# System on Fabrics Architecture Using Distributed Computing

Partheepan Kandaswamy, *Member, IEEE*, James A. Flint, *Senior Member, IEEE*, and Vassilios A. Chouliaras

***Abstract*—This paper describes a novel, distributed sensor network with parallel processing capability based on the instruction systolic array (ISA). A new computing paradigm is introduced where spatially distributed sensors are closely coupled to the processing elements, and the whole array forms a parallel computer. This may find applications in wearable devices for sensing the position and other metrics of a human body and rapidly processing that data. A new programming model to implement the distributed computer on fabrics is described. The fabric-based distributed computing concept has been validated using a number of parallel applications, including a real-time shape sensing and reconstruction application. The exemplar wearable system based on the ISA concept has been realized using off-the-shelf microcontrollers and sensors. Results show that the application executes on the prototype ISA implementation in real time, thus confirming the viability of the proposed architecture for fabric-resident computing devices.**

***Index Terms*—Instruction systolic array, sensor system networks, distributed sensor networks, sensor array, system on fabrics, shape reconstruction, wearable sensing devices, wearable system, fabric-resident computing device.**

## I. Introduction

IN TODAYS technological era, wearable electronics have become increasingly important in our day to day lives [1]. At the same time there have been substantial increases in the capabilites of microprocessors in terms of their processing throughput and their significantly reduced real term cost [2]. Since the processing elements themselves are low cost, it opens up the possibility to include multiple processor elements into a single design. Furthermore, in the sensors field, it opens up the possibility for each sensor to have its own microprocessor, which is a known paradigm. However the present paper takes this a stage further and proposes that as well as individual sensors posessing their own processing element, they also have the capability to participate as an integral part of a parallel computer. This computer, as well as being distributed in terms of procesing capability is also physically distributed as the sensors are required to be by the sensing task in hand. One example of such a task might be in an instrumented garment with mutiple sensors measuring the position or other parameters of a human body.

Manuscript received March 1, 2018; revised May 2, 2018; accepted May 19, 2018. Date of publication June 6, 2018; date of current version June 26, 2018. The associate editor coordinating the review of this paper and approving it for publication was Dr. Rosario Morello. *(Corresponding author: Partheepan Kandaswamy.)*

The authors are with the Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough LE11 3TU, U.K. (e-mail: p.kandaswamy@lboro.ac.uk).

Digital Object Identifier 10.1109/JSEN.2018.2842692

Several authors have considered the general concept of decentralised processing in sensor systems [3]. An advantage of decentralisation is the increased robustness of the system in question. In some distributed networks, nodes can also be configured in a flexible way and added or removed as required [4]. Processing the data in a distributed manner shows potential for avoiding the fusion of a multitude of sensor data at once, and adding units would be cost-effective since it would mostly involve duplicating the basic design [5]. Inspired by perception in biological systems, distribution of a large number of simple sensing devices is gaining more support in detection applications [5]. A focus on fusion of sensor signals instead of strong analysis algorithms, and a scheme to distribute sensors, results in new issues. This is especially true in wearable computing, where sensor data continuously changes, and clothing provides an ideal supporting structure for simple sensors [6]. The present paper considers somewhat more powerful but still relatively simple sensors which have processing capability that allows them to communicate and participate actively in parallel tasks with other near-neighbors.

Electronic Textiles (e-textiles) are fabrics that feature electronics and interconnections woven into them, presenting physical flexibility and typical size that cannot be achieved with other existing electronic manufacturing techniques [7]. Components and interconnections are intrinsic to the fabric and thus are less visible and not susceptible of becoming tangled or snagged by surrounding objects. E-textiles can also more easily adapt to fast changes in the computational and sensing requirements of any specific application, this one representing a useful feature for power management and context awareness [7]. The vision behind wearable computing foresees future electronic systems to be an integral part of our everyday outfits. Such electronic devices have to meet special requirements concerning wearability. Wearable systems will be characterized by their ability to automatically recognize the activity and the behavioural status of their own user as well as of the situation around them, and to use this information to adjust the systems' configuration and functionality [8].

Miniaturization of microelectronics through system architecture and progress in the research of new materials has enabled the integration of computing capabilities into clothing [9]. The vision behind wearable computing, including their sensor system, is to make electronic systems an integral part of everyday clothing in the future delivering services such as intelligent personal assistants and medical monitoring devices. Whilst multi-core processors are the norm in desktop,

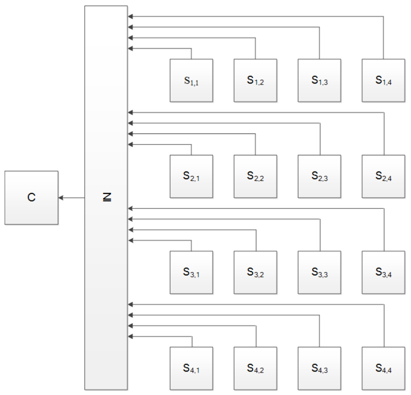

C
IN
$S_{1,1}$ $S_{1,2}$ $S_{1,3}$ $S_{1,4}$
$S_{2,1}$ $S_{2,2}$ $S_{2,3}$ $S_{2,4}$
$S_{3,1}$ $S_{3,2}$ $S_{3,3}$ $S_{3,4}$
$S_{4,1}$ $S_{4,2}$ $S_{4,3}$ $S_{4,4}$

Fig. 1. Single compute unit where IN shows an interconnection such as bus.

