A SCALABLE AND PIPELINED EMBEDDED SIGNAL PROCESSING SYSTEM USING OPTICAL HYPERCUBE INTERCONNECTS*

HÅKAN FORSBERG[1)], MAGNUS JONSSON[2)], AND BERTIL SVENSSON[1,2)]

[1)] Department of Computer Engineering, Chalmers University of Technology, Göteborg, Sweden

[2)] Computing and Communication Laboratory, Halmstad University, Halmstad, Sweden

ABSTRACT

In this paper, we propose a system suitable for embedded signal processing with extreme performance demands. The system consists of several computational modules that work independently and send data simultaneously in order to achieve high throughput. Each computational module is composed of multiple processors connected in a hypercube topology to meet scalability and high bisection bandwidth requirements. Free-space optical interconnects and planar packaging technology make it possible to transform the hypercubes into planes and to take advantage of many optical properties. For instance, optical fan-out reduces hardware cost. This, altogether, makes the system capable of meeting high performance demands in, e.g., massively parallel signal processing. An example system with eleven computational modules and an overall peak performance greater than 2.8 TFLOPS is presented. The effective inter-module bandwidth in this configuration is 1,024 Gbit/s.

Keywords: Parallel computing systems; embedded signal processing; optical interconnects; hypercube topology

1 INTRODUCTION

Algorithms recently proposed for applications in embedded signal-processing (ESP) systems, e.g. in radar and sonar systems, demand sustained performance in the range of 1 GFLOPS to 50 TFLOPS [1]. As a consequence, many processing elements (PEs) must work together, and thus the interconnect bandwidth will increase [2]. Other requirements that typically must be fulfilled in ESP-systems are real-time processing, small physical size, and multimode operation. To be able to cope with all these constraints at the same time, new parallel computer architectures are required.

Several such parallel and distributed computer systems for embedded real-time applications have been proposed in the literature, including those systems making use of fiber-optics in the interconnection network to achieve high bandwidth. See, for instance, Jonsson [3,15].

Actually, by introducing optical technologies in ESP-systems, many hard-hitting requirements can be met. For example, physical size can be reduced and the bandwidth over the cross section that divides a network into two halves, usually referred to as bisection bandwidth, can be improved [4]. High bisection bandwidth (BB) reduces the time it takes to redistribute data between computational units that process information in different dimensions, and this property is of high importance in ESP-systems [4].

However, to make the best use of optics in interprocessing computing systems, all optical and optoelectronic properties must be taken into consideration. These properties include transmission in all spatial dimensions, light coherence, and high fan-out etc. [5].

In fact, it has been shown that optical free-space interconnected 3D-systems (systems using all three spatial dimensions for communication), with globally and regularly interconnected nodes, arrayed on planes, are best suited for parallel computer architectures using optics [6-9]. Folding optically connected 3D-systems into planes will also offer precise alignment, mechanical

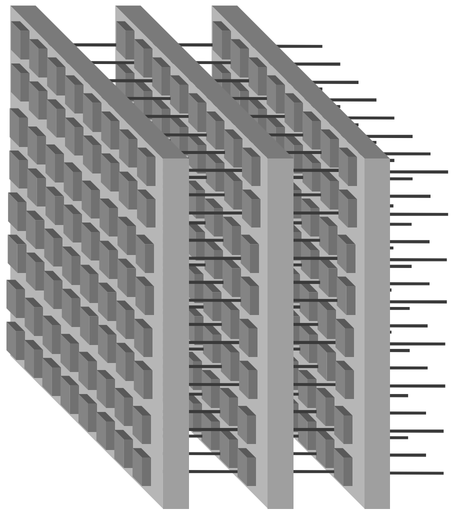

Figure 1: Three 6D-hypercubes, transformed into planes and massively interconnected – an example of the proposed system.

Patent Pending # 0004015-4, Swedish Patent and Registration Office robustness, and temperature stability at a relatively low cost [10].

In this paper, we propose a system consisting of several clusters of free-space optically interconnected processing elements. The clusters are linked together in a pipelined fashion to attain high throughput and to meet the nature of multimode signal processing applications in ESP-systems. The processing elements within a cluster are globally and regularly connected in a hypercube topology and, finally, transformed into a plane. Optical properties like transmission in all spatial dimensions, light coherence, high bandwidth, and fan-out etc., are utilized.

