 1998. http://pcl.cs.ucla.edu/projects/parsec/.
2. S.J. Cox, D.A. Nicole and K. Takeda, Commodity High Performance Computing at Commodity Prices, Proc. 21st World Ocean and Transputer User Group Technical Meeting, Canterbury, pp.19-26, 1998.
3. Cray website (http://www.cray.com).
4. J. Dambre, H. Van Marck, and J. Van Campenhout, Quantifying the impact of optical interconnect latency on the performance of optoelectronic FPGAs, 1999. (PI '99) Proceedings. The 6th International Conference on Parallel Interconnects, pp. 91 –97, 1999.
5. J.A.B. Dines, J.F. Snowdon, N. McArdle, Optical Highways for Computing Architectural and Topological Issues, Conference on Lasers and Electro-Optics Europe, 1998. CLEO/Europe. p. 195, 1998
6. M.G. Forbes and A.C. Walker, Wideband transconductance-transimpedance post-amplifier for large photoreceiver arrays, Electron. Lett, vol.34, no.6, pp. 589-590, 1998.
7. J. Gourlay, Tsung-Yi Yang, J.A.B. Dines, J.F. Snowdon and A.C. Walker, Development of free-space digital optics in computing, Computer, Volume: 31 Issue: 2 , Feb. 1998, Page(s): 38 –44.
8. T. Hauser, T.I. Mattox, R.P. LeBeau, H.G. Dietz, P.G. Huang, High-Cost CFD on a Low-Cost Cluster, SC2000, Dallas, Texas, USA, 2000.
9. E. Hecht, Optics, 3rd Edition, Addison-Wesley, 1998.
10. J.L. Hennessy, D.A. Patterson, Computer Architecture A Quantitative Approach, 2nd Edition, Morgan Kaufmann Publishers, Inc., 1996.
11. J.M.D. Hill, D.B. Skillicorn, Lessons Learned from Implementing BSP, 'High Performance Computing and Networks', Springer Lecture Notes in Computer Science, Vol. 1225, pp. 762-771, 1997.
12. International Technology Road Map for Semiconductors, 1999 (http://www.semichips.org).
13. V. Kumar, A. Grama, A. Gupta, G. Karypis, Introduction to Parallel Computing, Benjamin/Cummings Publishing Company (1994).
14. B. Layet, J.F. Snowdon, Comparison Of Two Approaches For Implementing Free-Space Optical Interconnection Networks, Optics Communications, Vol. 189, pp. 39-46, March 2001.
15. D. A. B. Miller, Physical Reasons for Optical Interconnection, Special Issue on Smart Pixels, Int'l J. Optoelectronics 11 (3), 155-168 (1997).
16. J.M. Nash, P.M. Dew, M.E. Dyer, J.R. Davy, Parallel Algorithm Design on the WPRAM model, 'Abstract Machine Models for Highly Parallel Computers' Oxford University Press, pp. 83-100, 1995.
17. D.T. Neilson, S.M. Prince, D.A. Baillie and F.A.P. Tooley. Optical design of a 1024-channel free-space sorting demonstrator. Applied Optics, Vol. 36, No. 35, 10 December 1997.
18. Rambus website (http://www.rambus.com).
19. D.B. Skillicorn, J.M.D. Hill, W.D. McColl, Questions and Answers about BSP, Scientific Programming, 6 (3), pp. 249-274, 1997.

20. Technology roadmap, Optoelectronic interconnects for integrated circuits, European Commission, Esprit programme Long Term Research, Microelectronics advanced research initiative MEL-ARI OPTO, June 1998.
21. A. Tiskin, The Design and Analysis of Bulk-Synchronous Parallel Algorithms, PhD thesis, University of Oxford, 1998.
22. A.C. Walker, M.P.Y. Desmulliez, M.G. Forbes, S.J. Fancey, G.S. Buller, M.R. Taghizadeh, J.A.B. Dines, C.R. Stanley, G. Pennelli, A.R. Boyd, P. Horan, D. Byrne, J. Hegarty, S. Eitel, H.-P. Gauggel, K.-H. Gulden, A. Gauthier, P. Benabes, J.-L. Gutzwiller and M. Goetz, Design and Construction Of An Optoelectronic Crossbar Switch Containing A Terabit Per Second Free-Space Optical Interconnect, IEEE Journal of Selected Topics in Quantum Electronics, Vol.5, No.2, 236-249, March/ April 1999.
23. T.L.Worchesky, K.J. Ritter, R. Martin, B. Lane, Large arrays of spatial light modulators hybridised to silicon integrated optics, Appl. Optics 35, pp. 1180-1186, 1996.
24. http://tldp.org/HOWTO/Parallel-Processing-HOWTO.html.