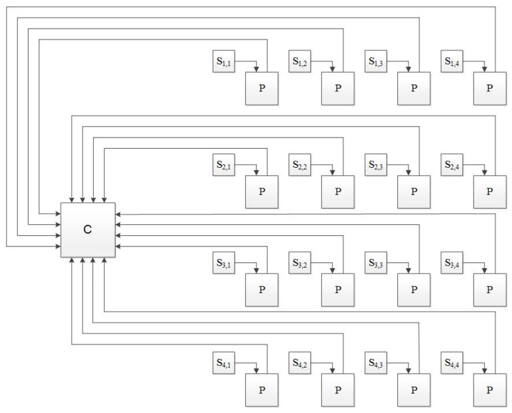

C
P
$S_{1,1}$ $S_{1,2}$ $S_{1,3}$ $S_{1,4}$
$S_{2,1}$ $S_{2,2}$ $S_{2,3}$ $S_{2,4}$
$S_{3,1}$ $S_{3,2}$ $S_{3,3}$ $S_{3,4}$
$S_{4,1}$ $S_{4,2}$ $S_{4,3}$ $S_{4,4}$

Fig. 2. Single compute unit with distributed processing elements and sensors.

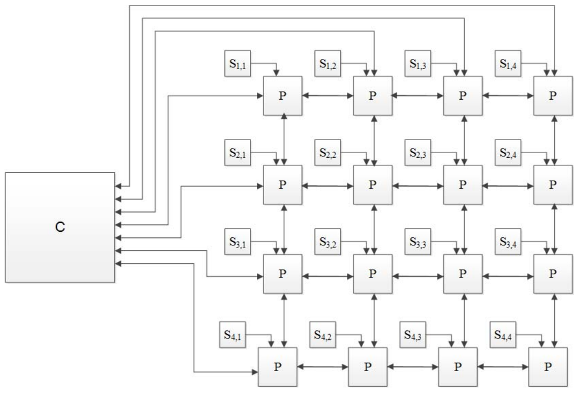

C
P
$S_{1,1}$ $S_{1,2}$ $S_{1,3}$ $S_{1,4}$
$S_{2,1}$ $S_{2,2}$ $S_{2,3}$ $S_{2,4}$
$S_{3,1}$ $S_{3,2}$ $S_{3,3}$ $S_{3,4}$
$S_{4,1}$ $S_{4,2}$ $S_{4,3}$ $S_{4,4}$

Fig. 3. Chosen architecture showing communication between processing elements.

embedded and mobile computing, it is noted that substantial further research and development is required to deploy such multicore processors on wearable technology and enable new applications. This paper addresses this particular vacuum in the current state-of-the-art by introducing a new wearable computing platform based on the concept of Instruction Systolic Array (ISA) and demonstrates its suitability with a shape-sensing application executing on prototype hardware consisting of an array of discrete microcontrollers and sensors.

## II. Multi-Sensor, Multi-Processor Wearable System Architecture

There are a number of possible mutiprocessor architectures that could be applied human body position sensing or a similar wearable sensing task. Consider a rectangular matrix of $N \times M$ sensors, each capable of capturing analogue data with an upper-frequency $f$ with a requirement to continuously process data, producing a result. The application is assumed to require processed data from all sensors in the sensor matrix.

Considering the architecture in Fig.1 it can be seen that a single control unit, C, needs to process samples at a rate of $2.N.M.f$ to satisfy the Nyquist criterion. The architecture relies on a single control unit, using a shared bus architecture which will result in further implementation overheads. This is nevertheless preferable as it reduces the number of physical interconnections that would be needed between the sensors and the central processor.

The architecture of Fig.2 introduces distributed processing with all such processing elements connected to the control unit and the sensors attached to the processing elements using dedicated local buses. Here the processing elements are required to process samples at $2f$ samples/second and after preprocessing data can be passed to the control unit. However, this offers limited advantage if the purpose is to process data which involves fusing information from adjacent sensors.

Finally, the architecture of Fig.3 is based on distributed processing elements with each such element physically connected to its neighbors and sensors, the latter via dedicated buses. The processing can be carried out locally at each processing element with the whole network of processing elements and sensors viewed as a distributed computer. This architecture is considered promising compared to other three architectures and is the subject of this paper. It is worth emphasizing that this is different to a conventional parallel architecture, which will be discussed later, because the processing elements are physically spaced out to coincide with their local sensors. Indeed it is possible for the processing elements and sensors to be manufactured as one single integrated circuit (IC) with each such IC connected by bus wires. Bus wire connection would then be facilitated using conductive thread or printed conductive wires on the fabric.

Further benefits of the chosen architecture are the locality of data and sensors and its suitability for implementation using the Instruction Systolic Array principles discussed in the next section.

## III. Systolic Architecture

The systolic mode of parallel processing, originally introduced by Kung and Leiserson [10] in 1978, rapidly gained research interest due to its ability to handle computationally demanding algorithms. This section briefly discusses some of the fundamental theory behind systolic arrays to set the present paper in context. The authors note that research into systolic arrays has been dormant for some years and at the time writing there was no known prior work using these arrays in physically distributed wearable systems. The Instruction Systolic Array (ISA) as an architectural concept was first introduced in [11]. The key properties of the ISA are local

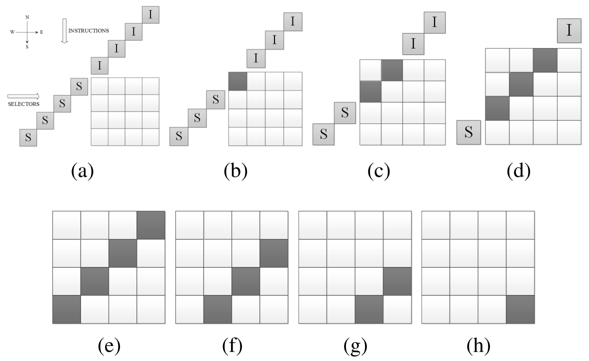


Fig. 4. Execution of an ISA where I indicates an instruction and S indicates a selector bit.

communication for data and control flow, modularity and scalability, local data handling and logical mapping.