Figure 1 depicts an example of the proposed system. Note, however, that the intra-cluster hypercube formation is not shown here but later in the article.

2 HYPERCUBES

The hypercube is a topology that has been investigated extensively in the literature. One reason for its popularity is that many other well-known topologies like lower-dimensional meshes, butterflies, and shuffle-exchange networks can be embedded into the hypercube structure [12]. Another reason is that this topology can be used to implement several algorithms requiring all-to-all communication, e.g. matrix transposition, vector reduction, and sorting [13].

Geometrically, a hypercube can be defined recursively as follows: The zero-dimensional hypercube is a single processor. An n-dimensional hypercube with $N = 2^n$ PEs is built of two hypercubes with 2^{n-1} PEs, where all PEs in one half are connected to the corresponding PEs in the other half. See Figure 2. In this figure, a 6-dimensional hypercube is shown (Figure 2c). This hypercube is built of two 5D-hypercubes, which in turn are built of 4D-hypercubes (Figure 2b). The 4D-hypercube is further subdivided into 3D-hypercubes (Figure 2a). Note that the thick lines in Figure 2c consist of eight interconnections each.

A disadvantage of the hypercube is its complexity. It requires more and longer wires than the mesh, since not only the nearest neighbors are connected to each other if the dimension is greater than three, i.e. more dimensions than physical space [13]. The fact is that the required amount of electrical wires (of different length) in a relatively small hypercube will be enormous. Consider, for instance, an implementation of a 6D-hypercube on a circuit board, where the transfer rate of a unidirectional link between two processing elements must be in the order of 10 Gbit/s. This implementation requires 12,288 electrical wires, of different length, each clocked with a frequency of 312.5 MHz (32-bit wide links assumed). Since the wires are not allowed to cross each other physically, many layers will be required.

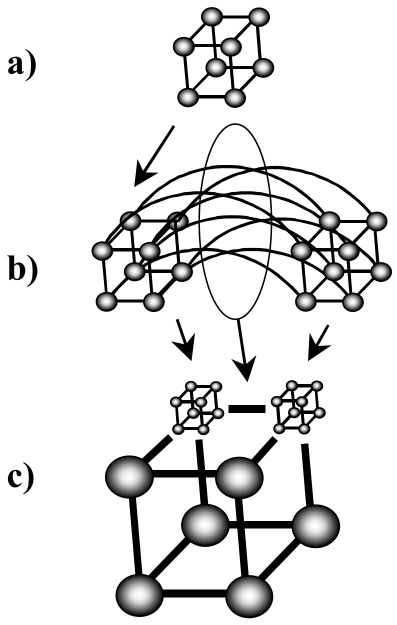

Figure 2: a) 3D-hypercube. b) 4D-hypercube built of two 3D-hypercubes. c) 6D-hypercube built of two 5D-hypercubes, which in turn are built of 4D-hypercubes.

2.1 Corner turning in hypercubes

In the introduction it was stated that interconnection networks in ESP-systems must be able to efficiently redistribute data between computational units that process information in different dimensions. In Figure 3, this reorganization process is shown. Here, the first cluster of processing elements, left cube, computes data in one dimension (marked with an arrow). Next working unit, right cube, computes data in another dimension, and thus redistribution must be performed.

This redistribution of data, referred to as corner turning in the literature, accounts for almost all of the interprocessor communication in ESP-systems [4]. Note also that corner turning requires all-to-all communication.

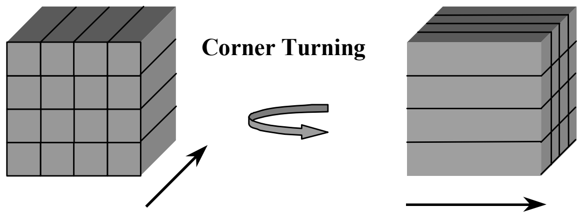

Figure 3: Redistribution of data between processing elements, computing in different dimensions.

How about corner turning in hypercubes? A corner turn is actually, from a mathematical point of view, a matrix transposition [4]. Therefore, as stated above, algorithms exist for this interconnection topology. Also, since the BB scales linearly with the number of processors in hypercubes, higher dimensions lead to very high BB.

In general, it seems that the hypercube is a exceedingly good topology for ESP-systems. The only disadvantage being its interconnection complexity. However, by using optical properties in free-space optically interconnected 3D-systems folded into planes, the interconnect complexity can be greatly reduced.

