In-network processing based hardware acceleration for situational awareness

Md Fasiul Alam[1]

National and Kapodistrian University of Athens
Department of Informatics and Telecommunications
fasiul@di.uoa.gr

Abstract. In this research, an In-network processing (INP) based computational framework has been proposed that gradually refine the captured information while moving it upstream to the application. The refinement can go up to the point of knowledge extraction from the primitive or even fused data. It shows that the demanding tasks that previously were simply undertaken on the fixed infrastructure are now possible on the mobile end. It relies on a sophisticated in-network processing scheme that gradually refines the information captured by sensing elements to the level of application-exploitable knowledge. With this process, the scaling problem is tackled much more efficiently through a problem segmentation and exploitation of physics-based and human-based sources. The components executing on INP nodes are structured appropriately to delegate the demanding subtasks to some onboard accelerator. The core program executes on the central processing unit and exploits the particular characteristics of the side processor. Special system-on-chip (SoC) hardware for computational acceleration like central processor unit (CPU), digital signal processor (DSP) or field-programmable gate arrays (FPGAs) are desirable. These accelerated hardware supports software defined process on-chip memory that expedites all the advanced processing with a minimum energy overhead. In-network software defined processing (ISDP) capabilities has been considered that allows dynamic reconfiguration of the network topology. It is advantageous for the network to possess reliable and complete end-to-end network connectivity; however, even when the network is not fully connected, the system may act as conduits of information — either by connectivity gaps, or by distributing information from the network space.

Keywords: In-network Processing, Hardware acceleration, low power process, WSN, IoT, Situation awareness

[1]Desertion advisor: Stathes Hadjiefthymiades, Professor

1 Introduction

Due to the rapid development and spread of embedded computer technology over the last decade [1], sensor nodes are widely used in situational awareness and produce potentially abundant information for IoT applications. However, the disconnected, intermittent and limited communication environment that these nodes often operate in make the usage of internet-level communication solutions not applicable on the WSN level [2-4]. The direct consequence is the increase (already in place) of the amount of big data to be analyzed, which in the industrial field translates in the need to equip itself with platforms of data storage of enormous proportions [5]. The exponential increase of the data to be analyzed leads to the need for adopt new ways to provide answers immediate and reliable applications to high degree of criticality, which do not tolerate latencies in communication. A growing number of contemporary IoT applications require more from WSNs than simple data acquisition, (conditional) communication and collection to databases. For example, maintenance in nuclear applications, such as ISR systems, aim to improve the situation awareness of decision makers and expect preprocessed data that are already converted to human understandable form.

Data collection to central databases, as used in typical WSNs for monitoring, creates overhead, potential bottlenecks and offers limited resilience [6]. An additional aspect is that some of the potential WSN users, such as in-the-field extreme environment maintenance (i.e ATLAS, CERN), need situational information in a timely manner and would prefer to receive information tailored to their current information needs directly from the network to minimize delays and dependence on central infrastructure. In order to manage the large data flows and to minimize the bandwidth requirements, novel paradigms such as D2D and mist computing (an extension of fog computing) need to be exploited. Traditional data aggregation is a predecessor of these paradigms, but in its classical form is not enough for IoT applications, because the traditional flow of data from network edge (sensor nodes) to center (databases) remains.

In general, performing intelligence operations inside the network, such as eliminating irrelevant records and aggregating raw data, can reduce energy consumption and improve sensor network lifetime significantly [7]. This is referred to as sensor data processing, in which an intermediate proxy node is chosen to house the data transformation function to consolidate the sensor data streams from the data source nodes, before forwarding the processed stream to the sink. Sensor based IoT nodes sense, receive, process and transmit data to other nodes. In these functions, a node may exhibit different degrees of intelligence and sophistication. Some nodes involve little processing and transmit small quantities of information. Other are connected to sensors that generate large quantities of data (e.g. cameras), require large storage, high processing power and may transmit many data. Other types of nodes (i.e., knowledge discovery, consensus, reasoning or fusion) also exhibit varying needs. Consequently, node capabilities vary from highly restricted resources in terms of processing, storage, transmission bandwidth and energy to devices that compare to current smart phones in terms of resources. In-network processing is a technique employed in sensor database systems whereby the