In ISA, rather than data (which is more conventional), instruction opcodes and selector bits are pumped in a systolic fashion through the processor array. One instruction after the other is entered into the upper left processors and propagated in diagonal wavefronts through the array. While traveling through the array, each instruction is executed by each processor provided that execution is not inhibited. This masking mechanism is realized by selector bits that belong to each instruction and which flow in a horizontal fashion from left to right. An instruction opcode at a particular processing element is enabled by its selector bit which must be 1 otherwise, a no-operation instruction is executed.

Fig.4 shows how instructions together with selector bits are moved through a $4 \times 4$ array. The instruction stream and selector bits streams are combined (joined) as shown in Fig.4 allowing different algorithms to run on the same processor array. The fundamental model of a parallel computer can be seen as a mesh array of identical processors. The processors are capable of executing instructions from a small instruction set. Instructions, as well as selector bits, are used for controlling processing elements.

Continuous flow of instructions top-to-bottom (north to south) and selectors from left-to-right (west to east) are shown in Fig.4. Shaded boxes indicate the merging of instructions and selectors in the particular processing element and hence execution.

## IV. Novel Concept for On-Fabric Parallel Processing

Here a modified architecture is presented which integrates sensors into the parallel computer. The processing elements are arranged in a systolic manner as shown in Fig.5. Each processing element is connected to its neighbouring processing elements and also closely coupled to the sensors. Due to a large number of physical interconnections that are necessary to wire a suitable-sized sensor array, a compromise has to be taken by selecting the Inter-Integrated Circuit bus (I2C) standard and sharing it for both the ISA inter-element connections and sensors. Each processing element has four I2C buses, one per direction. The north and west ports act as slaves

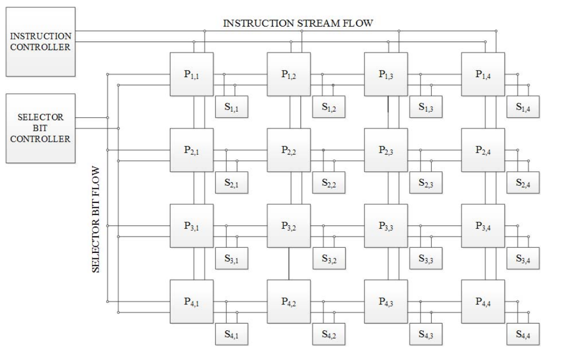


Fig. 5. System architecture-processor array (where P indicates a processing element, and S a sensor).

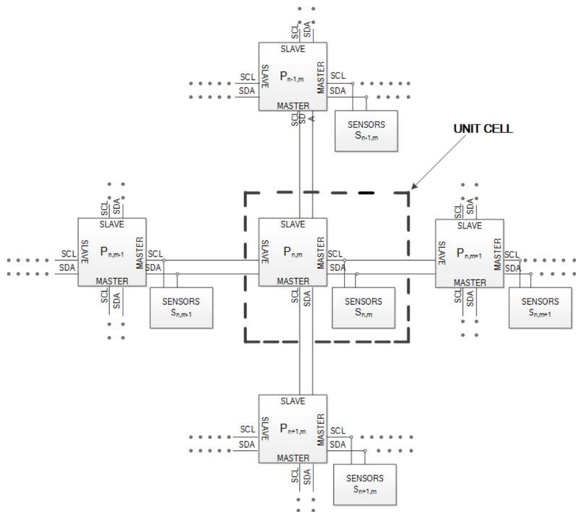


Fig. 6. Detail of I2C bus connections between processing elements.

and the east and south ports are the masters. The northern boundary of the array is connected to the instruction stream flow controller (the external host) which stores the array of instructions that need to be passed to the processing elements. The western boundary of the array is connected to selector bit flow controller (also the external host) which stores the array of selector bits that need to be passed to the processing elements. The sensors are closely coupled to the processing elements as shown in Fig.6 and the combined host-resident instruction stream and selector bits constitute the target application to execute on the array.

## V. The Implemented System

The ISA was implemented using 32-bit ARM microcontrollers of type Cortex-M0+ LPC824. A primary consideration of the selection was the availablity of four I2C bus interfaces. A total of 16 processing elements configured as a $4 \times 4$ array was used, along with selector bit and instruction controllers which were of the same microcontroller type. Each of the microcontrollers in the prototype array was powered and programmed with common firmware by an individual USB lead running to one of a pair of USB hubs. The selector and instruction controllers were programmed to feed the inputs of the array. A flowchart of the ISA operation is given in Fig.7.