2.2 Hypercubes in planar waveguides

There are several reasons to fold optically connected 3D-systems into planes, including those already mentioned in the introduction. One reason is the ability to cool, test, and repair the circuits in a simple way [10,11].

In optical planar technology, waveguides are made of glass or transparent semiconductor based substrates. These substrates serve as a light travelling medium as well as a motherboard for surface mounted optoelectronic and electronic chips [11]. Also micro-optical elements, such as beamsplitters and microlenses, can be attached on both the top and the bottom side of the substrate. To be able to enclose the optical beams in the light travelling medium, the surfaces are covered with a refractive structure. The beams will, hence, "bounce" on the surface.

In the following six steps, 1-6, and Figures, 4-9, we show how a 6D-hypercube topology is merged into a planar waveguide. Before we start with step one, it must be elucidated that; since we are going to merge a 6D-hypercube into a plane, it is natural to imagine that three of the topology dimensions are transformed into one physical direction on the substrate, here called horizontal direction. The other three topological dimensions are thus transformed into the other physical direction, here called vertical. Further, since the hypercube is symmetric, everything that is valid in one direction is automatically valid in the other direction.

- **Step 1: Transmitters in horizontal direction.**

  In a 6D-hypercube, each processing element has six neighbors. Physically, this corresponds to three horizontal and three vertical neighbors. In Figure 4, it is shown, both topologically and physically, how one PE, colored black, sends data to its three horizontal neighbors.

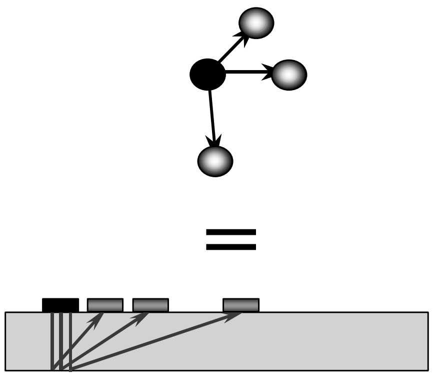

Figure 4: Topological and physical cross-section view of four PE. The black colored PE sends data to its three horizontal neighbors.

- **Step 2: Receivers in horizontal direction.**

  In the same way, a PE must be able to receive data from its three horizontal neighbors, see Figure 5.

Using diffractive elements, it is easy to create beamsplitters, i.e., to use the optical fan-out property. If we use beamsplitters here, we can actually reduce the number of horizontal transmitters with a factor of three, and thus reduce hardware cost, without destroying the hypercube topology. See Figure 6a. Note, however, that we must use some kind of channel time-sharing, when different data must be sent to all three neighbors at the same time, since only a single transmitter is available.

More important, since beamsplitting is not limited to one direction, we can actually reduce the number of transmitters with a factor of six (provided that we are using 6D-hypercubes). But, most important, if we use the

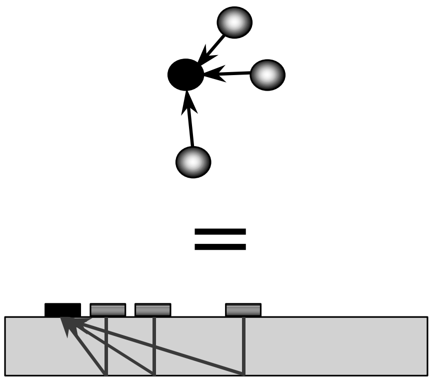

Figure 5: Topological and physical cross-section view of four PE. The black colored PE receives data from its three horizontal neighbors.

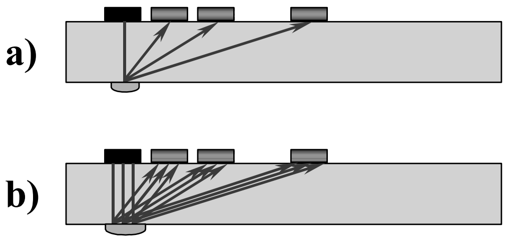

Figure 6: a) Beamsplitters are used to reduce the number of horizontal sending transmitters with a factor of three. b) Beamsplitters are used to increase the flexibility and multicast capacity in the network, at the expense of more receivers, but without additional transmitters.