data recorded is processed by the sensor nodes themselves. This is in contrast to the standard approach, which demands that data is routed to a so-called sink computer located outside the sensor network for processing. In-network processing is critical for IoT based sensor nodes because they are highly resource constrained, in particular in terms of battery power and this approach can extend their useful life quite considerably. Based on their capabilities, the nodes are assigned a specific role/type in order to achieve a certain task. Different nodes can cooperate in the execution of some workflow in order to achieve some goal/task also based on the semantics of the processed data. For instance, a set of nodes can participate in the execution of a workflow, which implements a high level fusion operator or, even, a distributed classification algorithm. The node can be envisaged as the basic INP units through which collaborative intelligence can be attained. Energy consumption is a crucial factor in a sensor network. INP is a useful technique to reduce the energy consumption significantly. Many different processing modules within the IoT infrastructure that can gradually refine the captured information while moving it upstream to the application. The refinement can go up to the point of knowledge extraction from the primitive or even fused IoT data. The components executing on such nodes (Fig. 1) are structured appropriately to delegate the demanding subtasks to some onboard accelerator (e.g., DSP, GPU). The core program executes on the central processing unit and exploits the particular characteristics of the side processor. Energy is the dominant constraint. Because the quantities of information to process are much larger and the processing algorithms are much heavier, nodes with higher processing capabilities are needed. Special hardware for computational acceleration like DSP or FPGAs is desirable.

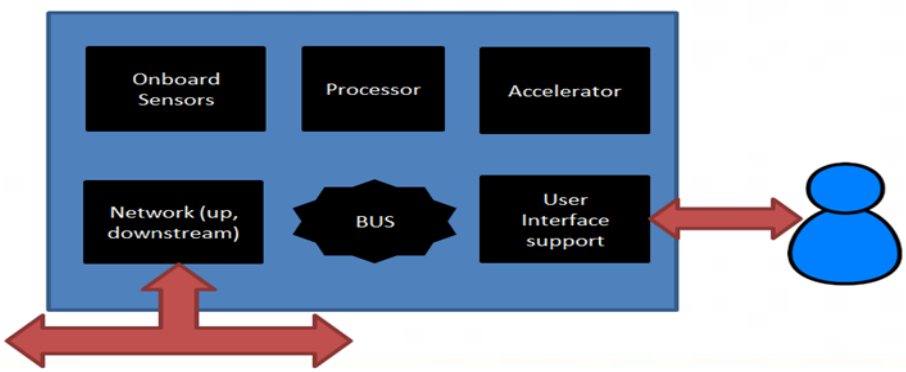


Fig. 1. INP Node

2 In-network Processing Architecture

In-network processing architecture usually involves information filtering/transformation nodes that rely on computationally demanding algorithms. Such nodes are based on accelerated, not conventional hardware that expedites all the advanced processing with a minimum energy overhead. This is imperative for the efficient operation of the

WSN as it addresses two important needs: performing demanding tasks at low energy cost, filtering the information to support energy efficiency and render the network sustainable over time. Different schemes for the processing of IoT generated data have been investigated in this chapter. The objective is to reduce the application processing needs in the discussed domains and drastically reduce the volume of data seen by the application. Currently, applications collect huge volumes of IoT data in their support databases for further processing in line with specific business logic. Many different processing modules within the IoT infrastructure have been orchestrated that gradually refine the captured information while moving it upstream to the application. The refinement can go up to the point of knowledge extraction from the primitive or even fused IoT data. With this scheme, the originally identified scaling problem is tackled much more efficiently through a problem segmentation and exploitation of in-network processing. The components executing on such nodes are structured appropriately so as to delegate the demanding subtasks to some onboard accelerator (e.g., DSP, GPU). The core program executes on the central processing unit and exploits the particular characteristics of the side processor. Energy is the dominant constraint. Because the quantities of information to process are much larger and the processing algorithms are much heavier, nodes with higher processing capabilities are needed. Special hardware for computational acceleration like digital signal processor (DSP) or field-programmable gate arrays (FPGAs) is desirable. As these nodes must be able to operate on batteries for relatively long periods devices that resemble modern smart phones may satisfy their requirements. IoT nodes sense, receive, process and transmit data to other nodes. In these functions, a node may exhibit different degrees of intelligence and sophistication. Some nodes involve little processing and transmit small quantities of information. Other are connected to sensors that generate large quantities of data (e.g. cameras), require large storage, high processing power and may transmit a lot of data. Other types of nodes (i.e., knowledge discovery, consensus, reasoning or fusion) also exhibit varying needs. Therefore, node capabilities vary from highly restricted resources in terms of processing, storage, transmission bandwidth and energy to devices that compare to current smart phones in terms of resources. Based on their capabilities, the nodes are assigned a specific role/type in order to achieve a certain task. Different nodes can cooperate in the execution of some workflow in order to achieve some goal/task also based on the semantics of the processed data. For instance, a set of nodes can participate in the execution of a workflow, which implements a high level fusion operator or, even, a distributed classification algorithm. The node can be envisaged as the basic INP units through which collaborative intelligence can be attained.