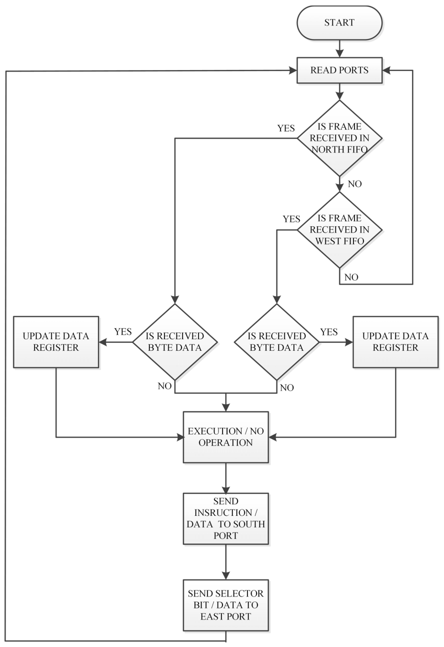
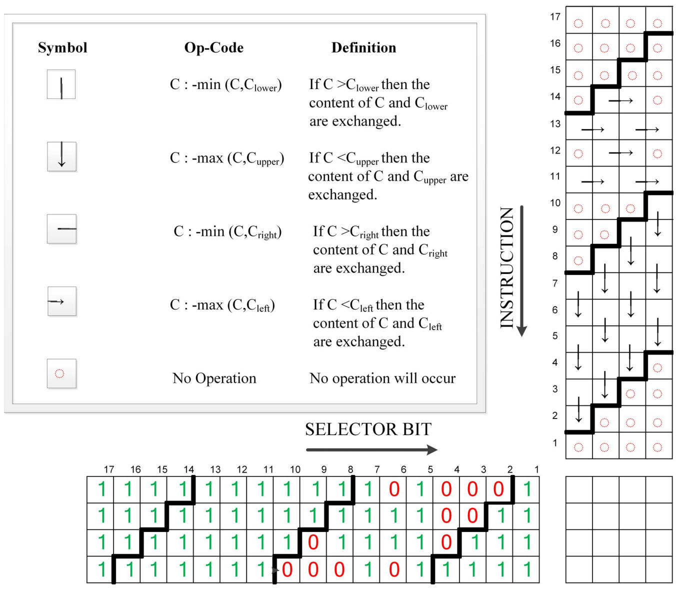


Fig. 7. Working of the ISA program.



| Symbol | Op-Code | Definition |
|---|---|---|
| │ | C : -min (C,$C_{lower}$) | If C >$C_{lower}$ then the content of C and $C_{lower}$ are exchanged. |
| ↓ | C : -max (C,$C_{upper}$) | If C <$C_{upper}$ then the content of C and $C_{upper}$ are exchanged. |
| ─ | C : -min (C,$C_{right}$) | If C >$C_{right}$ then the content of C and $C_{right}$ are exchanged. |
| → | C : -max (C,$C_{left}$) | If C <$C_{left}$ then the content of C and $C_{left}$ are exchanged. |
| ○ | No Operation | No operation will occur |

Fig. 8. ISA program for merge algorithm.

The instruction and selector bits are passed to the processing element; the latter is in listening mode waiting for the frame bytes to be received from the north and west slave I2C ports. Once both the instruction and selector bit are received they are then decoded and executed through an interrupt service routine. After execution the instruction and the selector bits are then forwarded to the neighbors through the south and east master I2C ports. Parallel programming of the machine as a whole is challenging as it requires careful design and there is very limited support from high level languages [12]. For this reason programs are presented in this paper in a graphical rather than textual format.

## VI. Merge Algorithm Validation

The merge algorithm was first proposed by Kunde *et al.* [11] as a simple sorting algorithm suitable for parallel computing. It was initially developed for use on parallel processors with local interconnections. It starts by comparing all indexed pairs of neighboring elements in the array. If any pair is in the wrong order, i.e. the first is larger than the second, the elements of the pair get switched. The above step is repeated continuously until all of the elements in the array are sorted. In case of parallel processors, this process takes place simultaneously in all the processing elements depending on instruction on the particular processing element [13].

### A. Algorithm

An ISA implementation of the parallel merge algorithm is illustrated below:

*Step 1:* Sort all columns of the $4 \times 4$ array by odd-even-transposition sort.

*Step 2:* Sort all rows of the $4 \times 4$ array by odd-even-transposition sort.

Odd-Even transposition sort means comparing all odd/even indexed pairs of adjacent elements in the array and if a pair is in the wrong order (the first is larger than the second) the elements are switched to achieve left to right ascending ordering. The ISA program for the merge algorithm is presented in Fig.8. The figure shows the set of instructions and selector bits that will flow through the array. In Fig.8, the instruction and the selector bit parts of the program are represented in a diamond shape made up of their respective instruction and selector bits diagonals. Diagonals 1 to 6 correspond to step 1 and diagonals 7 to 12 to step 2 of the merge algorithm.

### B. Result From the Processor Array

The program was run on the implemented parallel array and produced the performance figures given in Fig.9. The measurements indicate the time difference between the instruction being received and the selector bit being sent between the processing elements. The serial interface to individual controllers was used to interrogate the result. The processing element P(2,1) experiences more no operation instructions (NOPs) than P(3,3) thus P(2,1) takes 29.95 ms and P(3,3) takes 32.45 ms to execute all of the instructions. Results demonstrate that the application executes in 32.45 ms on the discrete prototype ISA implementation. Thus, the merge algorithm has been successfully implemented and validated on the prototype ISA using off-the-shelf microcontrollers.

## VII. Matrix Multiplication Validation

In linear algebra, matrix multiplication plays a key role because the product of the cells are calculated in various stages of many technical problems. Matrix multiplication is well suited to a parallel implementation including ISAs due to their capability for nearest-neighbor communications.

### A. Algorithm

The standard algorithm for matrix multiplication is as follows [14],

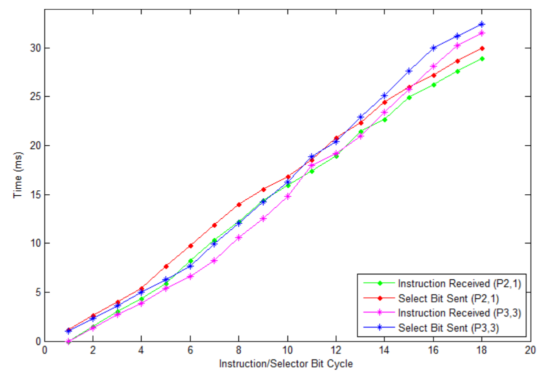


Fig. 9. Performance analysis for P(2,1) and P(3,3).