hypercube transpose algorithm described by Foster [13], to perform corner turning, we will not lose any performance at all, even if we reduce the number of transmitters with a factor of six, compared to a system without beamsplitters. The trick in this algorithm is that data is only exchanged in one dimension at a time. Note, however, that the hypercube transpose algorithm sends $(\log P)/2$ times more data and $P/\log P$ fewer messages, in total, compared to a simple transposition algorithm also described in [13] (P is the number of processing elements). Therefore, the hypercube transpose algorithm is preferable when transfer costs are low and message startups are expensive [13]. As a result, optical interconnects with its slightly higher startup cost and high bandwidth typically matches the transpose algorithm behavior better than pure electrical wires. (See [13] for a more detailed description of communication costs concerning the transpose algorithms discussed here.)

Beamsplitters can also be used to create an advanced hypercube topology with more capacity than the original, at the expense of more receivers, but without additional transmitters, see Figure 6b. Last but not least, other hybrid topologies can be created with beamsplitters, but these will not be considered in this article.

- **Step 3: Reduction of transmitters.**

  As explained above, we will not lose any performance at all, even if we reduce the number of transmitters with a factor of six, when we perform corner turning with the hypercube transpose algorithm described in [13]. Therefore, we will use the beamsplitters depicted in Figure 6a above. Note, however, that we split up the light beam in both horizontal and vertical directions, and thus reduce the number of transmitters maximally.

In the same way as we investigated optical fan-out, we can analyze optical fan-in. In Figure 5 for instance, it is fully possible to use a single receiver for all beams. In that case, we must synchronize all processing elements in the

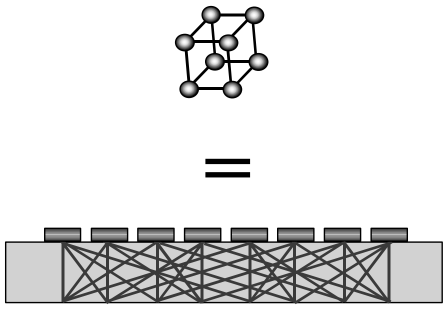

Figure 7: All transmitters and receivers in a horizontal row forms a 3D-hypercube.

hypercube; to be able to use some kind of time division multiple access, and thus avoid data collisions. With planar packaging technology, a synchronization clock channel is relatively easy to implement. Jahns [10] has, for instance, described a 1-to-64-signal distribution suitable for, e.g., clock sharing, with planar technology. However, since optical fan-in complicates our hypercubes with synchronization channels, we will leave this issue outside this paper.

- **Step 4: 3D-hypercubes.**

  In Figure 7, all transmitters and receivers in one row have been added. This corresponds topologically to a 3D-hypercube.

- **Step 5: 4D-hypercubes.**

  To realize hypercubes with higher dimensions than three, we must use the vertical direction. In Figure 8, a 4D-hypercube is shown both topologically and physically. The fourth dimension makes use of vertical space.

- **Step 6: 6D-hypercubes.**

  A 6D-hypercube makes full use of both horizontal and vertical space, see Figure 9. This physical layout corresponds to a full computational unit.

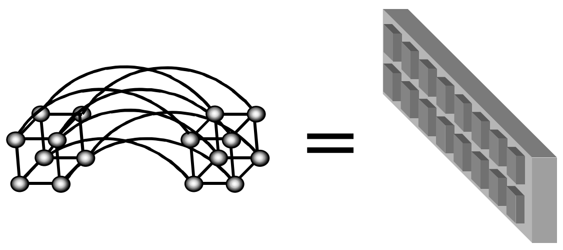

Figure 8: First extension in vertical direction - two 3D-hypercubes form a 4D-hypercube.

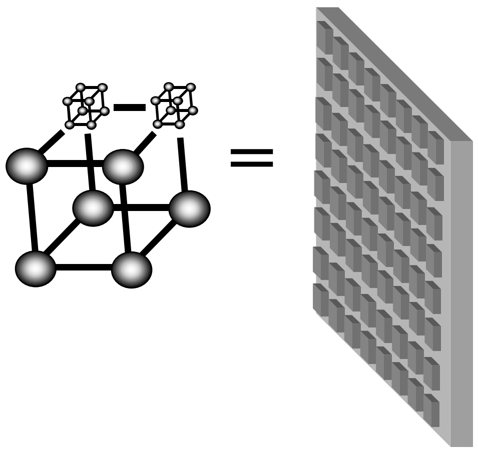

Figure 9: The whole computational unit – a 6D-hypercube.