Nodes can be categorized into the following sub-types subject to their capability:

- Sensing (S) node,
- Fusion (F) node,
- Consensus (C) node,
- Knowledge extraction (K) node,
- Sink (M) node.

Nodes can participate in the execution of some workflow in order to achieve some task/goal. The set of nodes that participates in the execution of some workflow can be also viewed as a composite node with certain input, output, and capability. A composite node can be further aggregated with other composite and/or basic nodes in order to construct an even more complex node, forming tree like structures. Based on such constructors of nodes, the main network of nodes and an overall view of the architecture are depicted in Fig. 2. Fig. 2 shows the role hierarchy in the user (data) plane. The several types of node are explained and analyzed below.

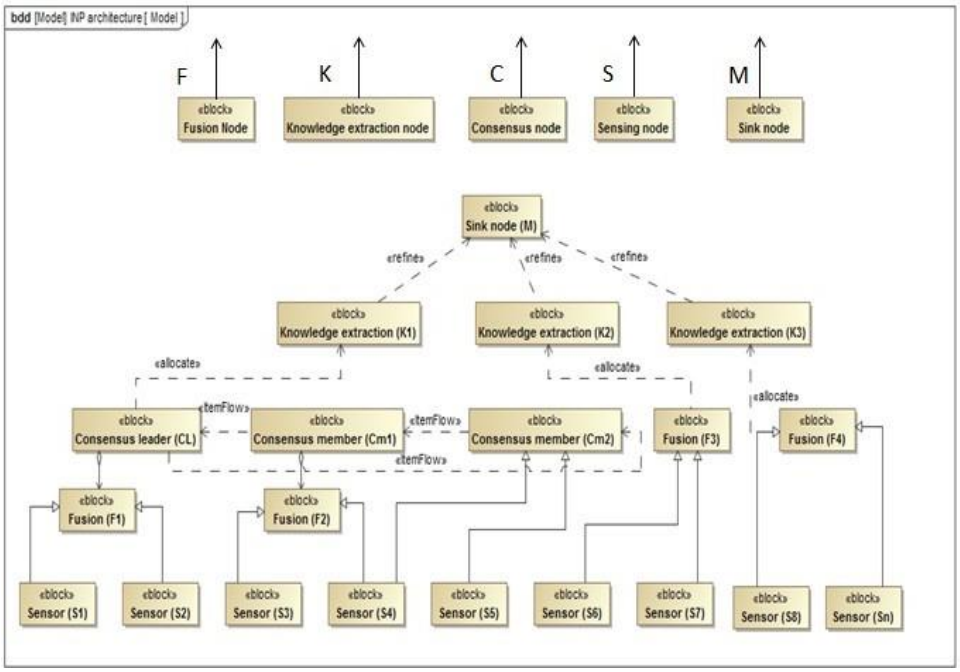
2.1 Sensing Node ('S' node)

The sensing node captures data from the environment and propagates this information to the network (either to 'F' or 'C' nodes). The processing capabilities of the sensing node can be limited since its main task is to sense and relay pieces of data to the upstream node. Additionally, several alternate routing schemes could be investigated as an add-on feature to optimize the overall network performance and avoid redundant retransmissions. The 'S' node, and each node that relays information, apart from processing, can adapt its relay/data dissemination mechanism. For instance, the 'S' node can adapt the key operational parameters of an epidemic-based information dissemination scheme (i.e., the forwarding probability and validity period). The sensing node can be embedded to various sensing devices, thus, a sensing device consists of numerous sensing 'S' nodes. The sensing device can, then, be abstracted as a composite 'S' node. For instance, numerous vision sensors can be implemented as an 'S' node with onboard cameras and DSP facilities. A vision sensor produces readings (not video) for image segments (tiles). An indicative example involves fire detection through a vision sensor that segments images into 16 x 16 tiles. Each tile is handled independently within the camera sensor network (and the originating node of course).