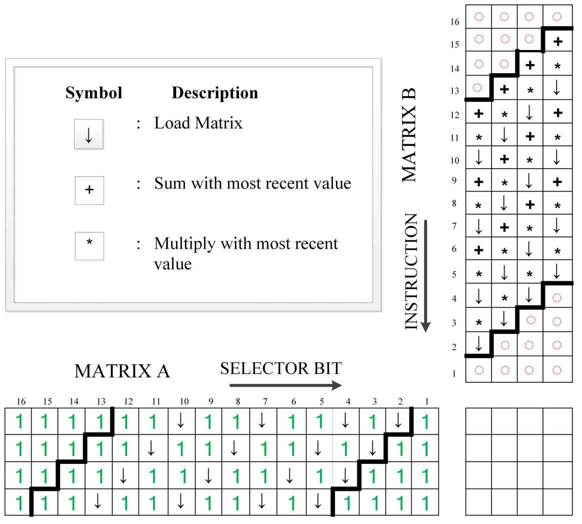


Fig. 10. ISA program for matrix multiplication.

*Step 1:* Each processing element accumulates one element of the product.

*Step 2:* This product is summed with the next element of the product and accumulated in the processing element.

*Step 3:* After all the row and column instructions and selector bits are executed a 4 × 4 matrix is obtained as the result of matrix A and matrix B.

The program shown in Fig.10. is for multiplication of two 4×4 matrices. The first column of matrix A is input at the west of the array, the first row of matrix B is input at the north. In this experiment the input matrices (data) are loaded into the processing elements through both instruction and selector bit array to reduce the number of execution cycles.

## *B. Result From the Processor Array*

Fig.11 shows performance data for matrix-multiplication presented in the same format as for the merge algorithm. In this code the processing elements P(2,1) and P(3,3) have the same number of NOPs and thus they complete the execution of all the instruction at the same time. Results demonstrate that the application executes in 30.95 ms on the prototype ISA implementation.

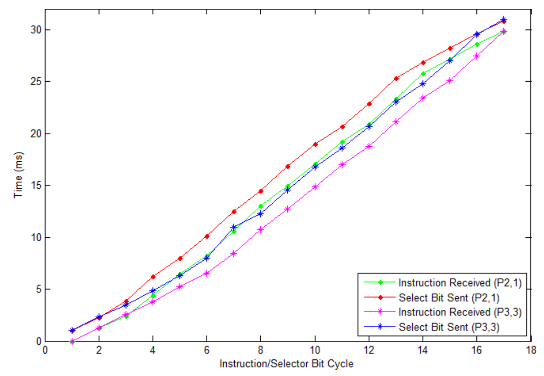


Fig. 11. Performance analysis for P(2,1) and P(3,3).

With the successful execution of two benchmark algorithms, the next section introduces a second, 2D mesh architecture prototype based on the ISA paradigm for distributed computing on fabrics. A real- time shape sensing and reconstruction application is introduced and demonstrated on the second ISA design which is suitable for a wearable system.

## VIII. Shape Reconstruction Application

Wearable self-sensing devices that can measure local deformation are gaining in popularity [15]. Only a small number of studies to date have reported 3D shape reconstruction from a worn sensing device [16], [17]. A potential application for deformation sensing and subsequent shape reconstruction is human posture sensing [18]. Other applications include 3D modelling of an object and wearable motion capture [17]. Real time measurements of the object provide continuous deformations of its shape; therefore, shape sensing applications typically use real time data to reconstruct the shape of an object [18].

One possible sensor that can be applied to such situations is a combined acceleration/magnetic sensor. These provide only two vector observations, which is the minimum for full orientation determination, however no minimization problem can be defined with one sensor. Hermanis *et al.* [19] have proposed the triad based computation for a fast, singularity-free and a computationally simple algorithm. They constructed two triads of orthonormal unit vectors, one formed from the general reference frame, the other from the sensor reference frame. The triads of the earth reference frame and the sensor reference frame are constructed from the earth gravity field vector $E_g$, magnetic field vector $E_m$, sensor measurement of gravity field vector $S_g$ and sensor measurement of magnetic field vector $S_m$. More details of the equations and mathematical approach below can be found in [19]:

$$e_1 = E_g \tag{1}$$

$$e_2 = \frac{E_g \times E_m}{|E_g \times E_m|} \tag{2}$$

$$e_3 = e_1 \times e_2 \tag{3}$$

$$s_1 = S_g \tag{4}$$

$$s_2 = \frac{S_g \times S_m}{|S_g \times S_m|} \tag{5}$$

$$s_3 = s_1 \times s_2 \tag{6}$$

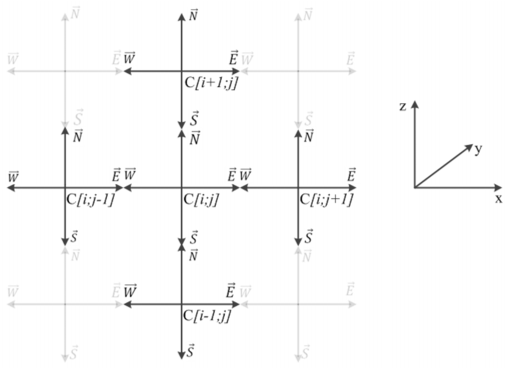


Fig. 12. Surface segment structure. Each segment consists of centre C and four direction vectors N, E, S and W.