3 SYSTEMS OF HYPERCUBES

3.1 Pipelined systems

If it is impossible to obtain the required performance with one computational unit, several computational units must work in co-operation. Also, since almost all applications in ESP-systems can be divided into computational parts that only need to send data forward to the next stage in a chain, it is natural to connect the computational units in a pipelined manner, as shown in Figure 1.

In Figure 1, all PEs in the leftmost plane can send data to the middle plane. But a single PE in the leftmost plane can only send data to an equivalent PE in the middle

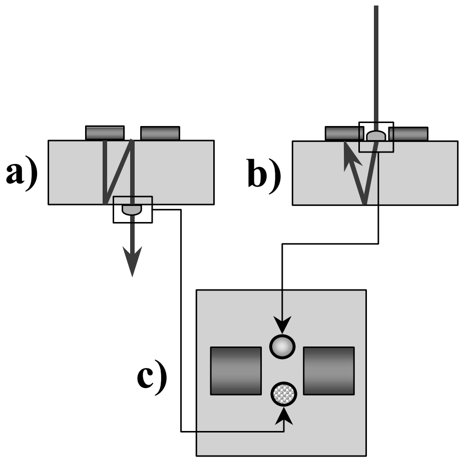

Figure 10: Lenses (or holes) needed to connect different computational units. a) Bottom lens, used for transmitter light to flow out to next unit. b) Top lens, used for collimating lights into the substrate from previous unit. c) Top view.

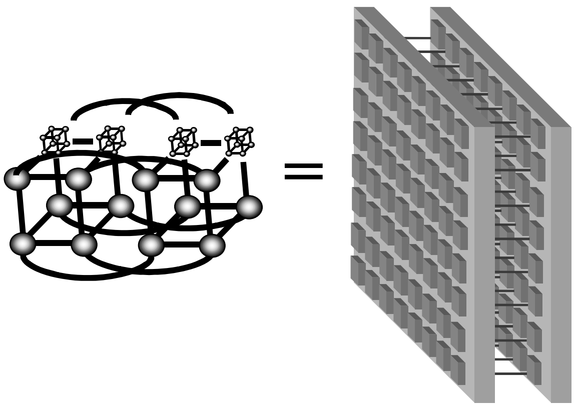

Figure 11: Topological and physical view over a 7D-hypercube.

plane. In the same way, the middle plane sends data but only to the rightmost plane.

In order to make the inter-module communication work, we have to open up the substrates, i.e., to let the light-beams propagate in via a lens from a previous unit, and also out to the next unit. In addition, we have to add diffractive elements, to steer the incoming beams inside the substrate, to be able to reach the right PE. In Figure 10, the lenses needed to connect different computational units are shown. Specifically, Figure 10a shows the bottom surface lens, while Figure 10b shows the top surface lens. Figure 10c shows the top view. Note that the bottom lens in this figure is shown through the substrate.

3.2 7D-hypercube – two planar arrays

In the previous section it was shown how to open up the substrate in order to connect computational units. However, this procedure only showed how the units were connected in one direction, i.e. one module could only receive from previous and send data to the next unit. By allowing communication in both directions, i.e., letting a module be able to send and receive data both forward and backward, a 7D-hypercube is actually formed by two planar arrays, see Figure 11. If more than two planes form an extended computational unit, the pure hypercube topology will not be preserved since only adjacent planes can communicate with each other. This, however, is not a limitation in many signal processing systems, due to the pipelined nature of the data flow.

3.3 Software scalability

If only one mode of operation is needed in the system, we can create a streamed architecture for that purpose. However, since it is very important for many ESP-applications, including airborne radars, to change mode of operation on the same system as needed in the application, we would like to have an architecture capable of multimode operations. Thus, different clusters of

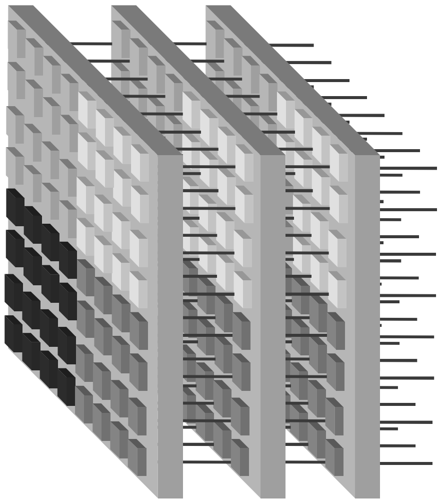

Figure 12: Four independently working chains of 4D-hypercubes. Each chain is marked with its own color.

computational units must be capable of working together in different ways.