2.2 Fusion Node ('F' Node)

The fusion node collects data from heterogeneous sources ('S' nodes or other 'F' nodes) and performs fusion operations in order to deduce more accurate information. The heterogeneity of the received information is bridged by the data semantics profile of each 'S' and 'F' node. That is, the 'F' node is aware of the different type of pieces of data that is responsible to apply fusion operators. Obviously, conventional aggregator operators (e.g., statistical mean, min/max operators) can be applied on data of same type. However, the 'F' node can apply intelligent information fusion techniques (e.g., theory of evidence) on pieces of data of different types. The 'F' node is capable of fusing based on certain quality/validity metrics. The deduced information is provided as input either to 'K' nodes or to other 'F' nodes. In the latter case, the output of a first-level fusion process feeds a second-level fusion (multi-level fusion) in order to conclude to information with higher degree of confidence. That is, a composite 'F' node represents a two-level fusion scheme, which consists of a set of 'F' nodes that fuse data from diverse data types. For instance, consider an 'F' node which produces probability of a fire event by fusing the results of (a) 'F' nodes, which aggregate temperature values, (b) of 'F'

nodes, which aggregate humidity values, and (c) of ‘F’ nodes which produce probabil-



ity of smoke and fire detection. One of the most obvious examples is receiving the input of a number of ‘S’ nodes and fusing them using some unsupervised learning method, e.g. PCA, k-means, etc., for dimensionality reduction.

Fig. 2. INP Architectures (Roles, Connections)

2.3 Consensus Node (‘C’ node)

Consider a neighborhood of ‘S’ and ‘F’ nodes, which is spatially defined. The ‘S’ nodes monitor contextual parameters and the ‘F’ nodes aggregate the corresponding measurements to compute an estimate in a completely distributed way. However, in order to deliver the locally measured data to a common (composite) ‘F’ node, the ‘S’ and ‘F’ nodes exchange their data by performing pair-wise aggregator operations (e.g., average), thus, converging to a common value for such nodes; then consensus is reached. The local estimate of the neighborhood is recorded at each participant node and, thus, can be recovered from any ‘surviving’ node in the neighborhood. Such neighboring nodes process and store information locally, as typical ‘S’ and ‘F’ nodes, but they behave as a single unit, the so called ‘C’ node. The nodes of a consensus group (neighborhood) try to harmonize possible inconsistencies in the received information so the whole group to conclude to a shared and commonly accepted measurement and/or knowledge. In Fig. 2, CM nodes are just members (nodes) of a consensus group that

communicate to each other while node CL is the leader of the group that represents the whole group in the network (e.g., it exchanges information with external nodes).

2.4 Knowledge Extraction Node ('K' node)

The 'K' node performs machine learning and data mining operations to extract new knowledge, such as classification models, frequent patterns, novelty and outlier detection, from the different data sources generated from the network nodes. A 'K' node can use as input the output generated by different node types, 'F', 'C' and even 'K'. In addition a 'K' node can also coordinate the distributed execution of data mining and machine learning operators over the different nodes of the network, in order to reduce the communication load, for example either by performing local dimensionality reduction via the 'F' nodes, or by requesting the generation of local models, and subsequently have the parameters/outcome of these models communicated instead of the actual data.

2.5 Sink Node (M)

The main role of the sink is to conceal IoT heterogeneity in terms of sensors, actuators, networking, or middleware through an interconnection unit. The sink node (M) concentrates the data stemming from the underlying IoT devices and performs the operations needed before feeding the applications with the requested information. The "M" node can be considered as a mediator between the underlying IoT and the application layer. The applications are fed with information by the network and are agnostic to the operation of the underlying IoT. A middleware capable of supporting intelligent pervasive applications can materialize this layer.