These triads are then used to form a matrix for global Earth reference, represented as $M_e$:

$$M_e = [e_1\ e_2\ e_3] \tag{7}$$

and matrix for sensor measurements represented as $M_s$:

$$M_s = [s_1\ s_2\ s_3] \tag{8}$$

The rotation matrix $R$ describes sensor orientation relative to the global reference frame and is calculated using the formula:

$$R = M_e M_s^T \tag{9}$$

The rotational matrix is used to calculate the surface segment orientation relative to the initial position that corresponds to sensor orientation relative to earth reference frame. Acceleration and magnetic sensor nodes are arranged in a regular grid along the surface. The model of the surface is divided into $n$ rigid segments. Each segment is described by four direction vectors.

$$\vec{N}_b = \left[0; 0; \frac{L_1}{2}\right]$$
$$\vec{E}_b = \left[\frac{L_2}{2}; 0; 0\right] \tag{10}$$
$$\vec{S}_b = \left[0; 0; -\frac{L_1}{2}\right] = -\vec{N}_b$$
$$\vec{W}_b = \left[-\frac{L_2}{2}; 0; 0\right] = -\vec{E}_b \tag{11}$$

From the segment structure of Fig.12 it can be deduced, that if a single control point location is known, then any other control point on the same segment row or column can be calculated by adding and subtracting the corresponding segment direction vectors from the following expressions,

$$C\left[i; j_{ref}\right] = C\left[i_{ref}; j_{ref}\right] + \sum_{k=i}^{i_{ref}-1} \left(-\vec{N}\left[k, j_{ref}\right] + \vec{S}\left[k+1, j_{ref}\right]\right) \quad \text{if } \left(i < i_{ref}\right) \tag{12}$$

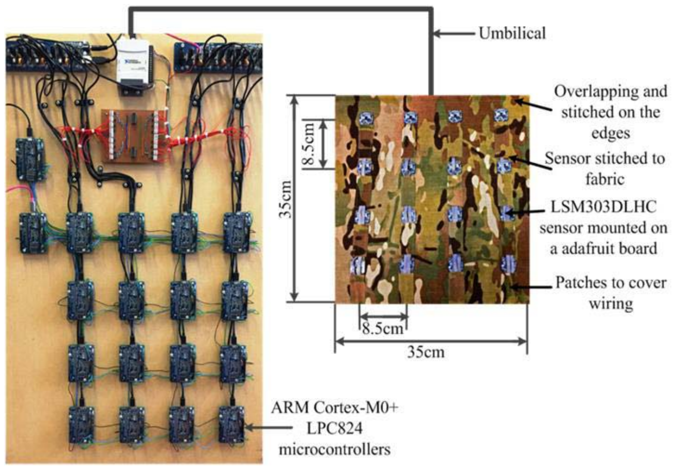

Umbilical
Overlapping and stitched on the edges
Sensor stitched to fabric
LSM303DLHC sensor mounted on a adafruit board
Patches to cover wiring
8.5cm
35cm
8.5cm
35cm
ARM Cortex-M0+ LPC824 microcontrollers

Fig. 13. Prototype board with microcontrollers (left) and Sensors embedded in fabric (right) [12].

Similarly, control points on the reference row ($i = i_{ref}$) can be calculated as:

$$C\left[i_{ref}; j\right] = C\left[i_{ref}; j_{ref}\right] + \sum_{k=j_{ref}}^{j-1} \left(\vec{E}\left[i_{ref}, k\right] - \vec{W}\left[i_{ref}, k+1\right]\right) \quad \text{if } \left(j > j_{ref}\right) \tag{13}$$

### A. Sensor-Equipped ISA Implementation

To implement the surface reconstruction the 4 × 4 array mentioned earlier was equipped with a sensor network consisting of 16 sensors (type LSM303DLHC acceleration/magnetic) stitched into a 35×35 cm fabric swatch as shown in Fig.13. An ISA implementation of the algorithm described in the previous section was used to measure performance. Each microcontroller, acting as a processing element, was assigned a unique ID to identify its position in the grid and calculated its allocated control points. Since the sensors are local to the processing element its values are immediately accessible. The orientation data is then averaged and stored for calculation of directional vectors. These directional vectors are shared between neighboring microcontrollers for the calculation of control points. Once these control points are calculated for each sensor, they are sent to the host computer via a serial port for 3D visualisation of the sensed object. The process of ISA computing the control points and the host drawing the visualisation continues indefinitely. The umblical shown in Fig.13 is for practical reasons. In a fully realised design the processing elements will be directly connected to the sensor array.

The instruction and selector bit of the ISA firmware (Fig.14) can be seen as diamond shaped consisting of instructions and selector bits respectively. This diagonal of instruction and their corresponding selector bit is used for implementing the shape reconstruction application. A set of NOPs is flushed through the array before and after the instruction and selector bit diagonal.

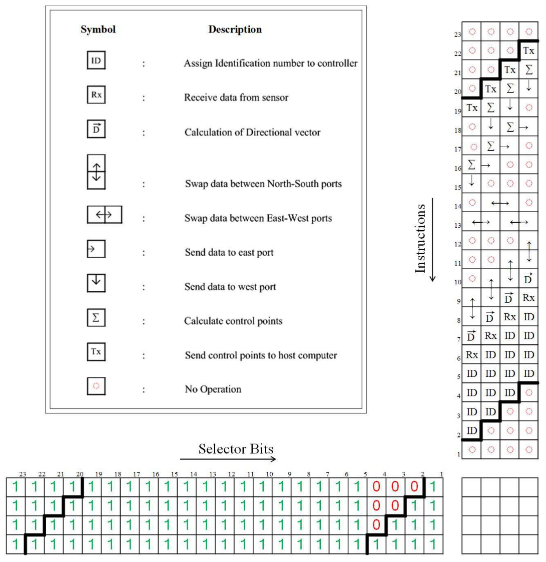
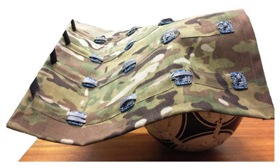
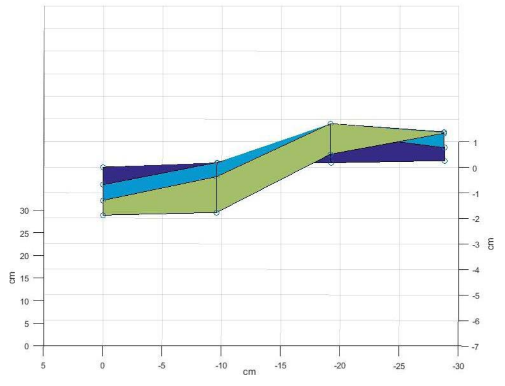