The pipelined system described in this paper has very good potential for mapping of different algorithms in various ways. In fact, the system can be partitioned in all three spatial dimensions. An example of this is shown in Figure 12. In this picture, four different algorithms are mapped at the same time, on four smaller systems of pipelined 4D-hypercubes. It is also possible to create 5D-hypercubes inside each of these four smaller systems by connecting two 4D-hypercubes in different planes.

3.4 Hardware scalability

To be able to increase system performance in the future, hardware scalability is of great importance. In the proposed system, higher performance can be achieved by:

a) adding more planar arrays in the chain,
b) enlarging physical size of the planes, or
c) adding more PEs within a plane, i.e. increasing the hypercube dimension, by either b) or denser packaging.

Note that the inter-module links are free-space optical interconnects and that all modules are identical; this facilitates the adding of more planes. However, special attention must be paid to how the modules are stacked onto each other; e.g. heat removal etc. must be taken into account.

4 CASE STUDY

The upper part of Figure 13 shows a typical signal processing chain in future radar systems. The chain consists of six algorithmic pipeline stages, namely, beam forming, pulse compression, doppler filtering, envelope detection, constant false alarm ratio, and extraction [14]. The lower part shows the corresponding mapping of these stages on eleven 6D-hypercubes. Note, however, that eight of these hypercubes are extended to four 7D-hypercubes, and thus two-way inter-module communication is assumed. Further, each PE is assumed to have a floating-point peak capacity of 4 GFLOPS (based on the hypothesis that each PE is capable of performing 4 floating-point instructions per clock cycle at a frequency

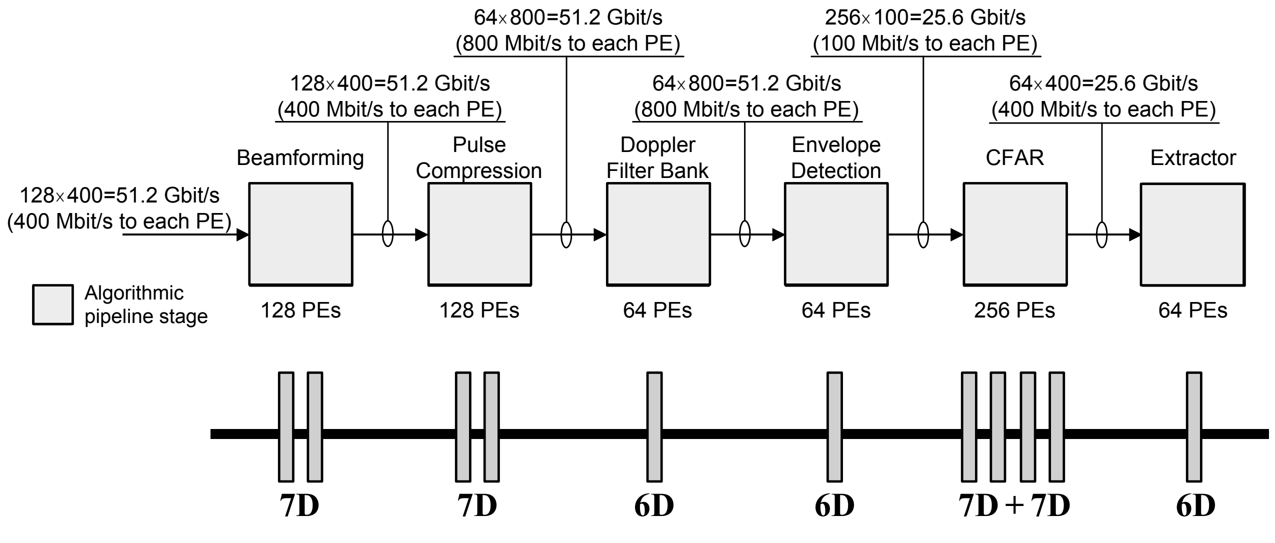

Figure 13: A typical signal processing chain in future radar systems. The upper part of the figure shows the maximum required data flow between the algorithmic pipeline stages. The lower part shows the mapping of these stages on a pipelined system of eleven planar packaged 6D-hypercubes.

of 1 GHz). A 6D-hypercube consisting of 64 PEs will, therefore, sum up to 256 GFLOPS, and the whole system of 11 hypercubes (704 PEs) sums up to more than 2.8 TFLOPS.