3 Energy Efficient Parallel Processing

This section describes the acceleration of in-network operations by means of energy efficient hardware. The adoption of a scheme that relies heavily on INP renders the architecture highly efficient, long lasting and drastically reduces the data processing load that the (sensor-supported) application or middleware should sustain. Even though such IoT architectures with increased INP characteristics can be designed and operated over conventional IoT hardware (e.g., motes with conventional processing elements) the use of accelerated nodes would further increase efficiency, thus, boosting the benefits of INP. The approach to follow in such scenarios is to identify the merits of the accelerated hardware that is currently available for the IoT deployment and contrast these to the peculiarities of the algorithms that are foreseen in the INP architecture. In this chapter, Parallella platform [8], which facilitates the energy efficient parallelization of tasks within resource-constrained nodes, have been focused. Here the important aspect is parallelization. Hence, the idea is to identify all the roles/algorithms that feature internal operations/tasks with clear parallelization needs/capabilities (e.g., operations on matrices). Therefore, the structure on the INP building blocks that are assuming is the one shown in Fig. 3. The algorithm is fed with several sources of data that, generally, need to undergo some preprocessing phase. The general-purpose processor should perform this data-preprocessing phase together with coordination tasks, which is part

of the accelerated hardware. The non-parallelizable task may also orchestrate the delivery of data to the memories of the parallel processing element. Then, the parallelized tasks are invoked followed by the collection/consolidation of results and execution resumes at the general-purpose processor.

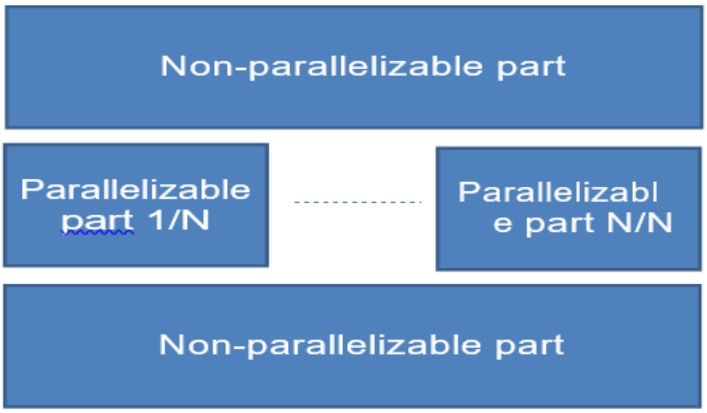


Fig. 3. Parallelized INP component

The algorithms that were introduced in INP integrated parts that can be efficiently handled in parallel. Typical examples are the PCA technique for dimensionality reduction and data compression, the multi-dimensional event detection and a data aggregation scheme that relies on FFT.

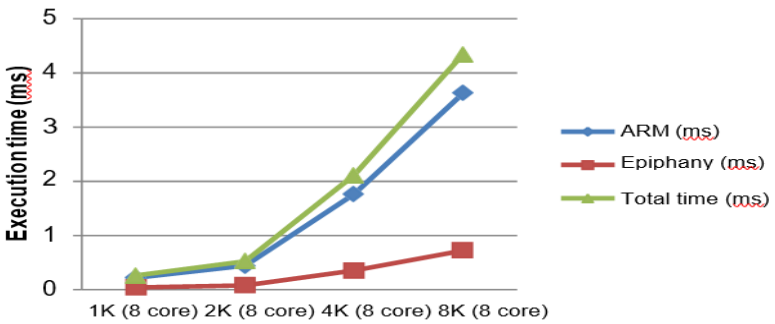


Fig. 4. FFT Turnaround times on Parallella (ARM Epiphany-ARM)

Fig. 4 shows the turnaround time (TAT) in the ARM-Epiphany combination for demanding FFT tasks. Specifically, the plot shows how TAT scales for different FFT sizes. FFT is invoked on the 4 and 8 cores processor in this benchmark but can be invoked in parallel on all 16 cores. Therefore, the Epiphany can be called to FFT 1024

samples (1K) for 16 streams in parallel. From the obtained results it is clear that times overlap and TAT minimally impacted as the number of streams increases from 1 to 16. Experimental results demonstrate up to 60% and 64% reduction in latency for GPU-to-GPU and CPU- to-GPU point to- point communications, respectively.