Symbol
Description
ID : Assign Identification number to controller
Rx : Receive data from sensor
D : Calculation of Directional vector
: Swap data between North-South ports
: Swap data between East-West ports
: Send data to east port
: Send data to west port
Σ : Calculate control points
Tx : Send control points to host computer
: No Operation
Instructions
Selector Bits

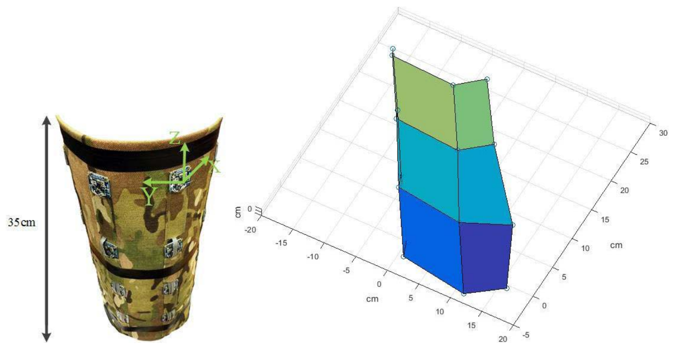
Fig. 14. Shape-sensing ISA program.


(a)



(b)

Fig. 16. (a) Fabric placed on the object. (b) Reconstructed shape.



(a) (b)

Fig. 15. (a) Fabric wrapped on a cylindrical object. (b) Reconstructed shape of the object.

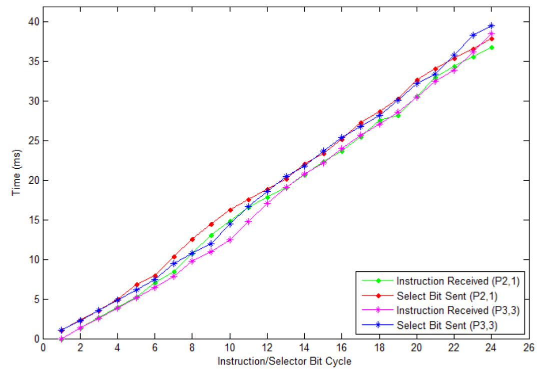


Fig. 17. Performance analysis for P(2,1) and P(3,3).

## *B. Experimental Results*

To evaluate the accuracy of proposed shape sensing method, a number of experiments were conducted by wrapping the fabric onto different objects. The first experiment involved wrapping the fabric around a cylindrical object with a diameter 15cm and height 35cm, which was resting on one of its end faces on a horizontal table and then reconstructing its shape. The fabric swatch wrapped around the object is shown in Fig.15(a). The reconstructed image of the cylindrical object is shown in Fig.15(b). The X, Y and Z axis represents the calculated distance between the sensors in cm.

The second experiment involved placing the fabric on a ball with a diameter 65cm and then reconstructing the shape. The fabric swatch placed on the object is shown in Fig.16(a) and the reconstructed shape of is shown in Fig.16(b).

The reconstructed shape represents minor deviation from the sensor location mainly on the boundary sensors. For reference, though this is not dependent on the ISA architecture, the accuracy was found to be within 0.4 cm of the expected position. There are numerous error sources which originate from sensor noise and sensor mechanical mounting errors. The sensor noise introduces errors to the measurement of Earth gravity and magnetic field vector component. Sensor mechanical mounting errors include orientation errors, which introduce misalignment of sensor reference frame and placement errors, which introduces differences in inter-sensor distances leading to orientation error.

Fig.17 shows the execution performance of the program in the same format to the benchmarks. A few instructions take longer to execute because of their implementation complexity. For example the seventh instruction on P(2,1) and the tenth instruction on P(3,3) which is a sensor read and takes a few milliseconds to carry out, average and store in the register for further computation. In the current implementation, shared buses are being used which act as a bottle neck.

Therefore delays occur through the communication. In a custom design, sensors could be more closely coupled to the processing element and the implementation can be carried out concurrently with the ISA processing function.

The wearable shape reconstruction application has been successfully implemented on the second ISA prototype system and executes in 39.55 ms on thus confirming the viability of our approach for fabric-resident computing devices.

## IX. Conclusion

The paper has discussed the rationale, design, implementation and benchmarking of a new concept for on-fabric sensor networks, prototyped with off-the-shelf microcontrollers. A physical prototype device has been demonstrated containing 16 processing nodes. The concept has been validated using several programming examples. The parallel architecture has been demonstrated on a fabric using a real time application.

It is envisaged that such a system would be implemented using VLSI technology and custom ASICs which would substancially improve the performance. The future work can address the scalability of the architecture in line with the vision to extend to large arrays and new applications. Several wearable applications in the field of medicine, military and sports can also be explored using the concepts and methodologies developed during this research. There can also be focus on extending the supported instructions, optimizing the communication medium and allowing for more concurrency, at node level, between computation and communication.

## References

[1] S. L. P. Tang, "Recent developments in flexible wearable electronics for monitoring applications," *Trans. Inst. Meas. Control*, vol. 29, nos. 3–4, pp. 283–300, Jul. 2016.

[2] J. McCann and D. Bryson, *Smart Clothes and Wearable Technology*, 1st ed. Sawston, U.K.: Woodhead, 2009.