In the rest of this section, we will focus on communication aspects in the chain above, and especially intra-cluster corner turning. Also, some mapping aspects will be pointed out. Other details of the chain will be left out. The following system parameters are assumed:

$$\tau = 10\ ms$$
$$R_{link,eff} = 8\ Gbit/s$$

where τ is the coherent processing interval (CPI), i.e. the time a computational unit is allowed to process a chunk of data, before a new chunk is arriving. Note that if CPI is 10 ms, 100 chunks of data are processed every second. $R_{link,eff}$ is the efficient transfer rate of a single link in one direction, when overhead is excluded, e.g. message startup time.

To start with, the maximum required transfer rate between two pipeline stages in Figure 13 is 51.2 Gbit/s. This is only 10% of the maximum aggregated transfer rate of 64 links between two 6D-hypercubes in one direction. (64 links, each with a transfer rate of $R_{link,eff}$, sum up to 512 Gbit/s.) Consequently, the effective two-way inter-module bandwidth is 1,024 Gbit/s. Therefore, data flow between stages where no redistribution of data is needed is not a problem. The computational unit simply sends recently calculated data, at the same time while performing new calculations. The redistribution of data, however, must be done on a 6D- or 7D-hypercube, after a complete calculation of a chunk of data, or just before the computation in the following stage. This will consequently steal working time in either the stage before the corner turn or in the subsequent stage; unless the PEs are capable of performing calculations on one datacube at the same time as they perform a corner turn on another datacube. Note, however, that this procedure requires more memory and that the total delay in the algorithmic pipeline chain will increase.

In Figure 13, two corner turns must be performed, the first between the pulse compression and doppler filter bank stages, and the second between the envelope detection and the CFAR.

When we perform a corner turn, we will use the hypercube transpose algorithm mentioned in Section 2.2 since this makes it possible to reduce hardware cost. In this algorithm, half of the chunk of data is exchanged in every dimension in the hypercube. Therefore, a full corner turn takes:

$$\frac{\frac{1}{2} D_{size} \log_2(P)}{BB(P)} \tag{1}$$

seconds. D_{size} is the size of the chunk of data in Gbit, P is the number of processors in the hypercube, i.e. $\log_2(P)$ is the hypercube dimension, and $BB(P)$ is the bisection bandwidth ((Gbit/s), calculated in both directions), in the P-processor hypercube. Further, D_{size} and $BB(P)$ are:

$$D_{size} = \tau R_{agg} \tag{2}$$

$$BB(P) = P R_{link,eff} \tag{3}$$

where R_{agg} is the required transfer rate (Gbit/s) between the algorithmic pipeline stages (as shown in Figure 13), where redistribution of data takes place. If we substitute Equation 2 and 3 in Equation 1, we will have a new expression for the time it takes to perform a full corner turn:

$$\frac{\frac{1}{2} \tau R_{agg} \log_2(P)}{P R_{link,eff}} \tag{4}$$

As a result, if we decide to perform the first corner turn in the pulse compression stage with 128 PEs, it will take 1.75 milliseconds, according to Equation 4. If we, on the other hand, decide to perform it in the doppler filtering stage with 64 PEs, it will take 3 ms. This corresponds to a significant part (30%) of the coherent processing time. Therefore, we better carry out the first corner turn in the pulse compression stage. Similarly, the second corner turn will take 1.5 ms in the envelope detection stage, and 0.875 ms in the first part of CFAR. Here, the choice of where to perform the corner turn is not so critical.

5 CONCLUSIONS

In this paper, we have presented a powerful system suitable for embedded signal processing. Several computational units, which are capable of working independently and sending data simultaneously, are massively interconnected to meet high throughput demands. The hypercube topology forms the interconnection network within a computational unit. Free-space optical interconnects and planar packaging technology make it possible to transform these multi-dimensional hypercubes into planar waveguides. Beamsplitters can reduce the number of transmitters and thus also the hardware cost. If appropriate algorithms are used in certain all-to-all communication patterns, the reduction of transmitters will not reduce the performance. This is of special importance when reorganization of data is performed between working units that process information in different dimensions. Also worth mentioning is the system scalability in all spatial dimensions.