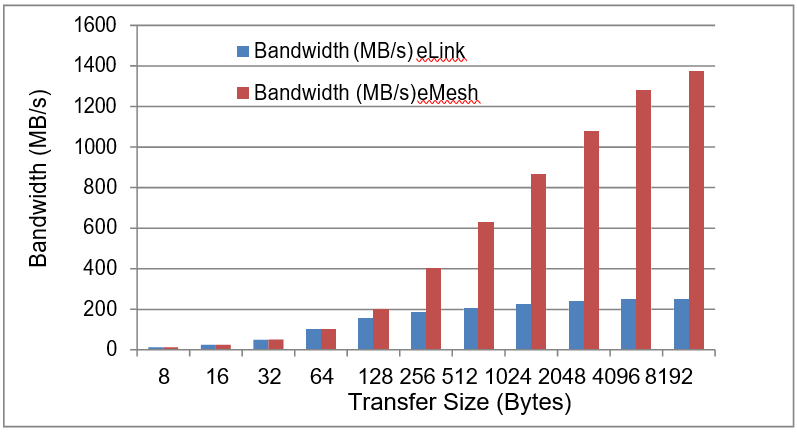


Fig. 5. Data transfer size over bandwidth

The results for the DMA and direct-write transfer size measurements are shown in Fig. 5. First the DMA transfer function provided by the SDK library was tested over the eMesh and eLink for different transfer sizes, capturing both the total transfer time and start-up time. Here we see a very large portion of the time spent sending data can be attributed to starting up the DMA engine (65.2% to 6.9%) in Fig. 6.

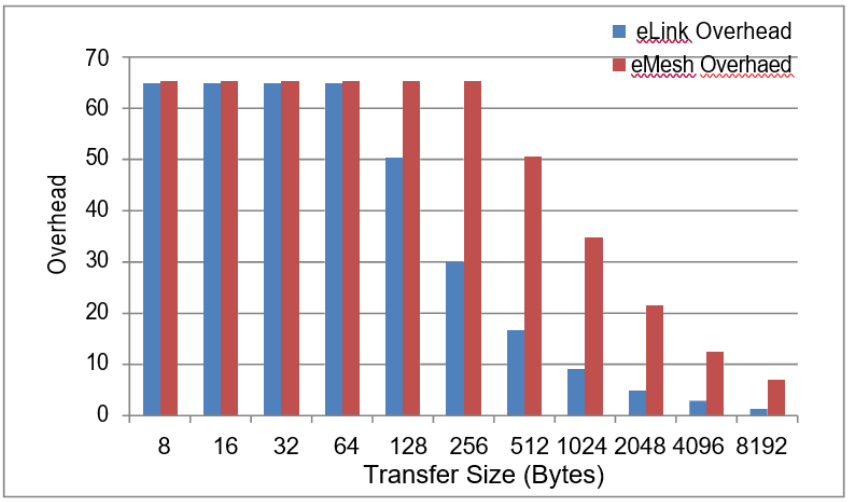


Fig. 6. DMA transfer size versus overhead

Several nodes within the IoT are implemented through accelerated hardware which

(a) Are assigned very specific roles in the data transformation process (raw data sampled somewhere are progressively turned into valuable knowledge)

(b) Have the capability of fulfilling such roles very efficiently and avoid unnecessary transmissions through the network paths thus rationalizing the use of scarce energy

(c) Have the capability to promptly react to changes in the network architecture and mitigate the "disturbances" to the overall data transformation plan through their reconfiguration capabilities.

The accelerated hardware that is considered is based on Software Define (SD) SoC elements that combine a conventional processing element like ARM processor with hardware programmability of FPGA [9-10]. The FPGA part of the node architecture is handling the core functionality that the assigned role prescribes for this particular node (e.g., event detection, spectral decomposition, fusion). This is installed/deployed in the FPGA according to the initial role assignment/planning that the network planners derive. Nodes should be equipped with the necessary components which are held inactive until external control stimulates the node. Within each node a supervisor/control process is typically executed on the processor part of the SoC (Fig. 7).