[3] A. Cerpa and D. Estrin, "ASCENT: Adaptive self-configuring sensor network topologies," Dept. Comput. Sci., UCLA, Los Angeles, CA, USA, Tech. Rep. UCLA/CSD-TR-01-0009, May 2001.

[4] A. Lim, "Distributed services for information dissemination in self-organizing sensor networks," *J. Franklin Inst.*, vol. 338, no. 6, pp. 707–727, 2001.

[5] K. Van Laerhoven, A. Schmidt, and H.-W. Gellersen, "Multi-sensor context aware clothing," in *Proc. 6th Int. Symp. Wearable Comput.*, 2002, pp. 49–56.

[6] F. Lorussi, W. Rocchia, E. P. Scilingo, A. Tognetti, and D. De Rossi, "Wearable, redundant fabric-based sensor arrays for reconstruction of body segment posture," *IEEE Sensors J.*, vol. 4, no. 6, pp. 807–818, Dec. 2004.

[7] M. Stoppa and A. Chiolerio, "Wearable electronics and smart textiles: A critical review," *Sensors*, vol. 14, no. 7, pp. 11957–11992, 2014.

[8] I. P. I. Pappas, T. Keller, S. Mangold, M. R. Popovic, V. Dietz, and M. Morari, "A reliable gyroscope-based gait-phase detection sensor embedded in a shoe insole," *IEEE Sensors J.*, vol. 4, no. 2, pp. 268–274, Apr. 2004.

[9] I. Locher, "Technologies for system-on-textile integration," Ph.D. dissertation, Dept. Inf. Technol. Elect. Eng., Swiss Fed. Inst. Technol., Zürich, Switzerland, 2006. [Online]. Available: https://www.research-collection.ethz.ch/handle/20.500.11850/149105

[10] H. T. Kung and C. E. Leiserson, "Systolic arrays (for VLSI)," Carnegie Mellon Univ., Dept. Comput. Sci., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-79-103, 1978.

[11] M. Kunde, H.-W. Lang, M. Schimmler, H. Schmeck, and H. Schröder, "The instruction systolic array and its relation to other models of parallel computers," *Parallel Comput.*, vol. 7, no. 1, pp. 25–39, Apr. 1988.

[12] P. Kandaswamy, J. A. Flint, and V. Chouliaras, "Shape reconstruction using instruction systolic array," in *Proc. IEEE Sensors*, Oct./Nov. 2017, pp. 364–366.

[13] H.-W. Lang, M. Schimmler, H. Schmeck, and H. Schröder, "Systolic sorting on a mesh-connected network," *IEEE Trans. Comput.*, vol. 34, no. 7, pp. 652–658, Jul. 1985.

[14] B. Schmidt, "Techniques for algorithm design on the instruction systolic array," Ph.D. dissertation, Dept. Comput. Sci., Loughborough Univ., Loughborough, U.K., 1999. [Online]. Available: https://dspace.lboro.ac.uk/2134/7161

[15] E. Koch and A. Dietzel, "Skin attachable flexible sensor array for respiratory monitoring," *Sens. Actuators A, Phys.*, vol. 250, pp. 138–144, Oct. 2016.

[16] P. J. Besl and D. N. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992.

[17] J. F. Vasconcelos, G. Elkaim, C. Silvestre, P. Oliveira, and B. Cardeira, "Geometric approach to strapdown magnetometer calibration in sensor frame," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 47, no. 2, pp. 1293–1306, Apr. 2011.

[18] T. Hoshi and H. Shinoda, "3D shape capture sheet based on gravity and geomagnetic sensing," in *Proc. 24th Sens. Symp.*, 2007, pp. 423–427.

[19] A. Hermanis, R. Cacurs, and M. Greitans, "Acceleration and magnetic sensor network for shape sensing," *IEEE Sensors J.*, vol. 16, no. 5, pp. 1271–1280, Mar. 2016.

**Partheepan Kandaswamy** was born in Chennai, India, in 1989. He received the B.E. degree in electronics and communication from Anna University, India, in 2011, and the M.Sc. degree in embedded microelectronics and wireless systems from Coventry University, U.K., in 2013. He is currently pursuing the Ph.D. degree with Loughborough University, U.K. His research interests include body sensor networks and on-fabric distributed computing. He is a member of the IET.

**James A. Flint** received the M.Eng. and Ph.D. degrees in electronic and electrical engineering from Loughborough University, in 1996 and 2000, respectively. He was a project engineer in the automotive industry. He became a Lecturer of Wireless Systems Engineering at Loughborough University in 2001, a Senior Lecturer in 2006, and a Reader in Wireless Systems Engineering in 2017. He has acted as a consultant to numerous high-profile companies in the wireless and automotive sectors. His research interests include fault-tolerant signal processing, novel acoustic and electromagnetic transducers, metamaterials, and electromagnetic compatibility. He is a Fellow of the IET and a Chartered Engineer in the U.K.

**Vassilios A. Chouliaras** was born in Athens, Greece, in 1969. He received the B.Sc. degree in physics and laser science from Heriot-Watt University, Edinburgh, U.K., in 1993, the M.Sc. degree in VLSI systems engineering from UMIST in 1995, and the Ph.D. degree from Loughborough University, U.K., in 2005. He was one of the founders of Axilica Ltd., a start-up company commercializing disruptive research and development into FPGA behavioral synthesis from UML system descriptions. He was an ASIC design engineer for a Telecom OEM and a processor architect for a configurable, extensible embedded processor vendor. He is currently a Senior Lecturer of Microelectronics and Computer Architecture with the Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, U.K. His research interests include CPU micro-architecture, high-performance embedded CPU implementations, OpenCL computing, custom instruction set design, and electronic system level design methodologies.