6 ACKNOWLEDGMENT

This work is financed by Ericsson Microwave Systems within the Intelligent Antenna Systems Project.

REFERENCES

[1] W. Liu and V. K. Prasanna, "Utilizing the power of high performance computing", *IEEE Signal Processing Magazine*, vol. 15, no. 5, Sept. 1998, pp. 85-100.

[2] G. W. Stimson, *Introduction to Airborne Radar*, 2nd Edition, SciTech Publishing, Inc., Mendham, NJ, USA, 1998.

[3] M. Jonsson, *High Performance Fiber-Optic Interconnection Networks for Real-Time Computing Systems*, Doctoral Thesis, Department of Computer Engineering, Chalmers University of Technology, Göteborg, Sweden, Nov. 1999, ISBN 91-7197-852-6. Thesis available at: *http://www.hh.se/staff/magnusj/*

[4] K. Teitelbaum, "Crossbar tree networks for embedded signal processing applications", *Proceedings of Massively Parallel Processing using Optical Interconnections*, MPPOI'98, Las Vegas, NV, USA, June 15-17, 1998, pp. 200-207.

[5] H. Forsberg, "Opportunities for parallel computer architectures using optical interconnects", Technical Report, No. 99-9, Department of Computer Engineering, Chalmers University of Technology, Göteborg, Sweden, June 1999. Report available at: *http://www.ce.chalmers.se/staff/rapid/*

[6] H. M. Ozaktas, "Towards an optimal foundation architecture for optoelectronic computing", *Proceedings of Massively Parallel Processing using Optical Interconnections*, MPPOI'96, Maui, HI, USA, Oct. 27-29, 1996, pp. 8-15.

[7] H. M. Ozaktas, "Fundamentals of optical interconnections – a review", *Proceedings of Massively Parallel Processing using Optical Interconnections*, MPPOI'97, Montreal, Canada, June 22-24, 1997, pp. 184-189.

[8] H. M. Ozaktas, "Toward an optimal foundation architecture for optoelectronic computing. Part I. Regularly interconnected device planes", *Applied Optics*, vol. 36, no. 23, Aug. 10, 1997, pp. 5682-5696.

[9] H. M. Ozaktas, " Toward an optimal foundation architecture for optoelectronic computing. Part II. Physical construction and application platforms", *Applied Optics*, vol. 36, no. 23, Aug. 10, 1997, pp. 5697-5705.

[10] J. Jahns, "Planar packaging of free-space optical interconnections", *Proceedings of the IEEE*, vol. 82, no. 11, Nov. 1994, pp. 1623-1631.

[11] J. Jahns, "Integrated free-space optical interconnects for chip-to-chip communications", *Proceedings of Massively Parallel Processing using Optical Interconnections*, MPPOI'98, Las Vegas, NV, USA, June 15-17, 1998, pp. 20-23.

[12] D. E. Culler, J.P. Singh, with A. Gupta, *Parallel Computer Architecture: A Hardware/Software Approach*, Morgan Kaufmann Publishers, Inc., San Francisco, CA, USA, 1999.

[13] I. Foster, *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering*, Addison Wesley Publishing Company, Inc., Reading, MA, USA, 1995.

[14] M. Jonsson, A. Åhlander, M. Taveniku, and B. Svensson, "Time-deterministic WDM star network for massively parallel computing in radar systems", *Proceedings of Massively Parallel Processing using Optical Interconnections*, MPPOI'96, Maui, HI, USA, Oct. 27-29, 1996, pp. 85-93.

[15] M. Jonsson, "Fiber-optic interconnection networks for signal processing applications" *4th International Workshop on Embedded HPC Systems and Applications (EHPC'99) held in conjunction with the 13th International Parallel Processing Symposium & 10th Symposium on Parallel and Distributed Processing (IPPS/SPDP '99)*, San Juan, Puerto Rico, Apr. 16, 1999. Published in *Lecture Notes in Computer Scienc*e. vol. 1586, Springer Verlag, 1999, pp. 1374-1385, ISBN 3-540-65831-9.