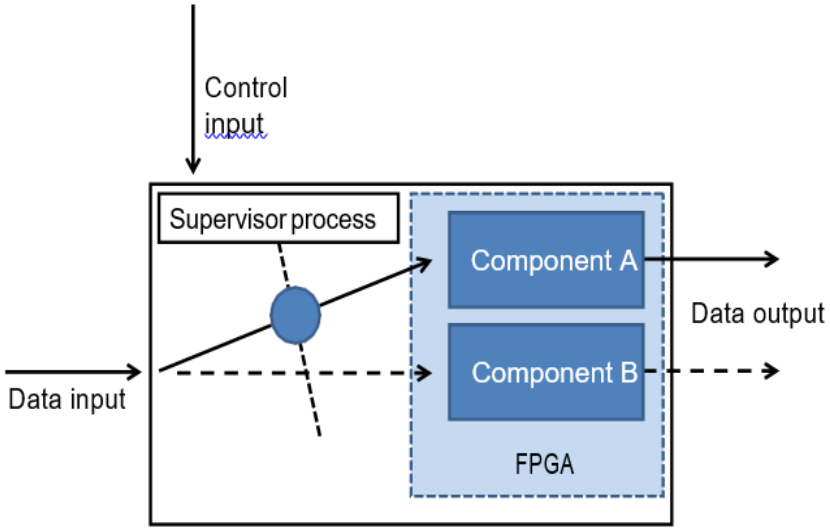


Fig. 7. Control process of IoT network

Table 1. Comparison of measurements

Packet size (Bytes)	32	128	512	2048	8192	32768
Number of packets	134217728	33554432	8388608	2097152	524288	131072
Time (s)	219	66	29	21	18.2	17.6
Speed (Megabytes/s)	19.61	65.08	148.10	204.52	235.99	244.03
PL clock @100 Maximum Burst size 16, total transfer size 4294967296 bytes						

Simulation is run for different packet sizes and for each packet size, performance is measured. The Table 1 summarizes the measurements. The measurement are carried out with 100 MHZ with burst size 16 and 256 respectively. These configurations absolutely depends on how we define ZYNQ HW design in Vivado (for example the Clock frequency, the burst size etc.). Four GBytes of data to DRAM have been transferred and from that speed of transfer is obtained. Packet size has been defined for each test. Total transfer size is divided by total packet in order to get number of packets. From that, performance is calculated (time and speed).

4 Conclusion

A generalized architecture of INP has been discussed in this thesis paper taking into account the role of each components interfaces to ensure efficient exchange of the information and optimization of the overall resources. In conventional network processing scenarios, problems frequently arise in a pipeline when certain data or pieces of information are not be readily accessible or available at the time they are needed by the pipeline or when computations take longer as a result of an exceptional condition. These problems contribute to an overall slowdown of the processing pipeline and lead to undesirable data transmission/processing stalls that markedly reduce performance. The INP system overcomes many of these issues and limitations by implementing discrete processing paths wherein each processing path is directed towards handling network traffic and data of a particular composition. The system architecture combines the energy efficiency with the flexibility and programmability of a system on a chip processor. As power consumption is the highest priority design constraint, the proposed system for WSNs/IoT uses two techniques to reduce power consumption. 1) Lightweight event handling in hardware: initial responsibility for handling incoming interrupts is given to a specialized processor, removing the software overhead that would be

required to provide event handling on a general-purpose processor. 2) Hardware acceleration for typical WSN/IoT tasks: modular hardware accelerators are included to complete regular application tasks such as data filtering.

References

1. Myers, Brad, Scott E. Hudson, and Randy Pausch. "Past, present, and future of user interface software tools." ACM Transactions on Computer-Human Interaction (TOCHI) 7.1 (2000): 3-28.
2. Carpenter, Mark Alan, Kathy Lockaby Khalifa, and David Bruce Lection. "Methods, system and computer program products for delayed message generation and encoding in an intermittently connected data communication system." U.S. Patent No. 5,859,973. 12 Jan. 1999.
3. Naeem, Tahir, and Kok-Keong Loo. "Common security issues and challenges in wireless sensor networks and IEEE 802.11 wireless mesh networks." 3; 1 (2009).
4. Fan, GaoJun, and ShiYao Jin. "Coverage problem in wireless sensor network: A survey." JNW 5.9 (2010): 1033-1040.
5. Minelli, Michael, Michele Chambers, and Ambiga Dhiraj. Big data, big analytics: emerging business intelligence and analytic trends for today's businesses. John Wiley & Sons, 2012.
6. Al-Karaki, Jamal N., and Ahmed E. Kamal. "Routing techniques in wireless sensor networks: a survey." IEEE wireless communications 11.6 (2004): 6-28.
7. Raghunathan, Vijay, et al. "Energy-aware wireless microsensor networks." IEEE Signal processing magazine 19.2 (2002): 40-50.
8. Adapteva, Epiphany Architecture Reference. http://adapteva.com/docs/epiphany_arch_ref.pdf, 2013
9. http://www.zedboard.org/
10. Prongnuch, Sethakarn, and Theerayod Wiangtong. "Heterogeneous Computing Platform for data processing." Intelligent Signal Processing and Communication Systems (ISPACS), 2016 International Symposium on. IEEE, 2016.